Discrimination of Pathogenic Species Using Oligonucleotide Frequencies of Barcoding

A Major Project dissertation submitted

in partial fulfilment of the requirement for the degree of

Master of Technology

In

Bioinformatics

Submitted by

Pratibha

(DTU/12/M.Tech/403)
Delhi Technological University, Delhi, India

Under the supervision of
Dr. Asmita Das

Department of Biotechnology
Delhi Technological University
(Formerly Delhi College of Engineering)
Shahbad Daulatpur, Main Bawana Road,
Delhi-110042, INDIA

CERTIFICATE

This is to certify that the dissertation entitled **Discrimination of pathogenic species using oligonucleotide frequencies of barcoding** submitted by **Pratibha (DTU/12/M.Tech/403)** in the partial fulfilment of the requirements for the award of the degree of Master of Technology, Delhi Technological University (Formerly Delhi College of Engineering, University of Delhi), is an authentic record of the candidate's own work carried out by him/her under my guidance.

The information and data enclosed in this thesis is original and has not been submitted elsewhere for honoring of any other degree.

Date:

Dr. Asmita Das
(Project Mentor)
Department of Bio-Technology
Delhi Technological University
(Formerly Delhi College of Engineering, University of Delhi)

Certificate (on letter head) from the second mentor in case project has been done outside DTU

DECLARATION

I hereby declare that this thesis is my own work and effort and that it has not been submitted anywhere for any award. Where other sources of information have been used, They have been acknowledged

Signature: ……………………………………….

Date: …………………

ACKNOWLEDGEMENT

I take this opportunity to express my profound gratitude and deep regards to my guide Dr Asmita Das and Mr. Sayan Chaterjee for their exemplary guidance, monitoring and constant encouragement throughout the course of this thesis. The blessing, help and guidance given by her time to time shall carry me a long way in the journey of life.

I also take this opportunity to express a deep sense of gratitude to Dr. Yasha Hasija, Dr. Monica Wahi, Dr. Pravir kumar, Dr. Navneeta for their cordial support, valuable information and guidance, which helped me in completing this task through various stages.

Lastly, I thank almighty, my parents, brother, sisters and friends for their constant encouragement without which this assignment would not be possible.

Pratibha
2K12/BIO/21

CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

COI	Cytochrome c oxidase subunit 1
DNA	Deoxy Nucleic Acid
EMBL	European Molecular Biology Lab
FISH-BOL	The Fish Barcode of Life Initiative
NCBI	National Centre for Bioinformatics Information
PCR	Polymerase Chain Reaction
RNA	Ribo Nucleic Acid

Discrimination of pathogenic species using oligonucleotide frequencies of barcoding

Pratibha

Delhi Technological University, Delhi, India

ABSTRACT

Now days, molecular taxonomy has become a more targeted tool to classify enormous diversity of the organisms present on earth. As a part of molecular taxonomy DNA based methods especially DNA bar coding serves as a more precise tool for identification and subsequent targeted techniques. The genuses of worms which are chosen are the popular organisms of various human and animal diseases. Currently, only physiological mode of diagnoses is available to detect the type of infection.

Our proposed technique works on knowledge based methods of diagnosis where the basis of diagnosis is taken as the whole genome of the organism. The characteristic of this sequence based method is based on the nucleotide indices and DNA bar coding techniques. This technique will help in a more precise and rapid diagnosis. This approach has been extended to other pathogenic organisms for accurate diagnosis and personalized treatment measures. The analytical tool has been developed in JAVA for better user interface and ease of use by researchers. Currently the tool is in offline standalone format but it can also be made an online tool hosted by a server with very minor changes.

1. <u>**INTRODUCTION**</u>

DNA barcoding is a method for characterizing types of species of organism utilizing a short DNA sequence chosen from a standard and concurred upon position in the genome. DNA standardized tag successions are short with respect to the whole genome and they might be acquired sensibly rapidly and efficiently. The cytochrome c oxidase subunit 1 mitochondrial area (COI) is developing as the standard standardized tag locale for higher creatures. DNA barcoding was especially helpful for marine living beings (Shander and Willassen, 2005), including fishes (Mason, 2003; Ward et al., 2005); soil microbes (Blaxter et al., 2004) and freshwater meiobenthos (Markmann and Tautz, 2005); and extinct birds (Lambert et al., 2005). In the rainforests, quick DNA-based entomological inventories were so compelling (Monaghan et al., 2005; Smith et al., 2005) that tropical biologists were the most dynamic advocates of DNA barcoding (Janzen, 2004). All the more practically, DNA standardized tags have ended up being valuable in bio security, e.g. for observation of infection vectors (Besansky et al., 2003) and obtrusive bugs (Armstrong and Ball, 2005), and in addition for law authorization and primatology (Lorenz et al., 2005).

A new approach, Oligonucleotide Frequency Range (OFR) of barcode loci for discrimination of species has been proposed. OFR of the loci which discriminates between species is characteristic of a species, i.e., the maxima and minima within a species did not overlap with that of other species. The species resolution ability of the barcode loci using p-distance, Euclidean distance of oligonucleotide frequencies, nucleotide-character based approach and OFR method has been compared in many species. The species resolution by OFR was either higher or comparable to the other methods (Tyagi *et al*,. 2010). We have used the same approach in discriminating the various species of pathogenic worms. The dinucleotide frequency and trinucleotide frequency of the species was calculated and compared for different species.

The diseases and epidemiological conditions caused by worms are as diverged as the types of worms which are responsible for them. The diseases caused by them varies from malaria to eye infections (Nimir *et al*,. 2012). The technique can enable the non-taxonomists to identify these pathogens, thereby helping to understand and curb disease carrying pests and pathogens. The program can be used as a confirmatory test to various diagnostic tools.

2. **REVIEW OF LITERATURE**

2.1 **Divergence in pathogenecity of worms**

According to a report published by NIA, diseases and epidemiological conditions caused by worms are as diverged as the types of worms which are responsible for them. The diseases caused by them varies from malaria to eye infections (Nimir *et al,*. 2012). Worms that behave as parasites come in thousands of different species, including, tapeworms, flatworms, flukes, roundworms and leeches. The worms can be as small as microscopic, or they may be as long as 9 meters.

Some worms cause sore and deforming conditions, while others are simply noticed by the host. Some worm infestations get cleared up after a short time, while others cause long-term problems that affect many distinct body organs and may even result into death.

2.2 **Gene based Diagnosis**

The diagnosis of any pathological condition is essentially based on the microscopic structure of cells and tissues. This continues as the standard procedure through which all other diagnostic tests are measured. Now days, the pathologists are dependent on the examination of the tissue section stained by histochemical methods. This method is supported by the advanced biochemical, immunological and molecular techniques. It is one of the ways that can be used to unravel the molecular mechanism in detecting the disease process. Technologies which are used for studying cellular process are same for both normal and abnormal cell. The cellular process can be analyzed either from protein to gene or from gene to protein. Previously both biochemical analysis and genetic analysis were separate. But in this era with the advances in recombinant DNA technology it has become possible to have a connection between the biochemical and genetic analysis (Premalatha *et al,*. 2014). Recently PCR amplification has been used to detect and compare various species like P. *vivax* and P. *falciparum* (Pattakorn *et al,*. 2010).

2.3 **DNA barcoding**

DNA barcoding was created as of late as a strategy for animal types distinguishing proof over a wide scope of eukaryotes taxa by sequencing an institutionalized short DNA part. Because of present day innovations, it is conceivable to do this with a small bit of any tissue taken from a life form at any formative stage, frequently without harming it. A variable 5' a large portion of mitochondrial gene Co1 is recommended as a standard locale for the greater part of creatures; it is not distinguished yet for organisms and plants. "The Barcode of Life Initiative" infers making and creating the scanner tag library for all the species on Earth to encourage both relegating of recently got examples to the known species and for finding new and obscure species or at any rate their temporary distinguishment. This methodology has an extraordinary potential for the utilization in worldwide biodiversity studies, particularly on account of inadequately explored taxa and situations. The activity being referred to includes finish of another electronic succession database with thorough standards for taxonomic data on the examples and records of their stockpiling and in addition for norms of grouping quality and their passage. Discriminating complaints of rivals to DNA barcoding are audited and additionally impediments of the methodology, the issues to be mulled over, and the fields where it might be utilized. Various late studies on distinctive creature aggregates convincingly show the viability of DNA barcoding and its possibilities. The last relies on upon accessibility of extensive and fair-minded reference database inferring right distinguishing proof of the source examples and sufficient information of intraspecies variety, so the Barcode Initiative would be more effective as a piece of the integrative examination of the taxa being bar-coded (Shneer, VS. 2009).

Species identification is an elementary part in recognizing and classifying biodiversity. Traditionally, identification is based on morphological features provided by taxonomic studies. Only taxonomy experts and trained technicians can identify taxa accurately, because it requires extensive experience and special skills. Moreover the classical method of species classification suffers from various drawbacks. Firstly, the two prime characters employed for species recognition viz phenotypic plasticity and genetic variability can lead to inaccurate identifications. Secondly, the approach neglects morphologically cryptic taxa that are common in several groups (Knowlton 1993; Jarman & Elliott 2000). Thirdly, as morphological keys are often valid only for a specific life stage or gender, many individuals cannot be identified. Finally, although new interactive versions serve as a major advance, the use of keys often calls for such a high level of expertise that misdiagnoses are frequent. (Herbert et al., 2003).

One of the most promising ways is the use of molecular in place of morphological data for taxa identification, which has been a fundamental idea of many biologists from a long period of time (Busse et al. 1996; Blaxter 2003). DNA barcoding, developed in 2003 to identify species, has helped to rejuvenate taxonomic research. DNA barcoding is a technique for identifying organisms based on a short, standardized fragment of genomic DNA. As a Linnaean binomial is an abbreviated label for the morphology of a species, the short sequence is an abbreviated label for the genome of the species (Vernooy et al., 2010). DNA barcoding allows researchers to develop a system for species identification based on digital characters, ultimately allowing for automated identifications, therefore promising to raise the capacity to identify, monitor, and manage biodiversity, with subtle societal and economic benefits. It also lifts the possibility of identifying the vectors of zoonotic diseases as well as the disease organisms themselves.

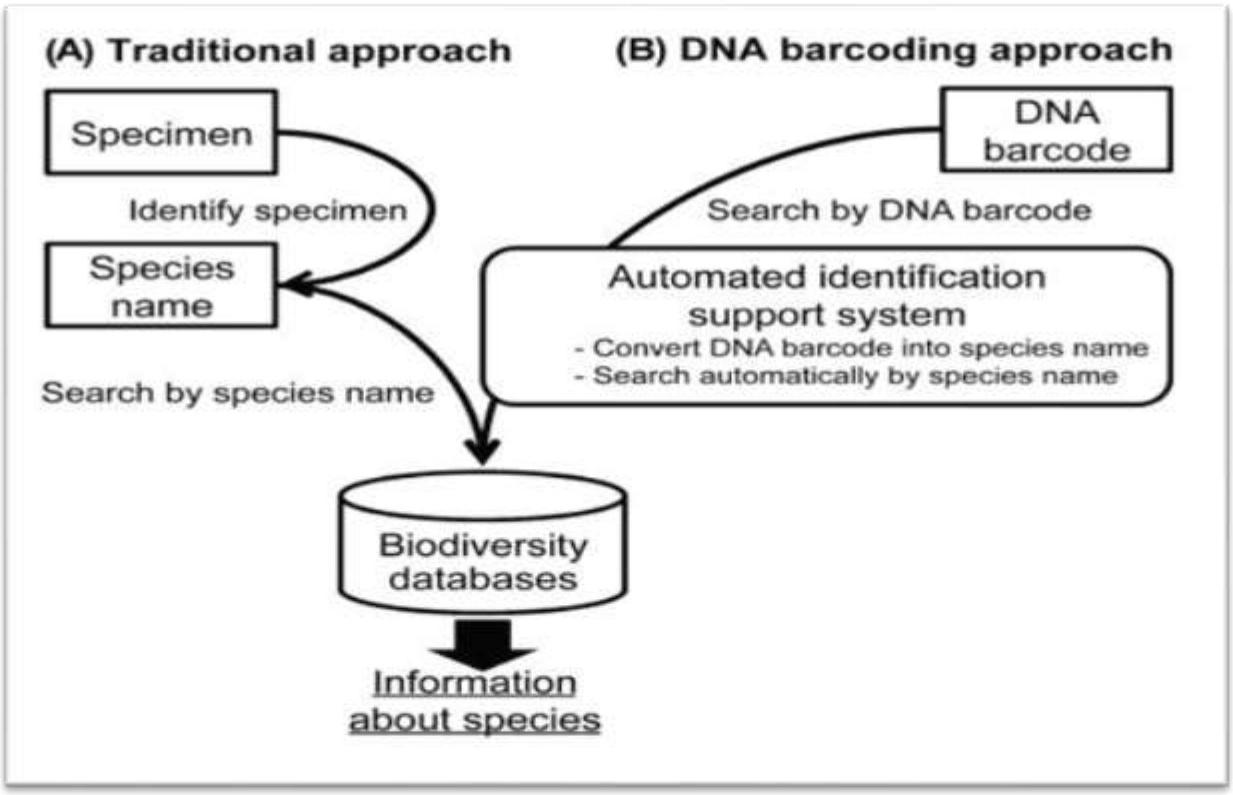

(A) Traditional approach
(B) DNA barcoding approach
Specimen
Identify specimen
Species name
Search by species name
DNA barcode
Search by DNA barcode
Automated identification support system
- Convert DNA barcode into species name
- Search automatically by species name
Biodiversity databases
Information about species

Figure 1: Workflow of traditional and DNA barcode approach of species identification

The 13 protein-coding genes in the animal mitochondrial genome are better targets because indels are rare since most lead to a shift in the reading frame. There is no compelling a priori reason to focus analysis on a specific gene, but the cytochrome c oxidase I gene (COI) does have two important advantages. First, the universal primers for this gene are very robust, enabling recovery of its 59 end from representatives of most, if not all, animal phyla (Folmer et al. 1994; Zhang & Hewitt 1997). Second, COI appears to possess a greater range of phylogenetic signal than any other mitochondrial gene. In common with other protein coding genes, its third-position nucleotides show a high incidence of base substitutions, leading to a rate of molecular evolution that is about three times greater than that of 12S or 16S rDNA (Knowlton & Weigt 1998). In fact, the evolution of this gene is rapid enough to allow the discrimination of not only closely allied species, but also phylogeographic groups within a single species (Cox & Hebert 2001; Wares & Cunningham 2001). Although COI may be matched by other mitochondrial genes in resolving such cases of recent divergence, this gene is more likely to provide deeper phylogenetic insights than alternatives such as cytochrome b (Simmons & Weller 2001).

2.3.1**Criteria to Select Barcode Gene**

- Universability: The gene used for barcoding should be present in a wide range of taxa.
- Specificity to variation : it should have high variability between species but should be conserved within the species so that intraspecific variation becomes insignificant
- Easiness on Employment: the gene should be retrievable by single primer pair and should be amenable to bidirectional sequencing. The gene should be short.

2.3.2 <u>DNA Barcodes</u>

- For Plants:

  Two regions of chloroplast DNA, ribulose–bisphosphate carboxylase (rbcL) and maturase K (matK) are used as standard barcodes in plants
- For Fungi:

  Internal transcribed spacer (ITS) are used as barcode in Fungi
- For Animals:

  A ~648 base-pair region of the mitochondrial cytochrome c oxidase subunit I (COI) gene is used as barcode in animals.

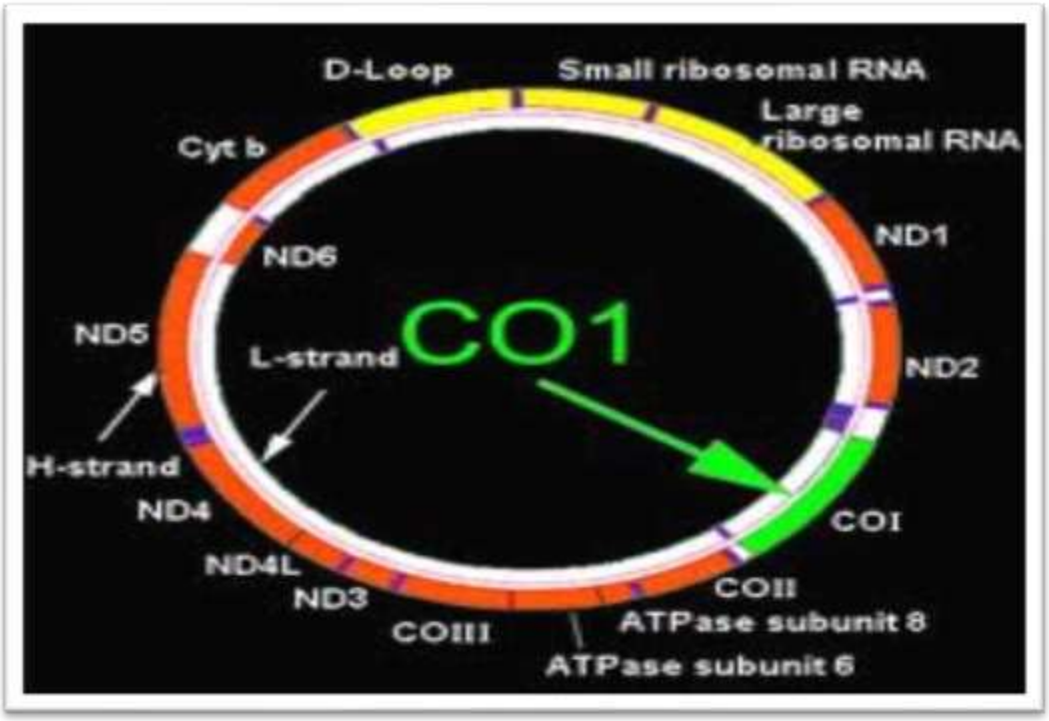


Figure 2: Depicting COI gene in mitochondria

2.3.3 <u>**Advantages of using COI gene as barcode**</u>

Mitochondrial genome is especially suitable for identifying species because of its high copy number, greater differences among species, few differences within species and absence of introns. COI is used as a gold standard for barcode because it is flanked by conserved regions and has a limited exposure to recombination. Moreover rate of molecular evolution is high in COI and it lacks indels.

2.3.4 **DNA Barcoding in different species**

DNA barcoding has also been used to identify marine metazoan species. More than 230,000 known species speaking to 31 metazoan phyla populate the world's seas. Maybe an alternate 1,000,000 or more species stay to be uncovered. There is a worry that species terminations may out-pace disclosure, particularly in assorted and jeopardized marine territories, for example, coral reefs. DNA standardized tags are helpful instruments to quicken species-level examination of marine biodiversity and to encourage preservation deliberations. The method used is based on standardized identification of metazoans using a 648 base-pair sequence of the mitochondrial cytochrome c oxidase subunit I (COI) gene. Scanner tags have additionally been utilized for populace hereditary and phylogeographic dissection, identification of prey in gut substance, location of obtrusive species, and fish wellbeing. All the more questionably, standardized identifications have been utilized to delimit species limits, uncover secretive species, and run across new species. Today there is a concern on the utilization of standardized tags for fast and progressively robotized biodiversity appraisal by high-throughput sequencing, including ecological barcoding and the utilization of standardized tags to locate species for which formal distinguishing features may never be conceivable (Bucklin *et al*,. 2010).

FISH-BOL is a deliberate worldwide examination venture initiated in 2005, with the objective to gather and analyse the associated DNA standardized tag arrangements and related voucher provenance information in a curated reference succession library to support the sub-atomic distinguishing proof of all fish species. Of the more or less 31,000 presently known fish species, 25% have been transformed effectively, with no less than one animal varieties from 89% of all families bar-coded so far(Becker *et al*,. 2011).

Fungal research is encountering another wave of methodological upgrades that most likely will help mycology as significantly as atomic phylogeny has done amid the most recent 15 years. Particularly the next generation sequencing advances might be relied upon to have a huge impact on fungal biodiversity and biology research. So as to understand the true ability of these promising strategies by quickening biodiversity appraisals, identification methods of parasites need to be adjusted to the developing requests of advanced huge scale biological studies. While the response may appear unimportant to most microbiologists, taxonomists working with organisms may have different perspectives. Barcoding has been used successfully in fungal research and has identified a number of species (Begerow *et al*,. 2011).

DNA barcoding was proposed as a strategy for identification of eukaryotic species through correlation of successions of a standard short DNA part -DNA standardized identification - from an obscure example to a library of reference arrangements from known species. These permits recognizing an organic entity at any phase of improvement from a little tissue test, fresh or saved numerous years ago. Molecular identification proof of plant specimens might be utilized as a part of different investigative and connected fields. It would likewise help to discover new species, which is especially helpful for cryptogamic plants. An ideal DNA scanner tag locale is a little section exhibit in all types of a real taxonomic gathering, having perpetual nucleotide arrangement in all parts of the same species, yet with sufficient variety to segregate among the species. This part ought to be flanked by low-variable sequences for utilization of general first stages in PCR for intensification and sequencing. The DNA standardized identification that is entrenched in creatures is a grouping of a part of the mitochondrial cytochrome c oxidase gene Co1. Be that as it may, hunting down DNA standardized identification in plants turned out to be an additionally difficult assignment. No DNA locale generally suitable for all plants and gathering the majority of the vital criteria has been found. Evidently, a multilocus or two-stage methodology ought to be requisitioned this reason. A few sections of the chloroplast genome (trnh-psba, matk, rpoc, rpob, rbcl) in consolidations of a few areas were proposed as applicant districts with most noteworthy potential, yet more illustrative specimens ought to be analyzed to pick the best competitor. The likelihood is examined to use as DNA standardized tag inner translated spacers (ITS) of atomic rRNA genes, which are profoundly variable, generally utilized in sub-atomic phylogenetic studies at the species level, additionally have a few impediments (Shneer, VS. 2009).

3.1**METHODOLOGY**

1
- Retrieved mitochondrial sequences of various worm genera from NCBI genome database

2
- The sequences were aligned with known COI gene using EMBL align tool

3
- Dinucleotide and trinucleotide frequencies were calculated by Java program.

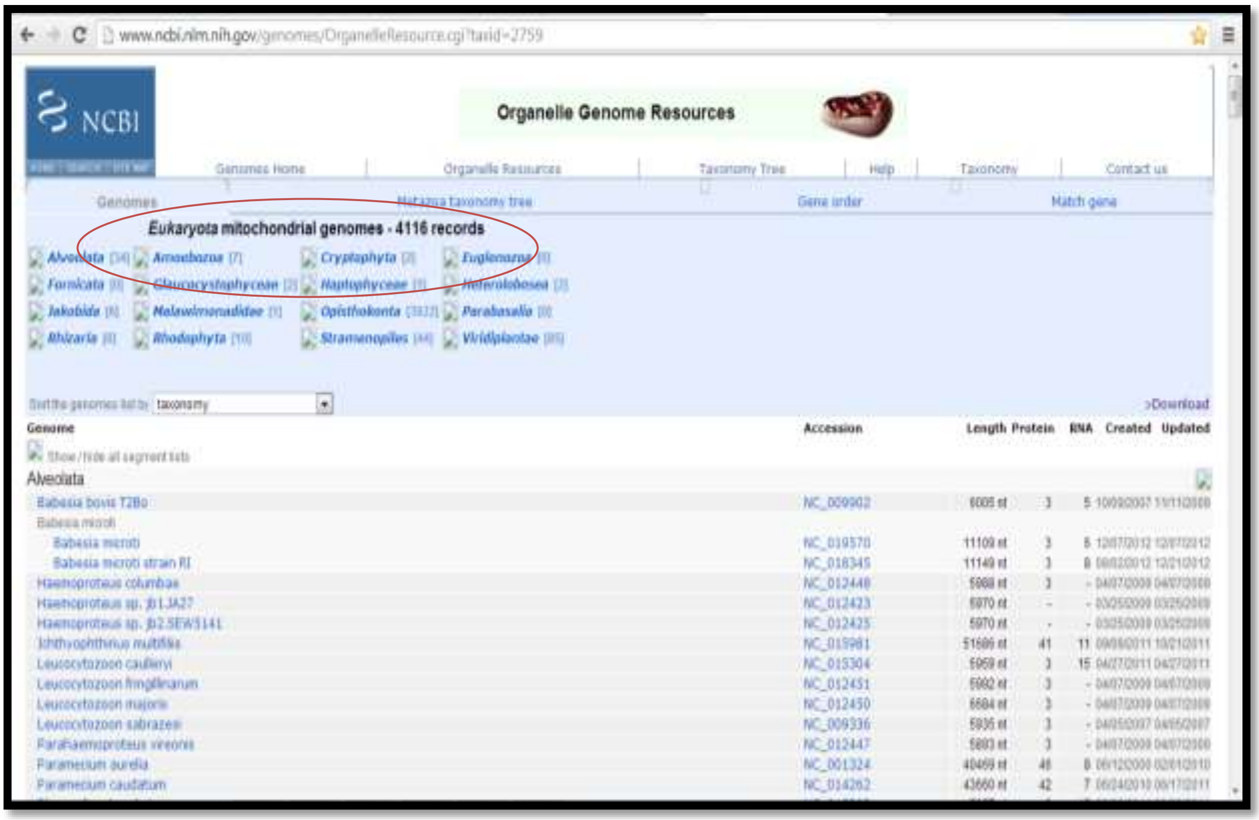
4
- Variance and oligonucleotide frequency difference of various species was calculated.

5
- The oligonucleotide frequencies of different species was compared.

6
- Statistical tests were done to find inter and intra specific differences.

3.1 **Sequence retrieval**

The whole mitochondrial genome of different species of worms was retrieved from organelle resource database of NCBI.



Eukaryota mitochondrial genomes - 4116 records

Alveolata [34] *Amoebozoa* [7] *Cryptophyta* [2] *Euglenozoa* [1]
Fornicata [1] *Glaucocystophyceae* [2] *Haptophyceae* [1] *Heterolobosea* [2]
Jakobida [6] *Malawimonadidae* [1] *Opisthokonta* [3932] *Parabasalia* [0]
Rhizaria [0] *Rhodophyta* [10] *Stramenopiles* [44] *Viridiplantae* [85]

Sort the genomes list by: taxonomy

>Download

Genome	Accession	Length	Protein	RNA	Created	Updated
Alveolata						
Babesia bovis T2Bo	NC_009902	6005 nt	3	5	10/09/2007	11/11/2008
Babesia microti						
Babesia microti	NC_019570	11109 nt	3	6	12/07/2012	12/07/2012
Babesia microti strain RI	NC_018345	11149 nt	3	8	08/02/2012	12/21/2012
Haemoproteus columbae	NC_012448	5988 nt	3	-	04/07/2009	04/07/2009
Haemoproteus sp. jb1.JA27	NC_012423	5970 nt	-	-	03/25/2009	03/25/2009
Haemoproteus sp. jb2.SEW5141	NC_012425	5970 nt	-	-	03/25/2009	03/25/2009
Ichthyophthirius multifiliis	NC_015981	51686 nt	41	11	09/08/2011	10/21/2011
Leucocytozoon caulleryi	NC_015304	5959 nt	3	15	04/27/2011	04/27/2011
Leucocytozoon fringillinarum	NC_012451	5982 nt	3	-	04/07/2009	04/07/2009
Leucocytozoon majoris	NC_012450	6684 nt	3	-	04/07/2009	04/07/2009
Leucocytozoon sabrazesi	NC_009336	5935 nt	3	-	04/05/2007	04/05/2007
Parahaemoproteus vireonis	NC_012447	5883 nt	3	-	04/07/2009	04/07/2009
Paramecium aurelia	NC_001324	40469 nt	46	8	06/12/2000	02/01/2010
Paramecium caudatum	NC_014262	43660 nt	42	7	06/24/2010	08/17/2011

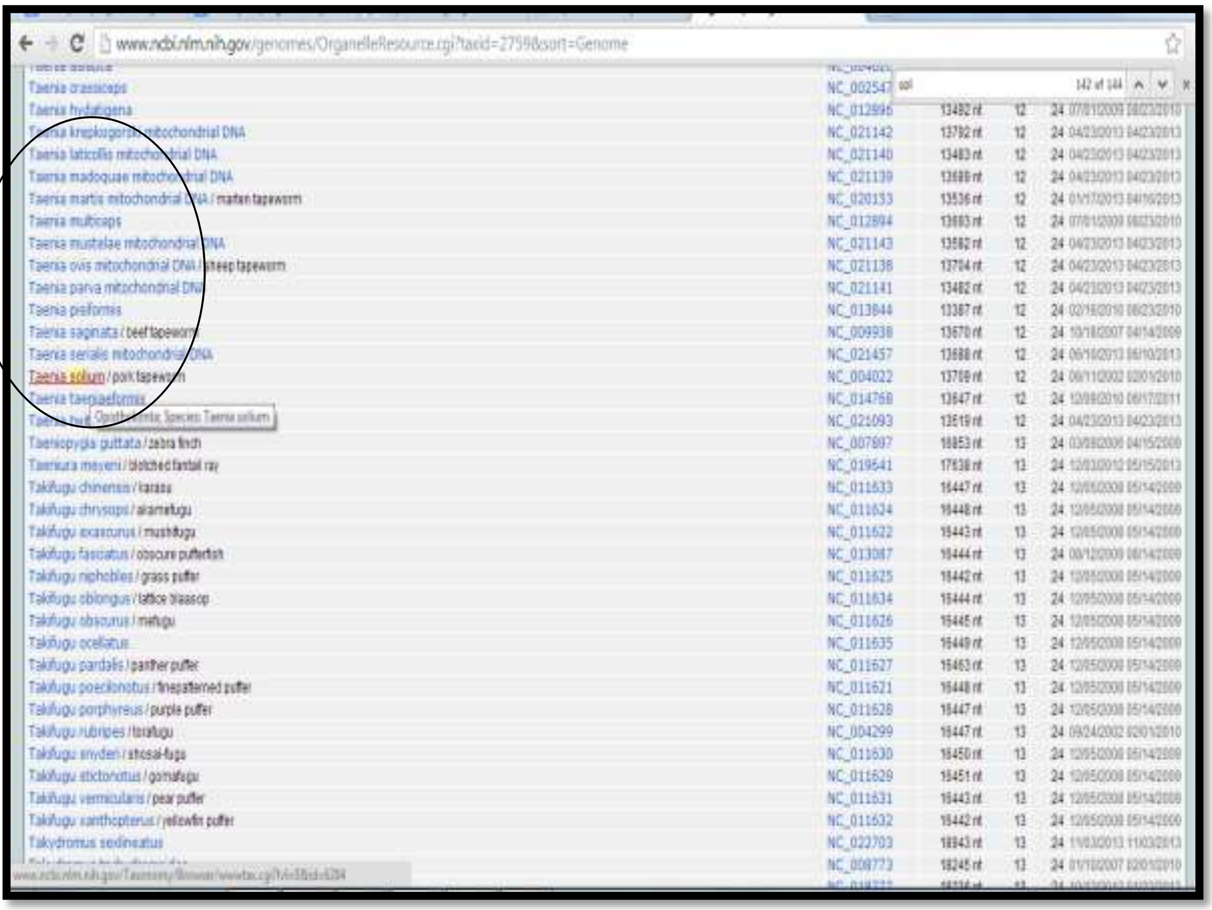
Fig 3: Depicting Organelle resource database of NCBI

Different worm genres were identified and retrieved using their accession numbers. The number of characters or base pairs present in the mitochondrial genome was mentioned in the database.

S no.	Genus	Species
1	*Angiostrongylus*	*Angiostrongylus cantonensis* *Angiostrongylus costaricensis* *Angiostrongylus vasorum*
2	*Diplogonoporus*	*Diplogonoporus balaenopterae* *Diplogonoporus balaenopterae*
3	*Dictyocaulus*	*Dictyocaulus eckerti* *Dictyocaulus viviparous*
4	*Ascaris*	*Ascaris lumbricoides* *Ascaris suum*
5	*Baylisascaris*	*Baylisascaris ailuri* *Baylisascaris procyonis* *Baylisascaris schroederi* *Baylisascaris schroederi*
6	*Toxocara*	*Toxocara canis* *Toxocara cati* *Toxocara malaysiensis*
7	*Taenia*	*Taenia asiatica* *Taenia crassiceps* *Taenia hydatigena* *Taenia krepkogorski* *Taenia laticollis* *Taenia madoquae* *Taenia martis* *Taenia multiceps* *Taenia mustelae* *Taenia ovis*

		Taenia parva *Taenia pisiformis* *Taenia saginata* *Taenia serialis* *Taenia solium* *Taenia twitchelli* *Taeniopygia guttata*

Table 1 List of different worm genera



www.ncbi.nlm.nih.gov/genomes/OrganelleResource.cgi?taxid=2759&sort=Genome

sol 142 of 144

Taenia crassiceps	NC_002547					
Taenia hydatigena	NC_012896	13492 nt	12	24	07/01/2009	08/23/2010
Taenia krepkogorski mitochondrial DNA	NC_021142	13792 nt	12	24	04/23/2013	04/23/2013
Taenia laticollis mitochondrial DNA	NC_021140	13483 nt	12	24	04/23/2013	04/23/2013
Taenia madoquae mitochondrial DNA	NC_021139	13689 nt	12	24	04/23/2013	04/23/2013
Taenia martis mitochondrial DNA / marten tapeworm	NC_020133	13536 nt	12	24	01/17/2013	04/16/2013
Taenia multiceps	NC_012894	13693 nt	12	24	07/01/2009	08/23/2010
Taenia mustelae mitochondrial DNA	NC_021143	13582 nt	12	24	04/23/2013	04/23/2013
Taenia ovis mitochondrial DNA / sheep tapeworm	NC_021138	13704 nt	12	24	04/23/2013	04/23/2013
Taenia parva mitochondrial DNA	NC_021141	13482 nt	12	24	04/23/2013	04/23/2013
Taenia pisiformis	NC_013844	13387 nt	12	24	02/16/2010	08/23/2010
Taenia saginata / beef tapeworm	NC_009938	13670 nt	12	24	10/18/2007	04/14/2009
Taenia serialis mitochondrial DNA	NC_021457	13688 nt	12	24	06/10/2013	06/10/2013
Taenia solium / pork tapeworm	NC_004022	13709 nt	12	24	06/11/2002	02/01/2010
Taenia taeniaeformis	NC_014768	13647 nt	12	24	12/09/2010	06/17/2011
Taenia twit[illegible]	NC_021093	13519 nt	12	24	04/23/2013	04/23/2013
Taeniopygia guttata / zebra finch	NC_007897	16853 nt	13	24	03/08/2006	04/15/2009
Taeniura meyeni / blotched fantail ray	NC_019641	17638 nt	13	24	12/03/2012	05/15/2013
Takifugu chinensis / karasu	NC_011633	16447 nt	13	24	12/05/2008	05/14/2009
Takifugu chrysops / akamefugu	NC_011624	16448 nt	13	24	12/05/2008	05/14/2009
Takifugu exascurus / mushifugu	NC_011622	16443 nt	13	24	12/05/2008	05/14/2009
Takifugu fasciatus / obscure pufferfish	NC_013087	16444 nt	13	24	08/12/2009	08/14/2009
Takifugu niphobles / grass puffer	NC_011625	16442 nt	13	24	12/05/2008	05/14/2009
Takifugu oblongus / lattice blaasop	NC_011634	16444 nt	13	24	12/05/2008	05/14/2009
Takifugu obscurus / mefugu	NC_011626	16445 nt	13	24	12/05/2008	05/14/2009
Takifugu ocellatus	NC_011635	16449 nt	13	24	12/05/2008	05/14/2009
Takifugu pardalis / panther puffer	NC_011627	16463 nt	13	24	12/05/2008	05/14/2009
Takifugu poecilonotus / finepatterned puffer	NC_011621	16448 nt	13	24	12/05/2008	05/14/2009
Takifugu porphyreus / purple puffer	NC_011628	16447 nt	13	24	12/05/2008	05/14/2009
Takifugu rubripes / torafugu	NC_004299	16447 nt	13	24	09/24/2002	02/01/2010
Takifugu snyderi / shosai-fugu	NC_011630	16450 nt	13	24	12/05/2008	05/14/2009
Takifugu stictonotus / gomafugu	NC_011629	16451 nt	13	24	12/05/2008	05/14/2009
Takifugu vermicularis / pear puffer	NC_011631	16443 nt	13	24	12/05/2008	05/14/2009
Takifugu xanthopterus / yellowfin puffer	NC_011632	16442 nt	13	24	12/05/2008	05/14/2009
Takydromus sexlineatus	NC_022703	16943 nt	13	24	11/03/2013	11/03/2013
[illegible]	NC_008773	18245 nt	13	24	01/10/2007	02/01/2010

Opisthokonta; Species: Taenia solium

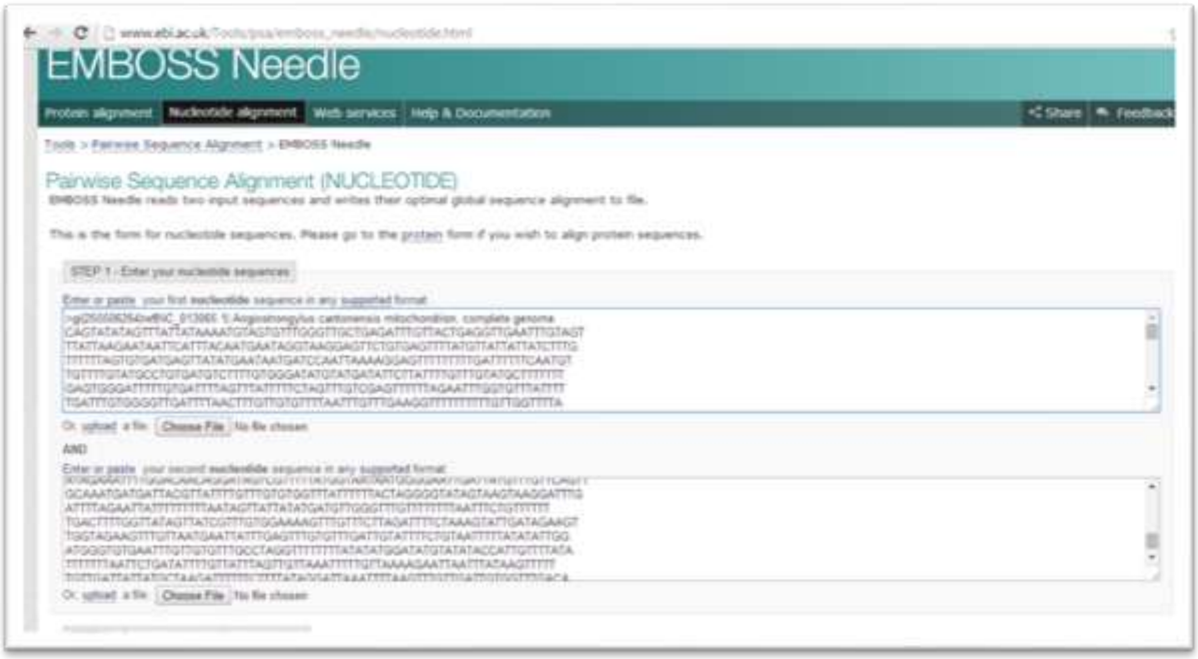
www.ncbi.nlm.nih.gov/Taxonomy/Browser/wwwtax.cgi?id=6204

Fig 4: Depicts mitochondrial genome retrieval of *taenia* genus

The fasta sequence of all the species listed above was downloaded from NCBI database. Different folders were made for different genus.

3.2 **Extraction of COI gene**

The whole sequence of COI gene (1527 bp) and partial sequence of COI gene (327 bp) was retrieved from NCBI database. The above sequences were aligned with both COI gene sequences using EMBL alignment tools.


www.ebi.ac.uk/Tools/psa/emboss_needle/nucleotide.html
EMBOSS Needle
Protein alignment
Nucleotide alignment
Web services
Help & Documentation
Share
Feedback
Tools > Pairwise Sequence Alignment > EMBOSS Needle

Pairwise Sequence Alignment (NUCLEOTIDE)

EMBOSS Needle reads two input sequences and writes their optimal global sequence alignment to file.

This is the form for nucleotide sequences. Please go to the protein form if you wish to align protein sequences.

STEP 1 - Enter your nucleotide sequences

Enter or paste your first nucleotide sequence in any supported format:

```
>gi|[illegible]|ref|NC_013065.1| Angiostrongylus cantonensis mitochondrion, complete genome
[illegible]
```

Or, upload a file: Choose File No file chosen

AND

Enter or paste your second nucleotide sequence in any supported format:

```
[illegible]
```

Or, upload a file: Choose File No file chosen

Figure 5 : Depicting EMBOSS Needle alignment.

3.3 <u>Discrimination of species using java program</u>

The selected values in the combo boxes are stored in two string variables named as s and t respectively. After getting the name of the species from the combo boxes, the corresponding DNA sequence is taken using the if else statement and the sequence is stored in a temporary string variable q. The string q is then processed for the functions like calculating frequencies.

3.3.1 <u>Oligonucleotide frequency calculation</u>

➢ For dinucleotide frequencies :

- Sixteen integer variables namely AA,AC,AG,AT,……,TT are declared and initialized to zero.
- The sequence is then parsed, and as any of the variables is encountered the value of that particular variable is incremented by 1 using the urinary operator "++".
- The variables are then divided by total length of the sequence.
- Then the results are rounded of and stored in an array and displayed on the screen.

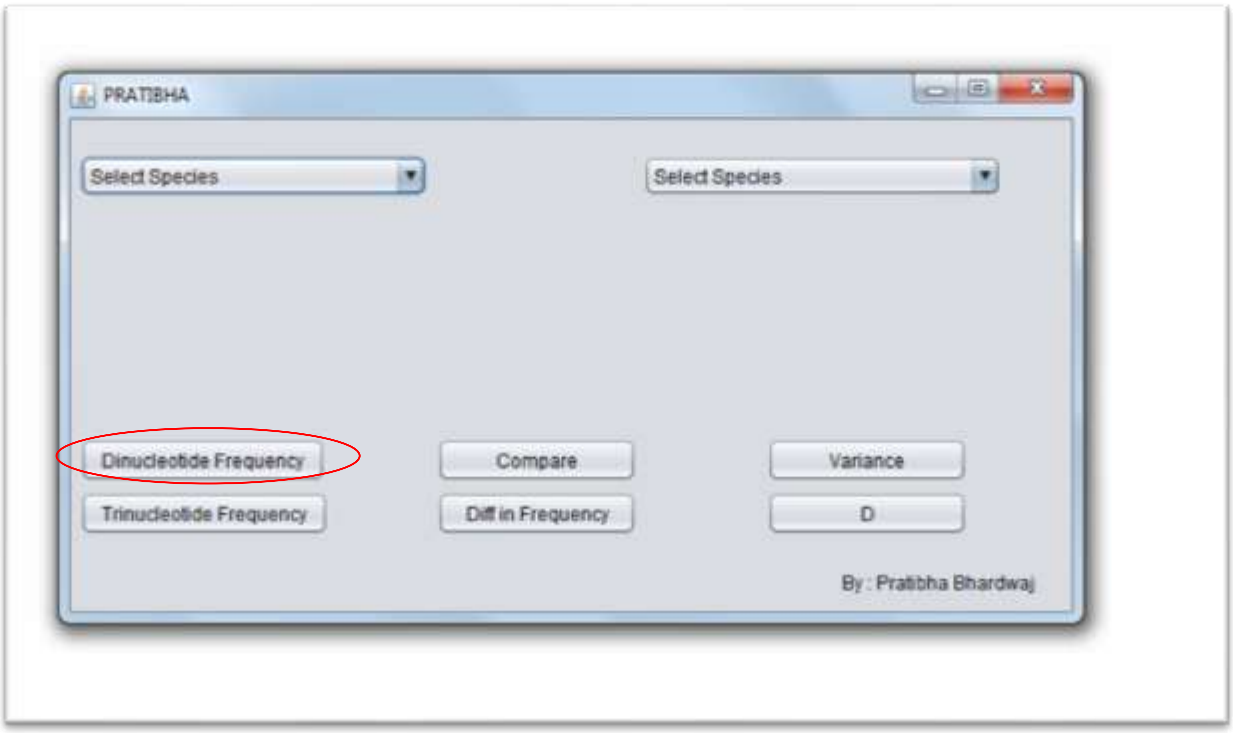


Figure 6: Depicting home screen with option for Dinucleotide Frequency.

- For trinucleotide frequency :
  - For trinucleotide frequencies sixty four integer variables namely AAA,AAC,……..,TTT are declared and initialised to zero.
  - The sequence is then parsed and as any of the variables is encountered the value of that particular variable is incremented by 1 using the urinary operator "++".
  - The variables are then divided by total length of the sequence.
  - Then the results are rounded of and stored in an array and displayed on the screen.

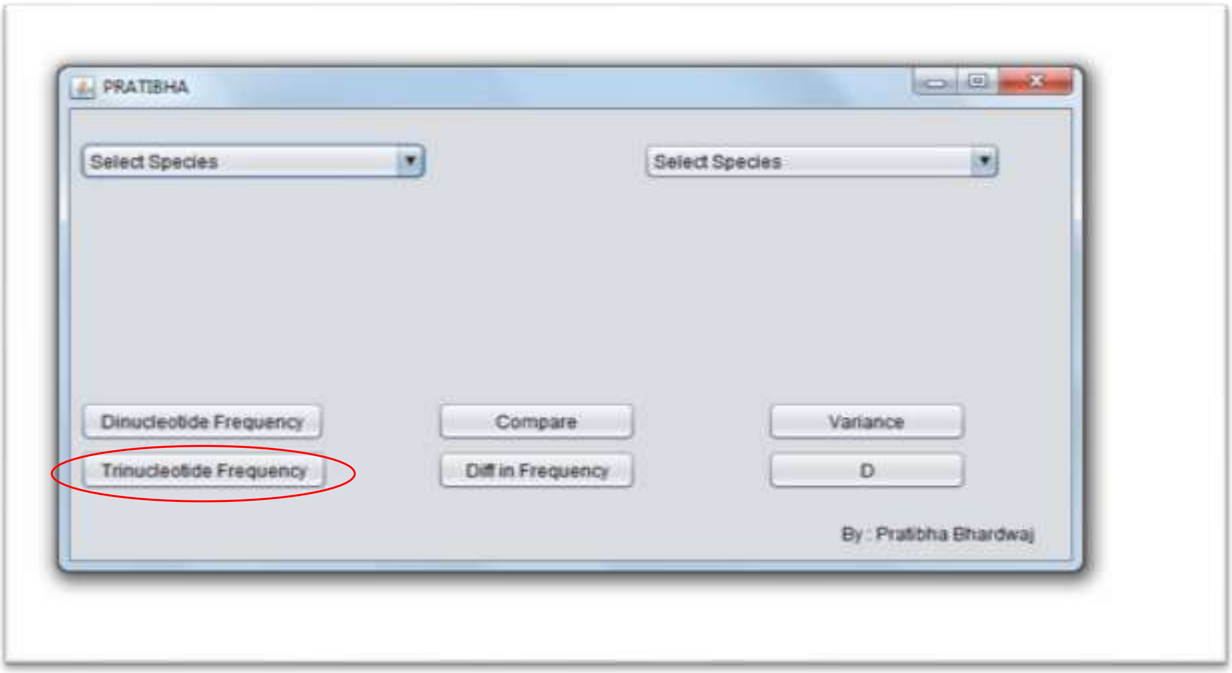


Figure7: Depicting home screen with option for Trinucleotide Frequency.

3.3.2 <u>Variance calculation</u>

- First the di and trinucleotide frequencies are obtained.
- The frequencies are then stored in two different arrays.
- Then the mean for di and tri nucleotide frequencies is obtained separately.
- The mean for dinucleotide frequencies is then subtracted from them and the mean for trinucleotide frequencies is subtracted from them.
- The sum of all the elements of array is then obtained for both the arrays.
- The sum is then divided by the no. of elements in the array.
- This result is the variance.
- This is then displayed on the screen.

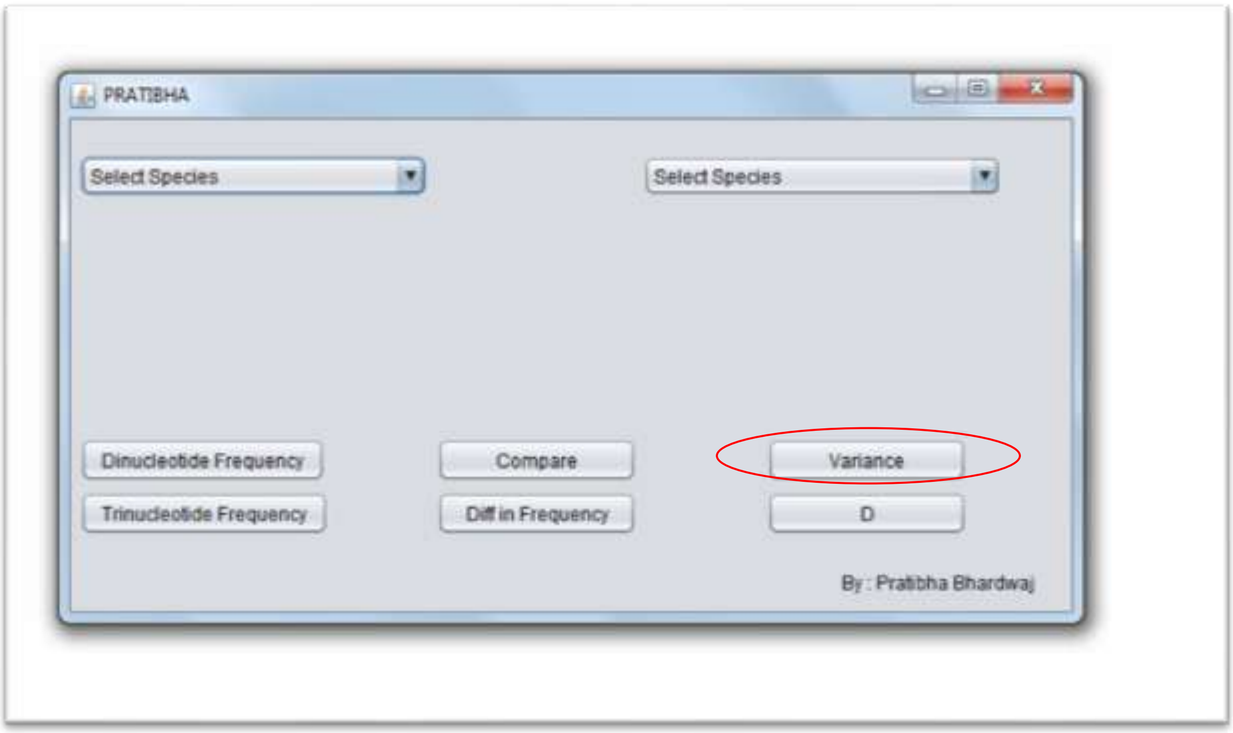


Figure 8: Depicting home screen with option for Variance.

3.3.3 <u>Calculation of Oligonucleotide frequency difference</u>

- This button allows user to know the difference in the maximum and minimum di and trinucleotide frequencies of the species.
- First the di and trinucleotide frequencies are obtained.
- Then four variables are declared namely max1, max2, min1, min2.
- Max1 represents the maximum dinucleotide frequency, min1 represents the minimum dinucleotide frequency.
- Max2, Min2 in the same way represent the maximum and minimum trinucleotide frequencies.
- The differences are then obtained by subtracting min1 from max1 and min2 from max 2.
- The differences are then displayed on the screen.

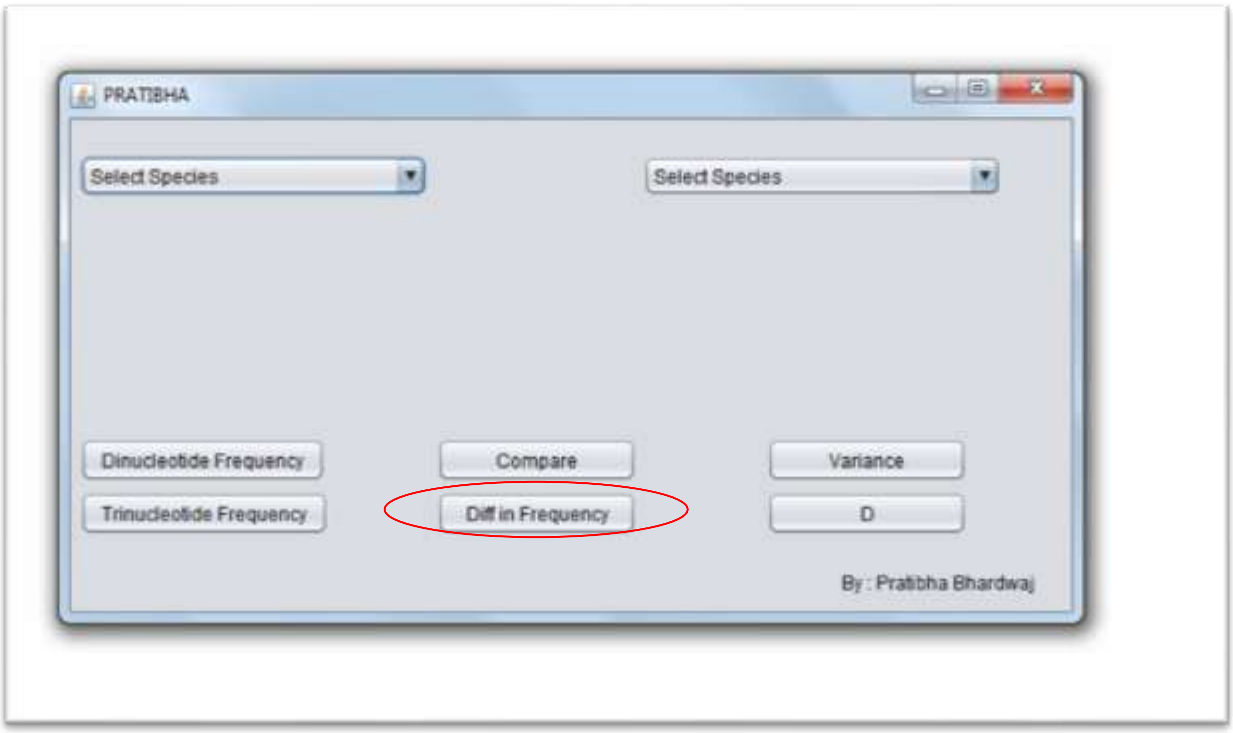


Figure9: Depicting home screen with option for Difference in Frequency.

3.3.4 Comparison of Oligonucleotide frequency

- For comparing two species, it plots two graphs between the di and tri nucleotide frequencies of the two species.
- The di and trinucleotide frequencies for the two species are first obtained by the same procedure as discussed above.
- Then datasets are created for both the species.
- Then using the "jfree chart" library provided by the java, the graphs are displayed.

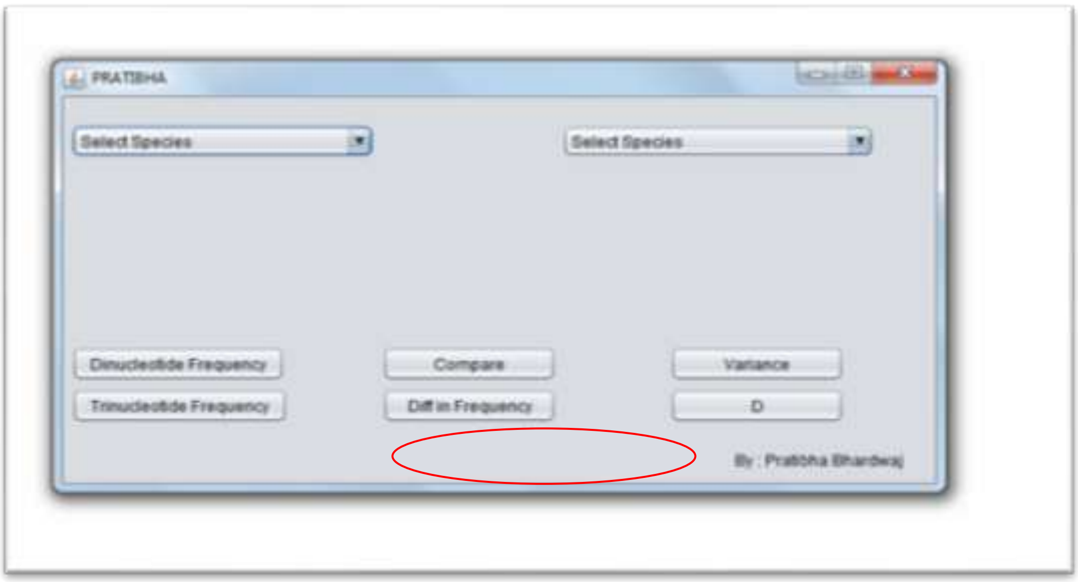


Figure10: Depicting home screen with option for comparison.

3.3.5 Statistical Analysis

The Euclidean distances (D) based on oligonucleotide frequency differences were calculated as follows.

$$D = \sqrt{\sum_{i=1}^{N} |F1 - F2|^2}$$

Where, N is the number of oligonucleotides, F1 and F2 represent the frequency of each type of oligonucleotide for species 1 and 2 respectively. Each distance was calculated from di- and trinucleotide frequencies.

- First the di and trinucleotide frequencies are obtained, and stored in arrays.
- The dinucleotide frequencies of one sequence are subtracted from the other.
- In the same way trinucleotide for one sequence are subtracted from the other.
- Then the sum is obtained for both the arrays.
- The sum is then rounded off and displayed.

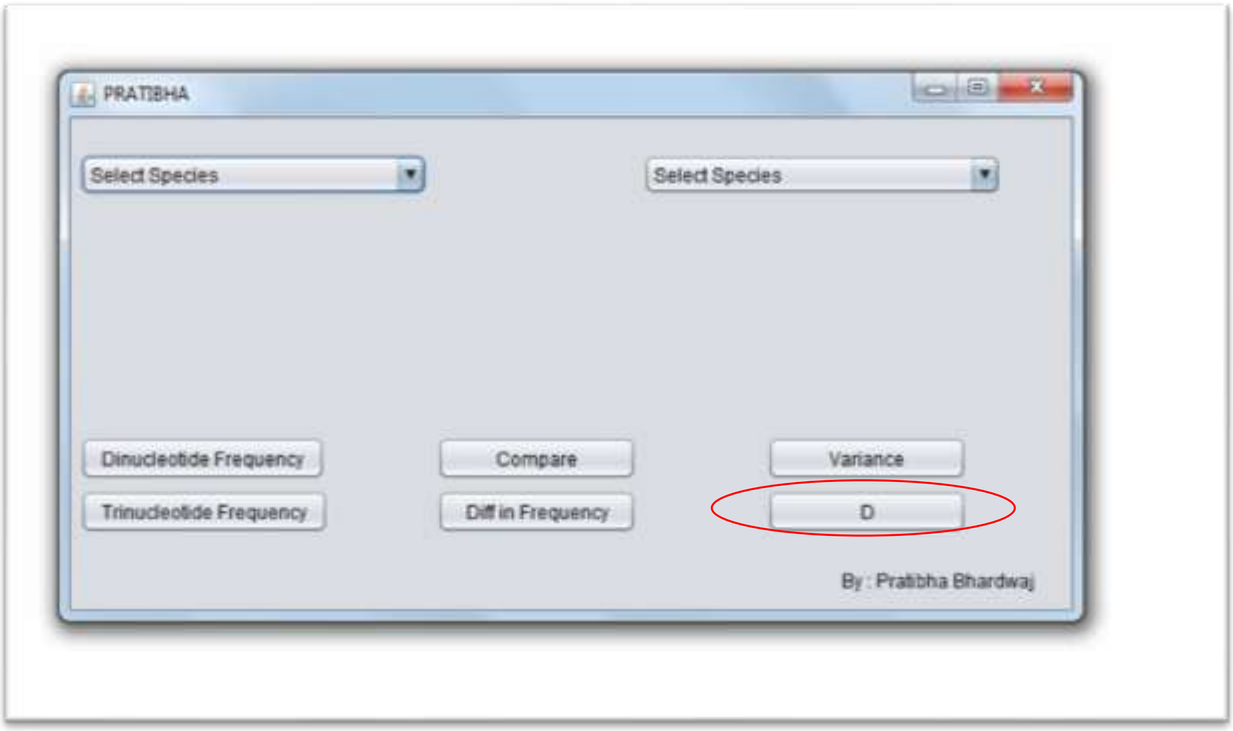
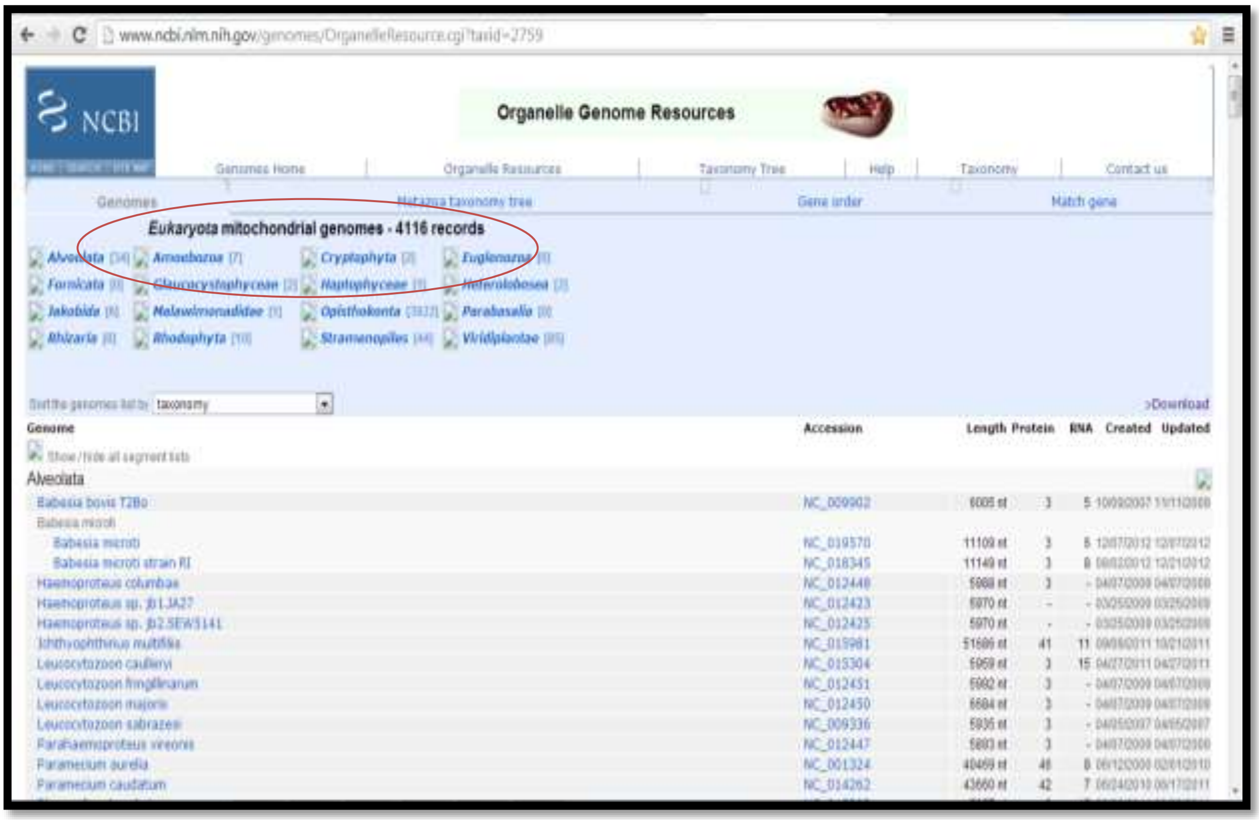


Figure11: Depicting home screen with option for D calculation.

4. **RESULTS**

4.1**Sequence retrieval**

The whole mitochondrial genome of different species of worms was retrieved from organelle resource database of NCBI.



www.ncbi.nlm.nih.gov/genomes/OrganelleResource.cgi?taxid=2759

Organelle Genome Resources

Genomes Home | Organelle Resources | Taxonomy Tree | Help | Taxonomy | Contact us

Genomes | Metazoa taxonomy tree | Gene order | Match gene

Eukaryota **mitochondrial genomes - 4116 records**

Alveolata [34] ***Amoebozoa*** [7] ***Cryptophyta*** [2] ***Euglenozoa*** [0]
Fornicata [0] ***Glaucocystophyceae*** [2] ***Haptophyceae*** [1] ***Heterolobosea*** [2]
Jakobida [8] ***Malawimonadidae*** [1] ***Opisthokonta*** [3822] ***Parabasalia*** [0]
Rhizaria [0] ***Rhodophyta*** [10] ***Stramenopiles*** [44] ***Viridiplantae*** [85]

Sort the genomes list by: taxonomy

>Download

Show / hide all segment lists

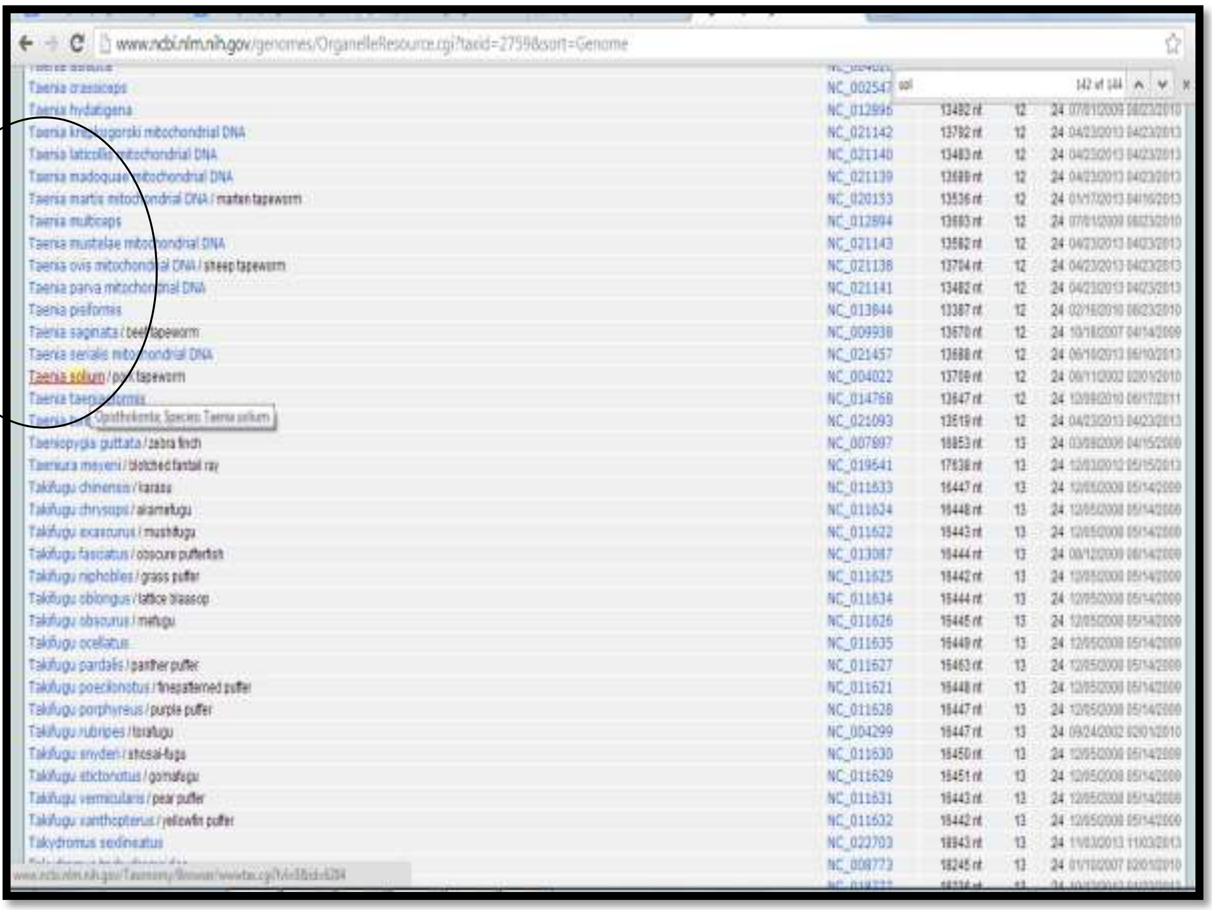
Genome	Accession	Length	Protein	RNA	Created	Updated
Alveolata						
Babesia bovis T2Bo	NC_009902	6005 nt	3	5	10/09/2007	11/11/2008
Babesia microti						
Babesia microti	NC_019570	11109 nt	3	5	12/07/2012	12/07/2012
Babesia microti strain RI	NC_018345	11149 nt	3	8	08/02/2012	12/21/2012
Haemoproteus columbae	NC_012448	5988 nt	3	-	04/07/2009	04/07/2009
Haemoproteus sp. jb1.JA27	NC_012423	5970 nt	-	-	03/25/2009	03/25/2009
Haemoproteus sp. jb2.SEW5141	NC_012425	5970 nt	-	-	03/25/2009	03/25/2009
Ichthyophthirius multifiliis	NC_015981	51686 nt	41	11	09/06/2011	10/21/2011
Leucocytozoon caulleryi	NC_015304	5959 nt	3	15	04/27/2011	04/27/2011
Leucocytozoon fringillinarum	NC_012451	5982 nt	3	-	04/07/2009	04/07/2009
Leucocytozoon majoris	NC_012450	5984 nt	3	-	04/07/2009	04/07/2009
Leucocytozoon sabrazesi	NC_009336	5936 nt	3	-	04/05/2007	04/05/2007
Parahaemoproteus vireonis	NC_012447	5983 nt	3	-	04/07/2009	04/07/2009
Paramecium aurelia	NC_001324	40469 nt	48	8	06/12/2000	02/01/2010
Paramecium caudatum	NC_014262	43660 nt	42	7	06/24/2010	06/17/2011

Figure12: Depicting Organelle resource database of NCBI

Different worm genuses were identified and retrieved using their accession numbers. The number of characters or base pairs present in the mitochondrial genome was mentioned in the database.

S no.	Genus	Species
1	*Angiostrongylus*	*Angiostrongylus cantonensis* *Angiostrongylus costaricensis* *Angiostrongylus vasorum*
2	*Diplogonoporus*	*Diplogonoporus balaenopterae* *Diplogonoporus balaenopterae*
3	*Dictyocaulus*	*Dictyocaulus eckerti* *Dictyocaulus viviparous*
4	*Ascaris*	*Ascaris lumbricoides* *Ascaris suum*
5	*Baylisascaris*	*Baylisascaris ailuri* *Baylisascaris procyonis* *Baylisascaris schroederi* *Baylisascaris schroederi*
6	*Toxocara*	*Toxocara canis* *Toxocara cati* *Toxocara malaysiensis*
7	*Taenia*	*Taenia asiatica* *Taenia crassiceps* *Taenia hydatigena* *Taenia krepkogorski* *Taenia laticollis* *Taenia madoquae* *Taenia martis* *Taenia multiceps* *Taenia mustelae* *Taenia ovis* *Taenia parva* *Taenia pisiformis* *Taenia saginata* *Taenia serialis* *Taenia solium* *Taenia twitchelli* *Taeniopygia guttata*

Table 1 List of different worm genera



www.ncbi.nlm.nih.gov/genomes/OrganelleResource.cgi?taxid=2759&sort=Genome

sol 142 of 144

Taenia crassiceps	NC_002547					
Taenia hydatigena	NC_012896	13492 nt	12	24	07/01/2009	08/23/2010
Taenia krepkogorski mitochondrial DNA	NC_021142	13792 nt	12	24	04/23/2013	04/23/2013
Taenia laticollis mitochondrial DNA	NC_021140	13483 nt	12	24	04/23/2013	04/23/2013
Taenia madoquae mitochondrial DNA	NC_021139	13689 nt	12	24	04/23/2013	04/23/2013
Taenia martis mitochondrial DNA / marten tapeworm	NC_020153	13536 nt	12	24	01/17/2013	04/16/2013
Taenia multiceps	NC_012894	13693 nt	12	24	07/01/2009	08/23/2010
Taenia mustelae mitochondrial DNA	NC_021143	13582 nt	12	24	04/23/2013	04/23/2013
Taenia ovis mitochondrial DNA / sheep tapeworm	NC_021138	13704 nt	12	24	04/23/2013	04/23/2013
Taenia parva mitochondrial DNA	NC_021141	13482 nt	12	24	04/23/2013	04/23/2013
Taenia pisiformis	NC_013844	13387 nt	12	24	02/16/2010	08/23/2010
Taenia saginata / beef tapeworm	NC_009938	13670 nt	12	24	10/18/2007	04/14/2009
Taenia serialis mitochondrial DNA	NC_021457	13688 nt	12	24	06/10/2013	06/10/2013
Taenia solium / pork tapeworm	NC_004022	13709 nt	12	24	06/11/2002	02/01/2010
Taenia taeniaeformis	NC_014768	13647 nt	12	24	12/09/2010	06/17/2011
Taenia [illegible] Opisthokonta; Species: Taenia solium	NC_021093	13519 nt	12	24	04/23/2013	04/23/2013
Taeniopygia guttata / zebra finch	NC_007897	16853 nt	13	24	03/08/2006	04/15/2009
Taeniura meyeni / blotched fantail ray	NC_019641	17638 nt	13	24	12/03/2012	05/15/2013
Takifugu chinensis / karasu	NC_011633	16447 nt	13	24	12/05/2008	05/14/2009
Takifugu chrysops / akamefugu	NC_011624	16448 nt	13	24	12/05/2008	05/14/2009
Takifugu exascurus / mushifugu	NC_011622	16443 nt	13	24	12/05/2008	05/14/2009
Takifugu fasciatus / obscure pufferfish	NC_013087	16444 nt	13	24	08/12/2009	08/14/2009
Takifugu niphobles / grass puffer	NC_011625	16442 nt	13	24	12/05/2008	05/14/2009
Takifugu oblongus / lattice blaasop	NC_011634	16444 nt	13	24	12/05/2008	05/14/2009
Takifugu obscurus / mefugu	NC_011626	16445 nt	13	24	12/05/2008	05/14/2009
Takifugu ocellatus	NC_011635	16449 nt	13	24	12/05/2008	05/14/2009
Takifugu pardalis / panther puffer	NC_011627	16463 nt	13	24	12/05/2008	05/14/2009
Takifugu poecilonotus / finepatterned puffer	NC_011621	16448 nt	13	24	12/05/2008	05/14/2009
Takifugu porphyreus / purple puffer	NC_011628	16447 nt	13	24	12/05/2008	05/14/2009
Takifugu rubripes / torafugu	NC_004299	16447 nt	13	24	09/24/2002	02/01/2010
Takifugu snyderi / shosai-fugu	NC_011630	16450 nt	13	24	12/05/2008	05/14/2009
Takifugu stictonotus / gomafugu	NC_011629	16451 nt	13	24	12/05/2008	05/14/2009
Takifugu vermicularis / pear puffer	NC_011631	16443 nt	13	24	12/05/2008	05/14/2009
Takifugu xanthopterus / yellowfin puffer	NC_011632	16442 nt	13	24	12/05/2008	05/14/2009
Takydromus sexlineatus	NC_022703	18943 nt	13	24	11/03/2013	11/03/2013
	NC_008773	18245 nt	13	24	01/10/2007	02/01/2010

www.ncbi.nlm.nih.gov/Taxonomy/Browser/wwwtax.cgi?lvl=3&id=6204

Fig 13: Depicts mitochondrial genome retrieval of *taenia* genus

The fasta sequence of all the species listed above was downloaded from NCBI database. Different folders were made for different genus.

4.2 **Extraction of COI gene from the sequences**

From whole mitochondrial genome of different worm species, the COI gene sequence was extracted using the COI gene sequence (1527 bp) retrieved from NCBI using EMBL alignment tools.

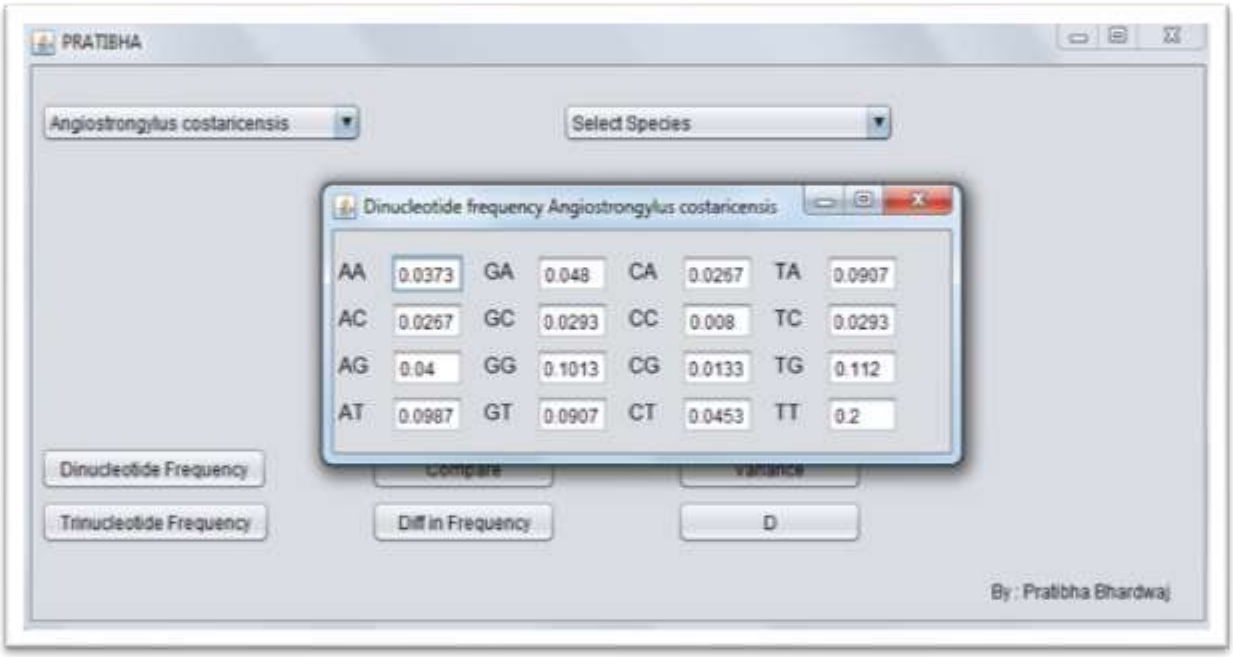
This was further reconfirmed by aligning with a partial COI sequence (372 bp) retrieved from NCBI database. The alignment files were saved. (Appendix 1)

4.3 **Discrimination of species using java program**

For discrimination, a java program was developed, to find out the various parameters required to discriminate various pathogenic species. The java code for the program is as mentioned. (Appendix 2)

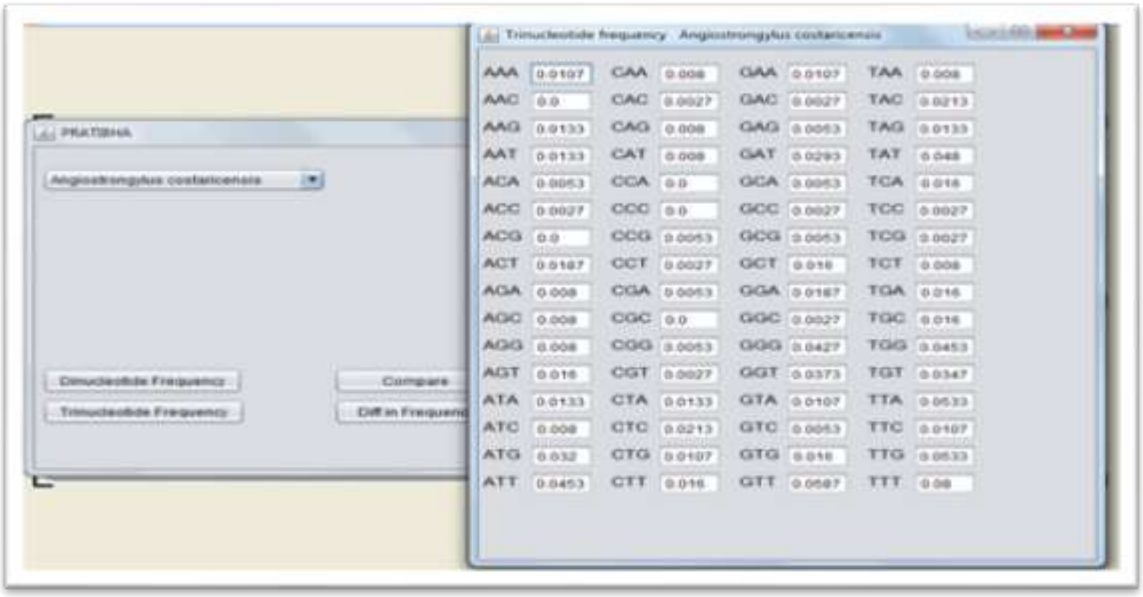
4.3.1 **Oligonucleotide frequency calculation**

Dinucleotide and trinucleotide frequencies were found out using the java application "Pratibha" and stored as two .xls files namely dinucleotide frequencies and trinucleotide frequencies. (Appendix 3)



AA	0.0373	GA	0.048	CA	0.0267	TA	0.0907
AC	0.0267	GC	0.0293	CC	0.008	TC	0.0293
AG	0.04	GG	0.1013	CG	0.0133	TG	0.112
AT	0.0987	GT	0.0907	CT	0.0453	TT	0.2

Figure14: Depicting dinucleotide frequencies of Angiostrongylus Costaricensis.

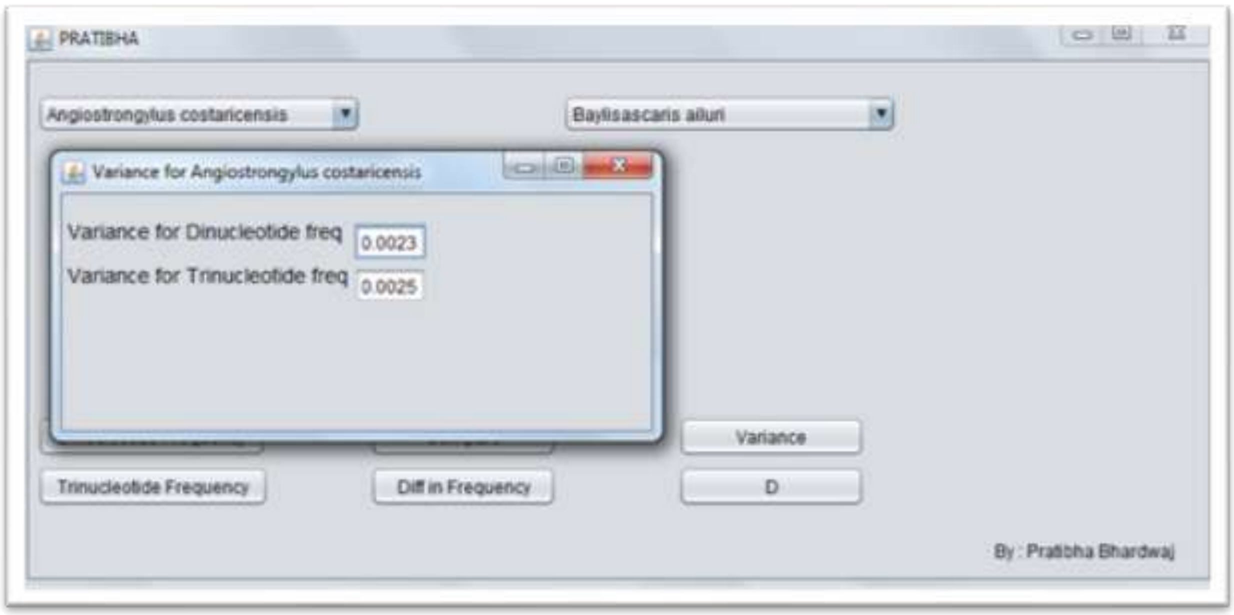

PRATIBHA
Angiostrongylus costaricensis
Dinucleotide Frequency
Trinucleotide Frequency
Compare
Diff in Frequenc
Trinucleotide frequency Angiostrongylus costaricensis

AAA	0.0107	CAA	0.008	GAA	0.0107	TAA	0.008
AAC	0.0	CAC	0.0027	GAC	0.0027	TAC	0.0213
AAG	0.0133	CAG	0.008	GAG	0.0053	TAG	0.0133
AAT	0.0133	CAT	0.008	GAT	0.0293	TAT	0.048
ACA	0.0053	CCA	0.0	GCA	0.0053	TCA	0.016
ACC	0.0027	CCC	0.0	GCC	0.0027	TCC	0.0027
ACG	0.0	CCG	0.0053	GCG	0.0053	TCG	0.0027
ACT	0.0187	CCT	0.0027	GCT	0.016	TCT	0.008
AGA	0.008	CGA	0.0053	GGA	0.0187	TGA	0.016
AGC	0.008	CGC	0.0	GGC	0.0027	TGC	0.016
AGG	0.008	CGG	0.0053	GGG	0.0427	TGG	0.0453
AGT	0.016	CGT	0.0027	GGT	0.0373	TGT	0.0347
ATA	0.0133	CTA	0.0133	GTA	0.0107	TTA	0.0533
ATC	0.008	CTC	0.0213	GTC	0.0053	TTC	0.0107
ATG	0.032	CTG	0.0107	GTG	0.016	TTG	0.0533
ATT	0.0453	CTT	0.016	GTT	0.0587	TTT	0.08

Figure15: Depicting trinucleotide frequencies of Angiostrongylus Costaricensis.

4.3.2 **<u>Calculation of variance</u>**

The variance of the dinucleotide and trinucleotide frequencies was calculated using the java application and stored in the above mentioned .xls files. (Appendix 3)


PRATIBHA

Angiostrongylus costaricensis

Baylisascaris ailuri

Variance for Angiostrongylus costaricensis

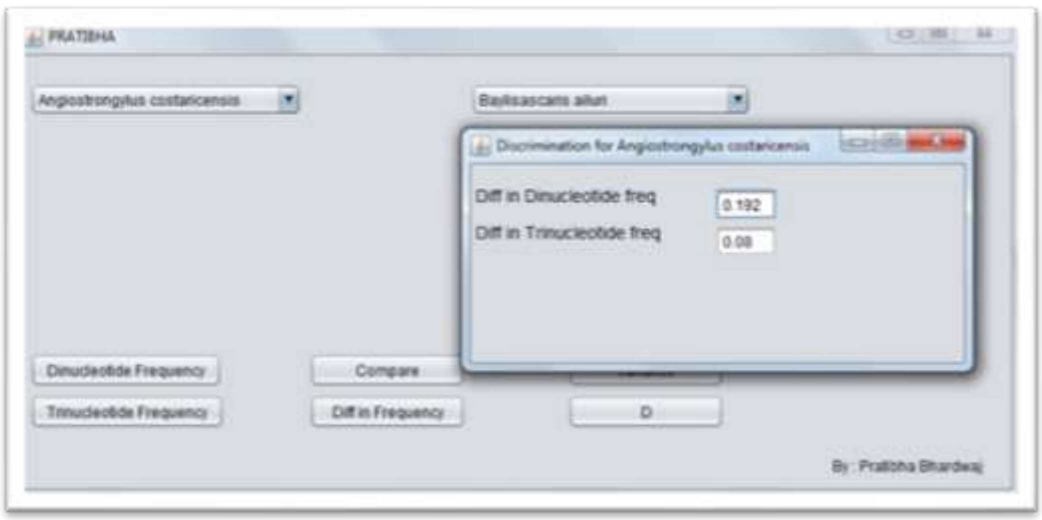
Variance for Dinucleotide freq 0.0023

Variance for Trinucleotide freq 0.0025

Figure16: Depicting variance of oligotide frequency of Angiostrongylus Costaricensis.

4.3.3 **<u>Difference in Oligonucleotide frequency</u>**

The difference in maximum and minimum dinucleotide and trinucleotide frequency was calculated using the java application and stored .xls file named as "Oligonucleotide frequency difference". (Appendix 3)


PRATIBHA

Angiostrongylus costaricensis

Baylisascaris ailuri

Discrimination for Angiostrongylus costaricensis

Diff in Dinucleotide freq 0.192

Diff in Trinucleotide freq 0.08

Figure16: Depicting the Oligonucleotide frequency difference for Angiostrongylus Costaricensis

4.3.4 **Comparison of Oligonucleotide frequencies**

A line graph was plotted between the dinucleotide and trinucleotide frequencies of two species.

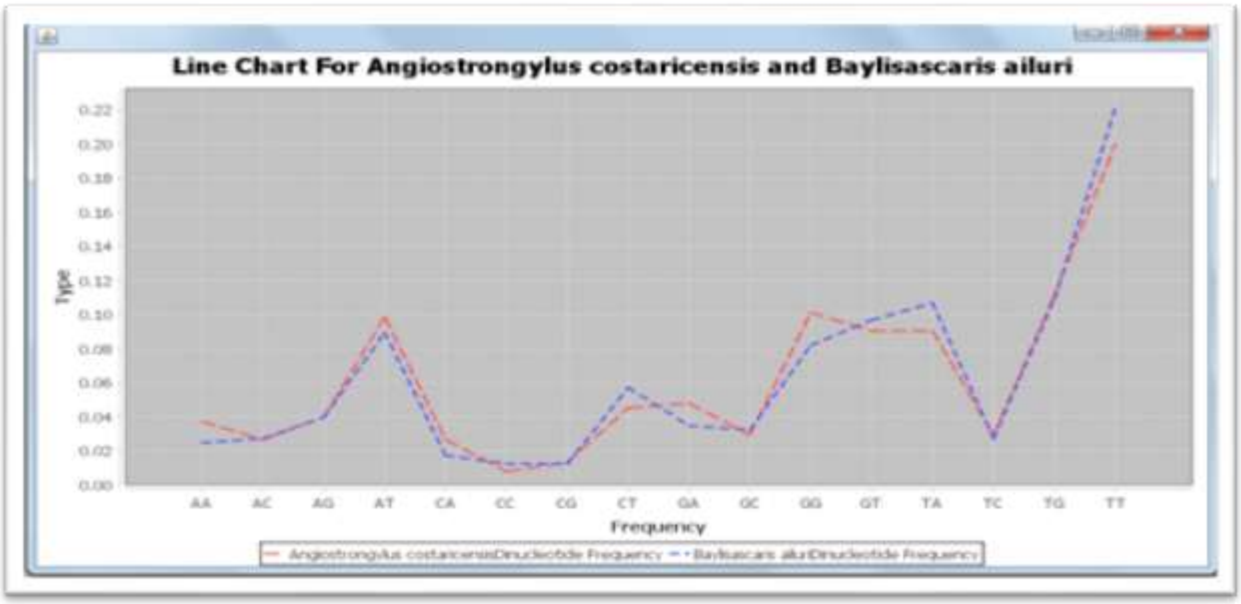


Figure17: Depicting the graph for the comparison of dinucleotide frequencies of the two species namely Angiostrongylus Costaricensis and Baylisascaris Aluri.

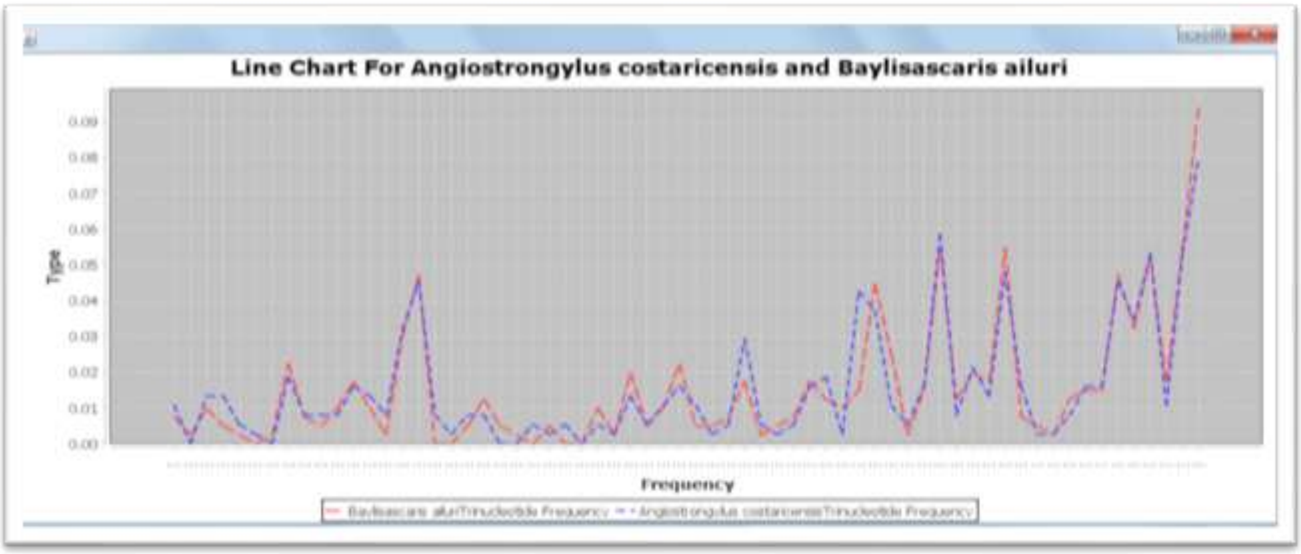
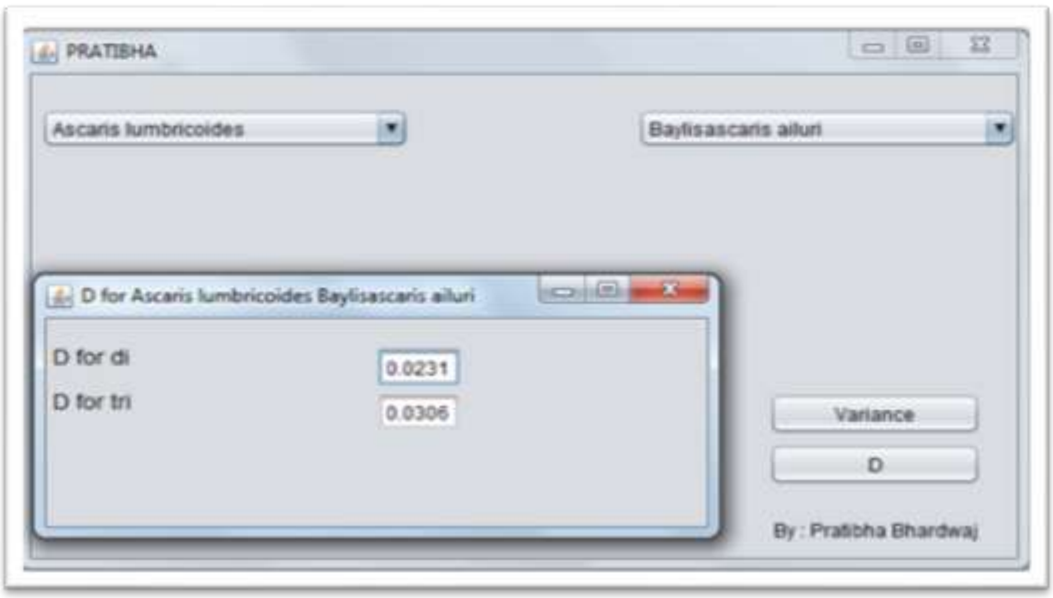


Figure18: Depicting the graph for the comparison of trinucleotide frequencies of the two species namely Angiostrongylus Costaricensis and Baylisascaris Aluri.

4.3.5 **Statistical analysis**

The Euclidean distance D was calculated using the formula

$$D = \sqrt{\sum_{i=1}^{N} |F1 - F2|^2}$$



D for di 0.0231

D for tri 0.0306

Figure19: Depicting D of Angiostrongylus Costaricensis.

5. **DISCUSSION AND FUTURE PERSPECTIVE**

The diversity in worms in the tropical region contributes to a wide range of worm borne diseases. According to a report published by NIA, diseases and epidemiological conditions caused by worms are as diverged as the types of worms which are responsible for them. The diseases caused by them varies from malaria to eye infections (Nimir *et al,*. 2012). The wide variety of worms like lungworms, roundworms, fluke worms etc. whose whole genome is available has been studied in this work.
Nowadays, we are more inclined towards gene based diagnostic techniques for better identification with precision, e.g., PCR based diagnostic methods. In a recent study, saliva and urine samples of *Plasmodium falciparum* and *Plasmodium vivax*-infected patients have been found to contain malarial DNA which can be amplified using PCR. However, only trace amounts of malarial DNA has been found in saliva and urine, the amount found is less than that is found in the concurrent blood samples of infected individuals, this has resulted into constrains in their usefulness in diagnosis. *Plasmodium* possess small mitochondrial genomes (mtDNA) of approximately 6 kb with a copy number of approximately 30 to 100 per parasite, the copy number of 18S rRNA is 4 to 8 per parasite nucleus . Therefore, the diagnostic sensitivity of a PCR assay targeting mtDNA is superior to that of an 18S rRNA-based technique. The mtDNA sequences of every malaria species are much conserved, on the other hand variation in sequences occurs between species. The cytochrome *b* gene (*Cytb*) found in mtDNA is used widely to study phylogenetics and evolutionary relationships among plasmodia. So, despite being low abundance, the sequence divergence as compared to 18S rRNA, *Cytb* has a potential role in differentiating malarial species. (Pattakorn *et al,*. 2010). Our aim was to device an *in silico* method to differentiate between the species of these worms based on their genomic variability. The recent trends in these techniques have been the use of DNA barcoding and oligo nucleotide frequency distribution (Tyagi et al., 2010). Our method is based on DNA barcoding using oligo nucleotide frequency distribution techniques. Oligonucleotide frequency range of a barcode locus can discriminate between species. Ability to discriminate species using very short DNA fragments may have wider applications in forensic and conservation studies. (Tyagi et al., 2010). A program was developed to analyse the barcode gene and give outputs in the form of dinucleotide frequencies, trinucleotide frequencies, their comparison and Euclidean distance. The variance between oligonucleotide frequencies is also calculated.
Our results show distinguishable variances for each species. We have been able to identify these species uniquely with the help of these indices. The results are comparable to reported researches in this field on other species like marine metazoan, fish, fungus etc (Bucklin *et al,*. 2010; Becker *et al,*. 2011; Begerow *et al,*. 2011).We have been able to curate all the indices to find out even a slight change in the species arising due to mutation (non-inheritable mutation) which may lead to increase or decrease in their virulence potential. Thus a treatment for such a mutation once diagnosed may be easily targeted. All the data have been included in Appendix II .The data will serve as a starting point to any researchers planning a work on virulence of the worms.

5.1 **<u>Unique Selling Proposition (USP) of our work:</u>**

The uniqueness of our work is the use of the DNA barcoding technique as an in silico diagnostic tool. Its originality lies in the fact that no research has been reported till date on bar coding of the worms especially pathogenic ones.

5.2 **<u>Conclusion:</u>**

The results of dinucleotide frequency distribution discriminates have found to discriminate between the species well (Appendix II). However to add more stringency to the process the trinucleotide frequency indices gives us an even more divergence with better resolution. (Chanda et al., 2010**)**
All *in silico* analysis are subject to a critical statistical test to prove its viability. We have used the popular method of finding out the Euclidian Distance (D) (Tyagi et al., 2010) to facilitate the process of validation of our work. The D values also correspond to our proposed method of finding the divergence.

5.3 **<u>Future milestones:</u>**

- These results can be used for confirmation of PCR diagnostic methods.

- Our tool can be expanded to accommodate most of the diagnostic technique for pathogens. The sequences of other pathogens can be added to the JAVA program and indices for them can be generated and used for disease diagnosis.

- These indices can also be used to trace drug resistivity of the pathogens. The resistivity arising even from point mutations like SNPs can be traced with these indices (Jeanette *et al*,. 2011).

REFERENCES

.

1. Amal; R. Nimir; Ahmed, Saliem; and Ibrahim, Abdel Aziz Ibrahim (2012), "Ophthalmic Parasitosis: A Review Article," Interdisciplinary Perspectives on Infectious Diseases, vol. 2012, Article ID 587402, 12 pages, 2012. doi:10.1155/2012/587402

2. Becker, S; Hanner, R; Steinke, D (2011). Five years of FISH-BOL: brief status report. Mitochondrial DNA. 2011 Oct;22 Suppl 1:3-9. doi:10.3109/19401736.2010.535528.Epub 2011 Jan 27. Review. PubMed PMID: 21271850.

3. Begerow, D; Nilsson, H; Unterseher, M; Maier, W (2010). Current state and perspectives of fungal DNA barcoding and rapid identification procedures. Appl Microbiol Biotechnol. 2010 Jun; 87(1):99-108.doi: 10.1007/s00253-010-2585-4. Epub 2010 Apr 20. Review. PubMed PMID: 20405123.

4. Blaxter, ML; Nematoda (2013). Genes, genomes and the evolution of parasitism. Adv Parasitol. 2003; 54:101-95. Review. PubMed PMID: 14711085.

5. Bucklin, A; Steinke, D; Blanco-Bercial L (2011). DNA barcoding of marine metazoa. Ann Rev Mar Sci. 2011; 3:471-508. Review. PubMed PMID: 21329214.

6. Buppan. P; Putaporntip, C; Pattanawong, U; Seethamchai, S; Jongwutiwes, S (2010). Comparative detection of Plasmodium vivax and Plasmodium falciparum DNA in saliva and urine samples from symptomatic malaria patients in a low endemic area. Malar J. 2010 Mar 9**; 9:72**. doi: 10.1186/1475-2875-9-72. PubMed PMID: 20214828; PubMed Central PMCID: PMC2843727.

7. Busse, HJ; Denner, EB; Lubitz, W (1996). Classification and identification of bacteria: current approaches to an old problem. Overview of methods used in bacterial systematic. J Biotechnol. 1996 Jun 6; **47(1):**3-38. Review. PubMed PMID: 8782421.

8. CBOL Plant Working Group (2009) A DNA barcode for land plants. *Proceedings of the National Academy of Sciences of the United States of America* ***106, 12794–12797.***

9. Eyualem, A; Blaxter, M (2003). Comparison of biological, molecular, and morphological methods of species identification in a set of cultured panagrolaimus isolates. J Nematol. 2003 Mar; **35(1)**:119-28. PubMed PMID: 19265985; PubMed Central PMCID: PMC2620612

10. Folmer, O; Black, M; Hoeh, W; Lutz, R; Vrijenhoek, R(1994). DNA primers for amplification of mitochondrial cytochrome c oxidase subunit I from diverse metazoan invertebrates. Mol Mar Biol Biotechnol. 1994 Oct;**3(5)**:294-9. PubMed PMID: 7881515.

11. Geller, J; Meyer, C; Parker, M; Hawk, H (2013). Redesign of PCR primers for mitochondrial cytochrome c oxidase subunit I for marine invertebrates and application in all-taxa biotic surveys. Mol Ecol Resour. 2013 Sep; **13(5)**:851-61. doi: 10.1111/1755-0998.12138. Epub 2013 Jul 13. PubMed PMID: 23848937.

12. Greenstone, MH; Vandenberg, NJ(2011). Barcode haplotype variation in north American agro ecosystem lady beetles (Coleoptera: Coccinellidae). Mol Ecol Resour.2011 Jul;**11(4**):629-37. doi: 10.1111/j.1755-0998.2011.03007.x. Epub 2011 Apr 3. PubMed PMID: 21457477.

13. Hammond, P. (1992). Species inventory. In *Global biodiversity: status of the earth's living resources (ed. B. Groombridge), pp.* 17–39. London: Chapman & Hall.

14. Hawksworth, D. L. & Kalin-Arroyo, M. T. 1995 Magnitude and distribution of biodiversity. In *Global biodiversity assessment* (ed. V. H. Heywood), pp. 107–191. Cambridge University Press.

15. Jarman, SN; Nicol, S; Elliott, NG; McMinn, A(2000). 28S rDNA evolution in the Eumalacostraca and the phylogenetic position of krill. Mol Phylogenet Evol. 2000 Oct;**17(1):**26-36. PubMed PMID: 11020302.

16. Jarman, S. N. & Elliott, N. G. (2000) DNA evidence for morphological and cryptic Cenozoic speciation in the Anaspididae, 'living fossils' from the Triassic. *J. Evol. Biol.* ***13,*** 624–633.

17. Knowlton, N. 1993 Sibling species in the sea. *A. Rev. Ecol. Syst.* ***24, 189–216.***

18. Kulsantiwong, J; Prasopdee, S; Ruangsittichai, J; Ruangjirachuporn, W; Boonmars, T; Viyanant, V; Pierossi, P; Hebert, PD; Tesana, S(2013). DNA barcode identification of freshwater snails in the family bithyniidae from Thailand. PLoS One. 2013 Nov 4;**8(11**):e79144. doi: 10.1371/journal.pone.0079144. PubMed PMID: 24223896; PubMed Central PMCID: PMC3817070

19. Lemay, JF; Herbert, AR; Dewey, DM; Innes, AM(2003). A rational approach to the child with mental retardation for the pediatrician. Pediatric Child Health. 2003 Jul;**8(6):**345-56. PubMed PMID: 20052328; PubMed Central PMCID: PMC2795455.

20. Nasrallah R, Laneuville O, Ferguson S, Hébert RL. Effect of COX-2 inhibitor NS-398 on expression of PGE2 receptor subtypes in M-1 mouse CCD cells. Am J Physiol Renal Physiol. 2001 Jul;**281(1):**F123-32. PubMed PMID: 11399653.

21. Premalatha, B; Ramesh, V; Babu, SP; Balamurali, PD (2014). Procedures to view aberrations--a travel from protein to gene: literature review. Indian J Dent Res.2014 Jan-Feb; **25(1)**:91-4. doi: 10.4103/0970-9290.131145. PubMed PMID: 24748307.

22. Shneer, VS(2009). [DNA barcoding of animal and plant species as an approach for their molecular identification and describing of diversity]. Zh Obshch Biol. 2009 Jul-Aug;**70(4)**:296-315 Review. Russian. PubMed PMID: 19799325.

23. Simmons, RB; Weller, SJ(2001). Utility and evolution of cytochrome b in insects. Mol Phylogenet Evol. 2001 Aug; 20(2):196-210. PubMed PMID: 11476629.

24. Sri-Aroon, P; Butraporn, P; Limsomboon, J; Kerdpuech, Y; Kaewpoolsri, M et al. (2005) Freshwater mollusks of medical importance in Kalasin Province, Northeast Thailand. Southeast Asian J Trop Med Public Health **36**: 653-657. PubMed: 16124433.

25. Vernooy, R; Haribabu, E; Muller, MR; Vogel, JH; Hebert, PD; Schindel, DE; Shimura, J; Singer, GA(2010). Barcoding life to conserve biological diversity: beyond the taxonomic imperative. PLoS Biol. 2010 Jul 13; **8(7)**:e1000417. doi: 10.1371/journal.pbio.1000417. PubMed PMID: 20644709; PubMed Central PMCID: PMC2903590.

26. Wares, JP; Gaines, SD; Cunningham, CW (2001). A comparative study of asymmetric migration events across a marine biogeographic boundary. Evolution. 2001 Feb; **55(2)**:295-306. PubMed PMID: 11308087.

27. Zhang, DX; Hewitt, GM (1997). Assessment of the universality and utility of a set of conserved mitochondrial COI primers in insects. Insect Mol Biol. 1997 May;**6**(2):143-50. PubMed PMID: 9099578.

APPENDIX

Appendix 1: Alignment files

```
NC_013065.1     7751 TTGATTGGAAGAAAAATTCTCTTACATTT-ATATATATAAGAAATATCAA   7799
                                    .|||| ||||.|
EU340360.1        1 ------------------------GATTTGATATCT--------------     12

NC_013065.1     7800 GGTGGTTTGTCGGTTTGATTAGAAAGTTCTAATCATAAGGATATTGGTAC   7849
                       ||.||||||..||||            .||||||||..|.|||||..
EU340360.1       13 --TGATTTGTCTCTTTG-------------GATCATAAGCGTGTTGGTGT     47

NC_013065.1     7850 TCTTTATTTTTTGTTTGGTTTATGGTCGGGT--ATGT-TAGGTACTGCTT   7896
                     |.|||||.||.|..||||   |..||.||||  ||.| |||||...|.||
EU340360.1       48 TGTTTATATTATTCTTGG---AGTGTGGGGTGGATTTATAGGTTTAGGTT     94

NC_013065.1     7897 TATCTTTGATTG-TTCGTTTGGAATTGTCTAAGCCTGGAATGTTGTTGTC   7945
                     ||..|||| ||| |.|||||..|.||.|.|.|.|||    |.||.|...|
EU340360.1       95 TAAGTTTG-TTGATACGTTTAAATTTTTGTGATCCT----TATTATAAAC    139

NC_013065.1     7946 TAAT---------GGGCAATTGTATAATTCAATT--ATTACGGCTCATGC   7984
                     |.||         ||     |.||||||  |||  ||.||...|||.|.
EU340360.1      140 TTATTCCTTGTGAGG-----TATATAATT--ATTTGATAACTAATCACGG    182

NC_013065.1     7985 TTTTTTGATGATTTTTTTTATGGTGATGCCTAGTATGATTGGTGGTTTTG   8034
                     |.|....||||||||||.|..|.||||||..|.|.||.||.|||||||
EU340360.1      183 TATAGCAATGATTTTTTTTTTTTTAATGCCTGTTTTAATAGGGGGTTTTG    232

NC_013065.1     8035 GTAATTGAATAT----TGCCTTTGATGTTGGGGGCTCCGGATATGAG---   8077
                     |||    |||||    |.||.||..|.|||.|         ||||||
EU340360.1      233 GTA----AATATCTTCTTCCGTTTTTTTTGAG---------TATGAGTGA    269

NC_013065.1     8078 ------TTTTCCTCGTTTGAATAATTTGAGTTTTTG-ATTAT--TACCAA   8118
                           |||.||.|||||.|||..||||||.|||| ||.||  |.|
EU340360.1      270 TTTACCTTTGCCCCGTTTAAATTCTTTGAGTGTTTGAATGATGGTTC---    316

NC_013065.1     8119 CTTCGATGTTT----TTGATTTTAGATTCTTGTTTTGTGGATATAGGATG   8164
                     ||||.||.|||    |.||.||.||   .||||.||.|||||.|  |.||
EU340360.1      317 CTTCAATATTTTATATGGAATTAAG---TTTGTATTATGGATCT--GGTG    361

NC_013065.1     8165 TGGGACTAGTTGGACTGTTTATCCGCCTTTGAGAAGTTTAGGTCATCCTG   8214
                     |     |.|||||||..|||||||.||||||.....|||||.|..|.|||
EU340360.1      362 T-----TGGTTGGACCTTTTATCCACCTTTGTCTTCTTTAGCTACTTCTG    406

NC_013065.1     8215 GTAGAAGG-GTGGATTTGGCGATTTTTAGTTTGCATTGTGCTGGACTGAG   8263
                     || |..|| ||||||.....||.||...|||.|||..|||||    |.|
EU340360.1      407 GT-GTTGGTGTGGATTACTTAATGTTCTCTTTACATCTTGCTG----GTG    451

NC_013065.1     8264 T-TCTATTTTG---GGGGGTATTAATTTTATGTGTACAACGAAGAAT-AT   8308
                     | ||||.||||   ||...|||.||||||||   |||.|| .|.||| .|
EU340360.1      452 TATCTAGTTTGATTGGTTCTATAAATTTTAT---TACTAC-TATAATGTT    497

NC_013065.1     8309 GCGAAGAAGTTCAATTTCTTTGGAACA--TATGAGTTTGTTTGTTTGATC   8356
                     |||...|||.|||..|||||     ||  |||.||         .|||||
EU340360.1      498 GCGTCTAAGGTCATGTTCTT-----CAGTTATTAG---------ATGATC    533

NC_013065.1     8357 TGTTTTTGTGACTGTTTTTTTGTTGGTGTTGTCTTTACCTGTGTTGGCCG   8406
                     |..|||..|.|||....|.|||||..||||.||.||.||.||..|.||.|
EU340360.1      534 TTATTTATTTACTTCGGTGTTGTTATTGTTATCGTTGCCGGTTCTTGCTG    583
```

```
NC_013065.1     8407 GGGCTATCACTATATTATTAACTGATCGTAATTTAAATACTTCTTTTTTT   8456
                     ..|.|||.||||.||.||...|||||||.||...||||.|||||||
EU340360.1      584 CAGGTATAACTATGTTGTTGTTTGATCGTAAATTTGGTACTGCTTTTTTT    633

NC_013065.1     8457 GATCCTAGTTC---TGGTGGTAATCCTTTGATTTATCGGCATTTGTTTTG   8503
                     ||.|| ||  |   ||||||.|||||.||.|.|.||..|||||.|||||
EU340360.1      634 GAGCC-AG--CAGGTGGTGGTGATCCTGTGTTATTTCAACATTTATTTTG    680

NC_013065.1     8504 ATTTTTTGGTCATCCTGAGGTTTATGTTTTGATTTTGCCTGCTTTTGGGA   8553
                     .|||||||||.||.||.||.||||||||||.||||||..|||||.|
EU340360.1      681 GTTTTTTGGTCACCCAGAAGTATATGTTTTGATATTGCCTGGATTTGGTA    730

NC_013065.1     8554 TTGTTAGACAGTCTACTTTATA--------TTTAACGGGTAAAAAAGAGG   8595
                     |.||.||.||        ||||        |||||   ||||.||..|
EU340360.1      731 TAGTAAGTCA--------TATATGTATGTCTTTAA---GTAATAATAA--    767

NC_013065.1     8596 TT-----TTTGGTTATT-TGGGTATGGTTTATGCTA----TTTTAAGAAT   8635
                     ||     |||||.|||| ||||| |.||||.|||||    ||.|    ||
EU340360.1      768 TTCTTCGTTTGGATATTATGGGT-TAGTTTGTGCTATGGGTTCT----AT    812

NC_013065.1     8636 TG---GTTTGATTGGTTGTGTGGTTTGGGCTCATCATATATATACGGTTG   8682
                     ||   |||||   ||..|.||.||||||||||.|||||.|.||.|||||
EU340360.1      813 TGTGTGTTTG---GGGAGAGTTGTTTGGGCTCACCATATGTTTATGGTTG    859

NC_013065.1     8683 GTATGGATTTAGATTCTCGTGCTTATTTTACTGCGGCTACTATAGTTATT   8732
                     |||||||.||.|..||..||.||.|||||.|.|.|..||.||..||||.
EU340360.1      860 GTATGGATGTAAAGACTTCTGTTTTTTTTTAGTTCTGTAACAATGATTATA    909

NC_013065.1     8733 GCGGTTCCTACGGGAGTGAAGGTTTTTAGGTGGTTGGCAACTTTATTTGG   8782
                     |...|.||.||        ||||.||.|||||.||    .|||.|||
EU340360.1      910 GGTATACCAAC--------AGGTATTAAGGTGTTT----TCTTGATT---    944

NC_013065.1     8783 TATAAAGATAT-TGTT------------------------TCAACCTATT   8807
                            |||| ||||                        |.|.||.||.
EU340360.1      945 -------ATATATGTTAGGGAGTAGTGGGTTGCGTGCGGCTGATCCAATA    987

NC_013065.1     8808 TTATTGTGGGTTTTGGGTTTTATTTTTTTATTTACTATCGGTGG-GTTAA   8856
                     .|.|.|||..||.|.||||||||.|||||.|||||..|.||||| ||| |
EU340360.1      988 CTTTGGTGAATTGTTGGTTTTATATTTTTGTTTACAGTTGGTGGTGTT-A   1036

NC_013065.1     8857 CGGGGGTTATATTGTCTAATTCTAGTTTGGAT--ATTATTTTACATGATA   8904
                     |.|||.|..|.||.|||..||||..||||||  ||||||  ||.||||
EU340360.1     1037 CTGGGATAGTTTTATCTGCTTCTGCTTTGGATAGATTATTT--CACGATA   1084

NC_013065.1     8905 CTT-ATTATGTAGTTAG-GCATTTTCATTATGTTTTAAGGTTGGGGGCTG   8952
                     ||| ||| ||| |.||| .||.||||||||||             ||||
EU340360.1     1085 CTTGATT-TGT-GATAGCTCACTTTCATTATGT-------------GCTG   1119

NC_013065.1     8953 TTTTTG---------------GTATTTTTACTGGTATTAGTCTT--TGAT   8985
                     |.||||               |||.||.||.||   ||.|| ||  ||||
EU340360.1     1120 TCTTTGGGTTCTTATAGGAGTGTAATTATAATG---TTTGT-TTGGTGAT   1165

NC_013065.1     8986 GAGGGTTTATGACTGGTTG-TGTTTATAATAAGTTGTATATAGT-----G   9029
                     ||...|||||.|.|||||| |.||||.||.|    |||| |.||     |
EU340360.1     1166 GACCTTTTATAATTGGTTGTTCTTTAAAAAA----GTAT-TTGTTGCAAG   1210

NC_013065.1     9030 GTGATGTTTTTTTTG---ATGTTTTTGGGTGTGAATTTGACTTTTTTTCC   9076
                     ||.|| |..||.|||   |||.||   |||.|.||||||..||||||||
EU340360.1     1211 GTCAT-TGATTATTGTCTATGGTT---GGTTTTAATTTGTGTTTTTTTCC   1256
```

```
NC_013065.1     9077 TTTGCATTTTGCTGGTTTACATGGTTATCC---TCGTAAGTATTTAG---   9120
                     ..|||||.|...|||.|.||||.|..||   ||||     .||||
EU340360.1      1257 GATGCATTATTTAGGTGTTCATGGGTTGCCACGTCGT-----GTTAGATG   1301

NC_013065.1     9121 --ATTATCCTGATGTTTATTCAGTTTGAAATGTTATTTCTTCTTTTGGGT   9168
                       ||.||||||..||||||..|||   ||.||         ||||.||.
EU340360.1      1302 TTATGATCCTGAGTTTTATTTGGTT---AAAGT---------TTTTAGGA   1339

NC_013065.1     9169 CTTTAATTAGTGTTTTTGCTTTATTTATATTTATTTTTTTGTT-GTTGGA   9217
                     .|.||..|.||||.||..||.|...||             ||| |.|||
EU340360.1      1340 GTATAGGTGGTGTGTTATCTGTGACTA-------------GTTCGATGG-   1375

NC_013065.1     9218 GTCTTTTTTTAGATATCGTTTGTTGTTACAGGATAATTATTATAATAGAA   9267
                          ||||||         ||||.||        |||||.|.|||.|
EU340360.1      1376 -----TTTTTA---------TGTTTTT--------ATTATGAGAATCG--   1401

NC_013065.1     9268 GTCCTGAGTATAGATATAGAAGTTATGTTTTTGGTCATAGTTATCAGTCA   9317
                        .|||||.|..|||.|     |.|||.||.|||     ||||
EU340360.1      1402 ---TTGAGTGTTTATAAA-----TGTGTGTTAGGT-----TTAT------   1432

NC_013065.1     9318 GAAGTTTATTTTAGAAGGAGGAGTTTAAAGCGTTAGAACTTTTAGTATAA   9367
                     ||.|||..|.|     ||..|.||.|..|..|||..|||||.|..|| .|
EU340360.1      1433 GAGGTTCTTGT-----GGGTGTGTATGTAATGTTCTAACTTGTCCTA-CA   1476

NC_013065.1     9368 TGTTATGTATTTTTAATTGCAAATTA-AAAGGTAAAGGGTTTTGTGATAA   9416
                     |.|||| .||     ||||   |||| |.||.|.|.||          |||
EU340360.1      1477 TCTTAT-CAT-----ATTG---ATTATATAGCTGACGG---------TAA   1508

NC_013065.1     9417 GATAGGAT-AAGATAGTCTGAGAGGTTCATATCCTTTAGGTGTTTTCTCT   9465
                     ||...||| ||   |||.|.||
EU340360.1      1509 GAATTGATCAA---AGTATTAG---------------------------   1527
```

Reconfirmation using 372 bp

```
NC_013065.1     8501 TTGATTTTTTGGTCATCCTGAGGTTTATGTTTTGATTTTGCCTGCTTTTG   8550
                                                           |.|||||
SMU82262           1 -------------------------------------------GGTTTTG     7

NC_013065.1     8551 GGATTGTTAGACAGTCTACTTTATA--------TTTAACGGGTAAAAAAG   8592
                     |.||.||.||.||        ||||        |||||   ||||||..
SMU82262           8 GAATAGTGAGTCA--------TATATGTATGACTTTAA---GTAAAAACA     46

NC_013065.1     8593 AGGTT-----TTTGGTTATTTGGGTATGGTTTA-----TGCTA----TTT   8628
                     |  ||     ||||||||     ||||||||     |||||    ||.
SMU82262          47 A--TTCTTTATTTGGTTAT-----TATGGTTTAGTGTGTGCTATGGGTTC     89

NC_013065.1     8629 TAAGAATTGGTTTGATT-GGTTGTGTGGTTTGGGCTCATCATATATATAC   8677
                     ||     |.||.||.|| |||.|.|||||.||.||||||||||.|.||.
SMU82262          90 TA-----TAGTATGTTTAGGTAGAGTGGTATGAGCTCATCATATGTTTAT    134

NC_013065.1     8678 GGTTGGTATGGATTTAGATTCTCGTGC---TTATTTTACTGCGGCTACTA   8724
                     |||||||||||||.||.|..|   |||   ||.|||||.|.|.|.|||||
SMU82262         135 GGTTGGTATGGATGTAAAGAC---TGCAATTTTTTTTAGTTCTGTTACTA    181

NC_013065.1     8725 TAGTTATTGCGGTTCCTACGGGAGTGAAGGTTTTTAGGTGGTTGGCAACT   8774
                     |..||||.|...|.|||||.||..|.||||||||....||||       |
SMU82262         182 TGATTATAGGTATACCTACCGGTATTAAGGTTTTCCCTTGGT-------T    224
```

```
NC_013065.1     8775 TTATTTGGTATAAAGATATTGT-------------TTCAACCTATTTTAT   8811
                     .|||.||   |.|||...||||            ||.|.|||.||.| |
SMU82262        225 ATATATG---TTAAGTGGTTGTGGGTTGCGTGTTGTTGATCCTGTTGT-T    270

NC_013065.1     8812 TG-TGGGTTTTGGGTTTTATTTTTTTATTTACTATCGGTGGGTTAACGGG   8860
                     || ||..||.|.|||||||.|||||.|||||..|.||||||.|.||.||
SMU82262        271 TGATGAATTGTTGGTTTTATATTTTTGTTTACAGTTGGTGGGGTTACTGG    320

NC_013065.1     8861 GGTTATATTGTCTAATTCTAGTTTGGATATTATTTTACATGATACTTATT   8910
                                     |.||||||   ||    |.|.|.||||.||
SMU82262        321 ----------------TATAGTTT---TA----TCTGCTTGATCCT----    343

NC_013065.1     8911 ATGTAGTTAGGCATTTTCATTAT-----GTTTTAAGGTTGGGGGCTGTTT   8955
                        |||.||||...||||||.||     ||||
SMU82262        344 ---TAGATAGGTTGTTTCATGATACTTGGTTT------------------    372
```

Appendix 2

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

package pratibha;

//import java.awt.Dimension;
import java.awt.BasicStroke;
import java.awt.Color;
import java.awt.Dimension;
import static java.lang.StrictMath.sqrt;
import java.text.DecimalFormat;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JTextField;
import org.jfree.chart.ChartFactory;
import org.jfree.chart.ChartPanel;
import org.jfree.chart.JFreeChart;
import org.jfree.chart.axis.NumberAxis;
import org.jfree.chart.plot.CategoryPlot;
import org.jfree.chart.plot.PlotOrientation;
import org.jfree.chart.renderer.category.LineAndShapeRenderer;
import org.jfree.data.category.CategoryDataset;
import org.jfree.data.category.DefaultCategoryDataset;


public class aJFrame extends javax.swing.JFrame {

    int a=10;
    String s="";
    String q="";
    String t="";
    /**
     * Creates new form aJFrame
     */
    public aJFrame() {
       initComponents();
    }
    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

       jFrame1 = new javax.swing.JFrame();
       jFrame2 = new javax.swing.JFrame();
       jColorChooser1 = new javax.swing.JColorChooser();
       jButton6 = new javax.swing.JButton();
       jComboBox1 = new javax.swing.JComboBox();
       jButton1 = new javax.swing.JButton();
       jButton2 = new javax.swing.JButton();
       jComboBox2 = new javax.swing.JComboBox();
       jButton4 = new javax.swing.JButton();
       jLabel2 = new javax.swing.JLabel();
       jButton3 = new javax.swing.JButton();
       jButton5 = new javax.swing.JButton();
       jButton7 = new javax.swing.JButton();

       javax.swing.GroupLayout jFrame1Layout = new javax.swing.GroupLayout(jFrame1.getContentPane());
       jFrame1.getContentPane().setLayout(jFrame1Layout);
       jFrame1Layout.setHorizontalGroup(
          jFrame1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
          .addGap(0, 400, Short.MAX_VALUE)
```

```
        );
        jFrame1Layout.setVerticalGroup(
            jFrame1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGap(0, 300, Short.MAX_VALUE)
        );

        javax.swing.GroupLayout jFrame2Layout = new javax.swing.GroupLayout(jFrame2.getContentPane());
        jFrame2.getContentPane().setLayout(jFrame2Layout);
        jFrame2Layout.setHorizontalGroup(
            jFrame2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGap(0, 400, Short.MAX_VALUE)
        );
        jFrame2Layout.setVerticalGroup(
            jFrame2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGap(0, 300, Short.MAX_VALUE)
        );

        jButton6.setText("jButton6");

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setBackground(new java.awt.Color(102, 0, 255));
        setBounds(new java.awt.Rectangle(0, 0, 0, 0));

        jComboBox1.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "Select Species", "Angiostrongylus cantonensis",
"Angiostrongylus costaricensis", "Ascaris lumbricoides", "Ascaris suum", "Baylisascaris ailuri", "Baylisascaris procyonis", "Baylisascaris
schroederi", "Baylisascaris transfuga", "Toxocara canis", "Toxocara cati", "Toxocara malaysiensis", "Aelurostrongylus abstrusus",
"Dictyocaulus eckerti", "Dictyocaulus viviparus", "Diplogonoporus balaenopterae", "Diplogonoporus grandis", "Taenia asiatica", "Taenia
crassiceps", "Taenia hydatigena", "Taenia krepkogorski", "Taenia laticollis", "Taenia madoquae", "Taenia martis", "Taenia multiceps",
"Taenia mustelae", "Taenia ovis", "Taenia parva", "Taenia pisiformis", "Taenia saginata", "Taenia serialis", "Taenia solium", "Taenia
twitchelli" }));
        jComboBox1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jComboBox1ActionPerformed(evt);
            }
        });

        jButton1.setText("Dinucleotide Frequency");
        jButton1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton1ActionPerformed(evt);
            }
        });

        jButton2.setText("Trinucleotide Frequency");
        jButton2.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton2ActionPerformed(evt);
            }
        });

        jComboBox2.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "Select Species", "Angiostrongylus cantonensis", "Angiostrongylus costaricensis", "Ascaris lumbricoides", "Ascaris suum", "Baylisascaris ailuri", "Baylisascaris procyonis", "Baylisascaris schroederi", "Baylisascaris transfuga", "Toxocara canis", "Toxocara cati", "Toxocara malaysiensis", "Aelurostrongylus abstrusus", "Dictyocauluseckerti", "Dictyocaulus viviparus", "Diplogonoporus balaenopterae", "Diplogonoporus grandis", "Taenia asiatica", "Taenia crassiceps", "Taenia hydatigena", "Taenia krepkogorski", "Taenia laticollis", "Taenia madoquae", "Taenia martis", "Taenia multiceps", "Taenia mustelae", "Taenia ovis", "Taenia parva", "Taenia pisiformis", "Taenia saginata", "Taenia serialis", "Taenia solium", "Taenia twitchelli" }));
        jComboBox2.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jComboBox2ActionPerformed(evt);
            }
        });

        jButton4.setText("Compare");
        jButton4.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton4ActionPerformed(evt);
            }
        });

        jLabel2.setText("By : Pratibha Bhardwaj");

        jButton3.setText("Diff in Frequency");
        jButton3.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton3ActionPerformed(evt);
            }
        });

        jButton5.setText("Variance");
        jButton5.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton5ActionPerformed(evt);
            }
        });

        jButton7.setText("D");
        jButton7.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton7ActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
                        .addGap(0, 0, Short.MAX_VALUE)
                        .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE, 128,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(19, 19, 19))
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jButton2)
                        .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                    .addGroup(layout.createSequentialGroup()
                        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(jButton1)
                            .addComponent(jComboBox1, javax.swing.GroupLayout.PREFERRED_SIZE, 222,
javax.swing.GroupLayout.PREFERRED_SIZE))

                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

                        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

                            .addComponent(jComboBox2, javax.swing.GroupLayout.PREFERRED_SIZE, 229,
javax.swing.GroupLayout.PREFERRED_SIZE)

                            .addGroup(layout.createSequentialGroup()

                                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

                                    .addComponent(jButton3, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)

                                    .addComponent(jButton4, javax.swing.GroupLayout.DEFAULT_SIZE, 128, Short.MAX_VALUE))

                                .addGap(83, 83, 83)

                                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

                                    .addComponent(jButton7, javax.swing.GroupLayout.PREFERRED_SIZE, 128,
javax.swing.GroupLayout.PREFERRED_SIZE)

                                    .addComponent(jButton5, javax.swing.GroupLayout.PREFERRED_SIZE, 128,
javax.swing.GroupLayout.PREFERRED_SIZE))

                                .addGap(22, 22, 22))))))

        );

        layout.setVerticalGroup(

            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

            .addGroup(layout.createSequentialGroup()

                .addGap(23, 23, 23)

                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

                    .addComponent(jComboBox1, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

                    .addComponent(jComboBox2, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 164, Short.MAX_VALUE)

                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

                    .addComponent(jButton4)

                    .addComponent(jButton1)

                    .addComponent(jButton5))

                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

                    .addComponent(jButton2)

                    .addComponent(jButton3)

                    .addComponent(jButton7))
                .addGap(18, 18, 18)

                .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE, 25, javax.swing.GroupLayout.PREFERRED_SIZE)

                .addContainerGap())

        );

        pack();

      }// </editor-fold>

      private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {

      ammy asIt = new ammy(s,t);

      CategoryDataset dataset =  asIt.GetData1(s,t);

      CategoryDataset dataset1 = asIt.GetData2(s,t);

      asIt.drawchart1(dataset);

      asIt.drawchart1(dataset1);

      }

      private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

                // TODO add your handling code here:

         if("Angiostrongylus cantonensis".equals(s))

        {

           q=
"GCTTTTGGGATTGTTAGACAGTCTACTTTATATTTAACGGGTAAAAAAGAGGTTTTTGGTTATTTGGGTATGGTTTATGCTAT
TTTAAGAATTGGTTTGATTGGTTGTGTGGTTTGGGCTCATCATATATATACGGTTGGTATGGATTTAGATTCTCGTGCTTATTT
TACTGCGGCTACTATAGTTATTGCGGTTCCTACGGGAGTGAAGGTTTTTAGGTGGTTGGCAACTTTATTTGGTATAAAGATAT
TGTTTCAACCTATTTTATTGTGGGTTTTGGGTTTTATTTTTTTATTTACTATCGGTGGGTTAACGGGGGTTATATTGTCTAATTC
TAGTTTGGATATTATTTTACATGATACTTATTATGTAGTTAGGCATTTTCATTATGTTT";

        }

        else if("Angiostrongylus costaricensis".equals(s))

        {

            q=
"GCTTTTGGGATTATTAGTCAATCTGCTTTGTATTTGTCAGGGAAGAAAGAGGTTTTTGGTTATTTAGGGATGGTTTATGCGAT
TTTAAGAATTGGGTTGATTGGGTGTGTAGTTTGAGCTCATCATATGTATACTGTTGGTATGGATTTGGATTCTCGTGCTTACTT
TACTGCAGCTACAATAGTTATTGCGGTTCCTACTGGGGTTAAAGTGTTTAGTTGGTTGGCTACACTTTATGGGATGAAAATGA
TGTTTCAGCCGATTTTGTTGTGGGTTATGGGGTTTATTTTTTTGTTTACTATTGGGGGTTTGACCGGGGTTATGTTATCTAATTC
AAGTTTGGATATTATTTTGCATGATACTTATTATGTGGTT";

        }

     else if("Angiostrongylus vasorum".equals(s))

        {

            q=
"GCTTTTGGGATTGTTAGTCAGTCGACTTTATATTTGACTGGGAAGAAGGAGGTGTTTGGTTATTTGGGGATGGTTTATGCGAT
TTTAAGGATTGGTTTGATTGGTTGTGTGGTGTGAGCTCATCATATGTATACTGTTGGTATAGATTTAGATTCTCGTGCTTATTT
TACTGCGGCTACTATGGTGATTGCGGTGCCGACTGGAGTGAAGGTTTTTAGTTGGTTGGCTACTTTGTTTGGGATGAAGTATA
GTATTTCAGCCTATTTTGTTGTGGGTTATAGGATTTATTTTTTTTTATTTACTATTGGGGGTTTGACGGGTGTGATATTGTCAAA
TTCGAGATTGGATATTATTTTACATGATACGTATTATGTGGTAAGTCATTTTCATTATGTGAGGTTGTTTCATGATACTTGGTT
T";

        }
```

```
        else if("Diplogonoporus balaenopterae".equals(s))

        {

            q=
"GGGTTTGGAATGATTAGACATGTTTGTAGTAACTTAGGTTGTTCATATGATACTTTTGGATTTTATGGTTTATTATTTGCTATG
TTTTCTATCGTTTGTCTTGGTAGGGTTGTATGGGGTCATCATATGTTTACGGTAGGTTTAGATGTGAAGACAGCTGTTTTTTTT
AGGTCAGTTACTATGATTATAGGGGTGCCTACTGGAATAAAGGTGTTTTCTTGGCTGTATATGATTTTAAAAAGTCGTGTCTC
TTTGCGTGAACCTATATTTTGGTGGGTGTTATCTTTTATCGTGCTGTTTACAATAGGGGGTGTTACTGGTATTATACTTTCTGC
TTGTGTTCTTGATAATATTTTGCATGATACTTGATTT";

        }

        else if("Diplogonoporus grandis".equals(s))

        {

            q=
"GGGTTTGGAATGATTAGACATGTTTGTAGTAACTTAGGTTGTTCATATGATACTTTTGGATTTTATGGTTTATTATTTGCTATG
TTTTCTATCGTTTGTCTTGGTAGGGTTGTATGGGGTCATCATATGTTTACGGTAGGTTTAGATGTGAAGACAGCTGTTTTTTTT
AGGTCAGTTACTATGATTATAGGGGTGCCTACTGGAATAAAGGTGTTTTCTTGGCTGTATATGATTTTAAAAAGTCGTGTCTC
TTTGCGTGAACCTATATTTTGGTGGGTGTTATCTTTTATCGTGCTGTTTACAATAGGGGGTGTTACTGGTATTATACTTTCTGC
TTGTGTTCTTGATAATATTTTGCATGATACTTGATTT";

        }

        else if("Aelurostrongylus abstrusus".equals(s))

        {

            q=
"GCTTTTGGTATTGTTAGTCAGTCTACTTTGTATTTGACGGGGAAGAAGGAAGTTTTTGGTTATTTAGGGATAGTTTATGCTAT
TATAAGTATTGGTTTAATTGGTTGTGTTGTTTGGGCTCATCATATATATACTGTTGGTATAGATTTGGATTCTCGTGCTTATTTT
ACGGCGGCTACGATGGTTATTGCTGTGCCAACGGGTGTTAAGGTTTTTAGTTGGTTGGCTACTTTATTTGGAATGAAGATAGT
GTTTCAGCCGGTTTTGTTGTGGGTTTTGGGTTTTATTTTTTTGTTTACTATTGGGGGGTTAACTGGGGTCATGCTTTCGAATTCT
AGTTTGGATATTATTTTGCATGATACTTATTATGTGGTTAGTCATTTTCATTATGTGTTGAGTTT";

        }

        else if("Dictyocaulus eckerti".equals(s))

        {

            q=
"GCTTTTGGTATTATTAGTCAGTCAACTTTGTATTTAACTGGTAAAAAGGAGGTTTTTGGTTCTTTGGGTATGGTTTATGCTATT
TTAAGTATTGGTTTAATTGGTTGTGTAGTATGAGCACATCATATATATACTGTTGGAATAGATTTGGATTCGCGTGCTTATTTT
ACTGCTGCTACTATGGTAATTGCTGTTCCTACGGGTGTAAAAGTTTTTAGTTGGTTGGCTACTTTGTATGGTTTAAAAATAGTA
TATAATCCTTTGTTGTTATGGGTTTTGGGTTTTATTTTTTTATTTACTATTGGGGGGTTAACTGGAGTTATTTTGTCAAATTCTA
GTTTAGATATTTTGTTACATGATACTTATTATGTTGTAAGGCATTT";

        }

        else if("Dictyocaulus viviparus".equals(s))

        {

            q=
"GCTTTTGGTATTATTAGACAATCTACTTTGTATTTAACTGGTAAAAAAGAGGTTTTTGGTTCTTTGGGTATGGTTTATGCTATT
TTAAGTATTGGTTTAATTGGTTGTGTTGTGTGGGCACATCATATGTATACTGTTGGGATGGATTTGGATTCGCGTGCTTATTTT
ACTGCTGCTACTATAGTAATTGCTGTTCCTACTGGAGTTAAGGTTTTTAGATGATTGGCTACTTTATATGGATTGAAAATGGTT
TATAATCCTTTGTTGTTGTGAGTTTTAGGTTTTATTTTTTTGTTTACTATTGGTGGTTTAACTGGTGTTATTTTGTCAAATTCTA
GTCTTGATATTTTGTTGCATGATACTTATTAT";

        }

        else if("Ascaris lumbricoides".equals(s))

        {
            q=
"GCTTTTGGTATTATTAGTCAGAGTAGTTTGTATTTGACTGGTAAAAAGGAGGTTTTTGGGTCTTTGGGTATGGTTTATGCTAT
TTTAAGTATTGGTTTGATTGGTTGTGTTGTTTGAGCTCATCATATGTATACTGTTGGTATGGATCTTGATTCTCGGGCTTATTTT
ACTGCTGCAACTATGGTTATTGCTGTTCCTACTGGTGTTAAGGTTTTTAGTTGGTTGGCTACCTTGTTTGGTATAAAAATGGTT
TTTCAGCCTTTACTTTTATGAGTTATGGGTTTTATTTTTTTGTTTACTATTGGTGGGTTAACCGGGGTTATACTTTCTAATTCTA
GTTTGGATATTATCTTGCATGATACTTATTATGTTGTTAGTCATTTTCATTATGTTCTTAGTTT";

        }

        else if("Ascaris suum".equals(s))

        {

            q=
"GCTTTTGGTATTATTAGTCAGAGTAGTTTGTATCTGACTGGTAAAAAGGAGGTTTTTGGGTCTTTGGGTATGGTTTATGCTAT
TTTGAGTATTGGTTTAATTGGTTGTGTTGTTTGAGCTCATCATATGTATACTGTTGGTATGGATCTTGACTCTCGGGCTTATTTT
ACTGCTGCAACTATGGTTATTGCTGTTCCTACTGGTGTTAAGGTTTTTAGTTGGTTGGCTACCTTGTTTGGTATAAAGATGGTT
TTTCAACCTTTACTTTTATGAGTTATGGGTTTTATTTTTTTGTTTACTATTGGTGGGTTAACCGGGGTTATGCTTTCTAATTCTA
GTTTGGATATTATCTTGCATGATACTTA-TTATGTTGTTAGTCATTTTCATTATGTCCTTAGTTT";

        }

        else if("Baylisascaris ailuri".equals(s))

        {

            q=
"GCTTTTGGTATTATTAGCCAGAGTAGGTTGTATTTAACTGGTAAAAAGGAAGTTTTTGGTTCTTTGGGTATGGTTTATGCTAT
TTTGAGTATTGGTTTAATTGGTTGTGTTGTTTGGGCTCATCATATGTATACTGTAGGTATGGATTTGGATTCTCGTGCTTATTTT
ACTGCGGCTACTATGGTTATTGCGGTCCCTACGGGTGTTAAGGTTTTTAGTTGGTTGGCTACTTTGTTTGGTATGAAGATGGTG
TTCCAGCCTTTACTTTTGTGAGTTATGGGTTTTATTTTTTTATTTACTATTGGCGGATTGACTGGGGTGATGCTTTCTAATTCTA
GTTTGGATATTATTTTGCATGATACTTA-TTATGTTGTTAGACATTTTCATTATGTTCTTAGTTT";

        }

        else if("Baylisascaris procyonis".equals(s))

        {

            q=
"GCTTTTGGTATTATTAGCCAAAGTAGGTTGTATTTAACTGGTAAGAAGGAAGTTTTTGGTTCTTTGGGAATGGTTTATGCTAT
TTTGAGTATTGGTTTGATTGGATGTGTGGTTTGGGCTCATCATATGTATACTGTGGGTATGGATTTGGATTCTCGGGCTTATTT
TACTGCGGCTACTATGGTTATTGCGGTTCCTACGGGAGTTAAGGTTTTTAGTTGGTTGGCCACTTTATTTGGTATGAAGATAGT
GTTTCAGCCTTTGCTTTTGTGGGTTATGGGGTTTATTTTTTTGTTTACTATTGGTGGTTTGACTGGGGTTATGCTTTCTAATTCT
AGTTTGGATATTATTTTGCATGATACTTATTATGTTGTTAGGCATTTTCATTATGTTCTTAGTTT";

        }

        else if("Baylisascaris schroederi".equals(s))

        {

            q=
"GCTTTTGGTATTATTAGTCAGAGTAGGTTGTATCTGACTGGTAAGAAGGAGGTTTTTGGTTCTTTGGGTATGGTTTACGCAAT
TTTGAGTATTGGTTTGATTGGTTGTGTTGTTTGAGCTCATCATATGTATACTGTGGGTATAGATTTGGATTCTCGTGCTTATTTT
ACTGCGGCTACTATGGTTATCGCAGTTCCTACGGGTGTTAAGGTTTTTAGTTGGTTGGCTACTTTGTTTGGTATGAAGATGGTG
TTTCAGCCTTTGCTTTTGTGGGTTATAGGATTTATTTTTTTGTTTACTATTGGTGGATTGACTGGGGTGATGCTTTCTAATTCTA
GTTTGGATATTATTTTGCATGATACTTA-TTATGTTGTTAGGCATTTTCATTATGTTCTTAGTTT";

        }

        else if("Baylisascaris transfuga".equals(s))

        {

            q=
"GCTTTTGGTATTATTAGTCAGAGTAGATTGTATTTAACTGGTAAAAAGGAGGTTTTTGGTTCTTTGGGTATGGTTTATGCTAT
TTTGAGTATTGGTTTAATTGGTTGTGTTGTTTGGGCTCATCATATGTATACTGTAGGTATGGATTTGGATTCTCGTGCTTATTTT
ACTGCGGCTACTATGGTTATTGCGGTTCCTACAGGTGTTAAGGTTTTTAGTTGGTTGGCCACTTTGTTTGGTATGAAGATGGTG
```

```
TTTCAGCCTTTACTTTTGTGGGTTATGGGGTTTATTTTTTTATTTACTATTGGTGGATTGACTGGGGTGATGCTTTCTAATTCTA
GTTTGGATATTATTTTGCATGATACTTATTATGTTGTTAGGCATTTTCATTATGTTCTTAGTTT";

    }

    else if("Toxocara canis".equals(s))

    {

       q=
"GCTTTTGGTATTATTAGGCAAAGTAGTTTGTATTTGACTGGTAAAAAGGAGGTTTTTGGTTCTTTAGGCATGGTTTATGCTAT
TTTAAGTATTGGTCTGATTGGCTGTGTAGTTTGGGCTCACCATATGTATACGGTGGGCATGGATTTGGATTCTCGTGCTTATTT
TACTGCGGCAACGATGGTTATTGCTGTGCCTACGGGGGTTAAGGTTTTTAGTTGGTTAGCCACTCTTTTTGGTATGAAGATGG
TGTTTCAACCTTTGCTTTTGTGGGTGCTGGGTTTTATTTTTTTATTTACTATCGGGGGGTTGACTGGTGTTATGTTATCTAATTC
TAGGTTGGACATTATCTTGCATGATACTTATTATGTTGTTAGTCATTTTCATTATGTTT";

    }

    else if("Toxocara cati".equals(s))

    {

       q=
"GATTTTTTGGGCATCCTGAGGTTTATATTTTGATTTTACCTGCCTTTGGTATTATTAGTCAAAGTAGTTTATATTTAACTGGTA
AGAAGGAGGTTTTTGGTTCTTTGGGCATGGTCTATGCTATTTTGAGTATTGGTTTGATTGGTTGTGTGGTGTGAGCTCACCACA
TGTATACTGTTGGTATAGACTTGGATTCTCGGGCTTATTTTACTGCGGCTACTATGGTTATCGCTGTGCCTACGGGTGTTAAGG
TTTTTAGTTGGTTGGCTACTCTTTTTGGTATAAAAATGGTTTTTCAACCTTTGCTTTTGTGAGTGTTGGGTTTTATTTTTTTGTTT
ACTATTGGTGGGCTTACTGGAGTTATGCTTTCTAATTCTAGTTTGGATATTATTTTGCATGACACCTATTATGTTGTGAGGCAT
TTCCACTATGTTT";

    }else if("Toxocara malaysiensis".equals(s))

    {

       q=
"GCTTTTGGTATTATTAGTCAGAGTAGTTTGTATTTAACTGGTAAGAAGGAAGTTTTTGGTTCGTTGGGGATGGTTTATGCTAT
TTTAAGTATTGGTTTGATTGGCTGTGTGGTTTGGGCTCATCATATGTATACCGTGGGTATAGATTTGGATTCTCGGGCTTATTT
TACTGCGGCGACTATGGTTATTGCTGTGCCTACTGGTGTTAAGGTTTTTAGTTGGTTGGCTACTCTTTTTGGTATGAAAATGGT
TTTTCAGCCTTTACTTTTATGGGTGTTAGGTTTTATTTTCTTGTTTACTATTGGGGGCCTTACTGGTGTGATGCTTTCTAATTCT
AGCCTTGATATTATTTTGCATGATACCTATTATGTTGTTAGACATTTTCATTATGTTT";

    }

    else if("Taenia asiatica".equals(s))

    {

       q=
"GGTTTTGGTATGATTAGTCATATATGTTTAAGAATAAGTATGTGTCCGGATGCTTTTGGTTTTTATGGTTTGTTATTTGCTATG
TTTTCAATAGTATGTTTGGGGAGAAGTGTGTGGGGTCATGATATGTTTACGGTTGGATTAGTTGTTAAGACTACTGTGTTTTTT
AGTTCGGTTACTATGATAATAGGAGTACCAACAGGAATAAAGGTTTTTACTTGACTTTATATGCTTTTAAATTCTCGTGTAAA
TAAGAGGGATCCTATATTGTGGTGGATAGTTTCTTTTATAGTGTTGTTTACCTTTGGTGGTGTGACTGGTATTGTGTTGTCTGC
TTGTGTATTGGATAAAGTTTTGCATGATACTTGATTT";

    }

    else if("Taenia crassiceps".equals(s))

    {

       q=
"GGTTTTGGAATTATTAGACATATTTGTTTGAAAATAAGTATGAATTGTGATTCTTTTGGTTTTTATGGATTGTTATTTGCTATG
TTTTCAATAGTTTGTTTAGGTAGGAGTGTTTGGGGTCATCATATGTTTACGGTTGGTTTAGATGTTAAGACTGCTGTTTTTTTT
AGTTCTGTTACTATGATTATAGGAGTACCTACAGGTATAAAGGTGTTTACTTGATTGTATATGCTTTTAAATTCGCGTGTGAA
CAAGAGTGATCCTATATTGTGGTGAATTGTTTCTTTTATAGTTTTATTTACGTTTGGTGGTGTTACTGGAATAGTATTGTCTGC
TTGTGTATTAGATAAAGTTCTTCATGATACTTGATTT";

    }
```

```
        else if("Taenia hydatigena".equals(s))

        {

           q=
"GGATTTGGAATTATTAGTCATATATGTTTGAGAATAAGTATGAGTCCTGATGCTTTTGGGTTCTATGGATTATTATTTGCTAT
GTTTTCAATAGTCTGTTTGGGTAGAAGTGTGTGGGGTCATCATATGTTTACTGTTGGGTTAGATGTTAAGACTGCTGTTTTTTT
TAGTTCTGTGACTATGATTATAGGTGTGCCTACTGGTATAAAGGTGTTTACTTGGTTATATATGCTTTTAAACTCTCATGTGAA
TAAGAGTGATCCTGTTGTTTGATGAATTGTTTCTTTTATAGTTTTGTTTACTTTTGGTGGGGTTACTGGTATTGTGTTGTCAGCA
TGTGTATTAGATAAAGTTCTTCATGATACCTGATTT";

        }

        else if("Taenia krepkogorski".equals(s))

        {

           q=
"GGATTTGGTATAATTAGTCATATATGTTTAAGAATAAGTATGTCTTCTGATGTGTTTGGGTTTTATGGTTTATTATTTGCTATG
TTTTCTATAGTTTGTTTAGGAAGAAGAGTGTGAGGTCATCACATGTTTACTGTTGGTTTAGATGTTAAGACTGCTGTGTTTTTT
AGCTCAATTACTATGATTATTGGTGTGCCTACTGGGATTAAGGTTTTTACATGATTATATATGTTATTAAATGCTCGAGTAAA
AAAGAGTGATCCTGTGTTGTGGTGAATTGTTTCATTTATAATATTGTTTACATTTGGTGGAGTTACTGGTATAGTATTGTCTGC
TTGTGTTTTAGATAAAGTGTTACATGATACTTGGTTT";

        }

        else if("Taenia laticollis".equals(s))

        {

           q=
"GGATTTGGTATAATTAGACATATATGTTTAAGTATTAGTATGTGTTCGGATGCTTTCGGTTTTTATGGTTTATTATTTGCTATG
TTTTCTATTGTTTGTTTAGGGAGAAGAGTTTGGGGTCATCATATGTTTACAGTTGGGTTAGATGTTAAGACGGCTGTATTTTTT
AGTTCTGTAACTATGATTATTGGTGTACCTACAGGTATAAAGGTTTTTACATGATTATATATGCTTTTAAATTCTCGGGTTAAA
AAGAGTGATCCTGTATTATGGTGGATAGTTTCTTTTATAGTTTTGTTTACGTTTGGTGGTGTTACAGGAATAGTGTTGTCTGCT
TGCGTATTAGATAAAGTATTACATGATACTTGATTT";

        }

        else if("Taenia madoquae".equals(s))

        {

           q=
"GGTTTTGGGATAATTAGTCATATATGTTTGAGGATTAGTATGTGTCCTGATGCTTTTGGTTTTTATGGTTTGTTATTTGCTATG
TTCTCAATAGTGTGTTTGGGAAGGAGTGTATGAGGTCATCACATGTTTACGGTTGGATTAGATGTTAAGACTGCTGTATTTTTT
TAGTTCGGTTACTATGATAATAGGAGTACCTACAGGAATAAAGGTTTTTACTTGACTTTATATGCTTTTAAATTCTCGTGTGA
ATAAGAGTGATCCTGTGTTATGATGGATAGTTTCTTTTATAGTATTGTTTACTTTTGGTGGTGTAACTGGTATTGTATTATCTG
CTTGTGTATTGGATAATGTTTTACATGATACTTGATTT";

        }

        else if("Taenia martis".equals(s))

        {

           q=
"GGTTTTGGTATAATTAGACATATTTGTCTAAATATAAGTATGAATTATGATTCTTTTGGTTTTTATGGTTTATTATTTGCTATG
TTTTCTATAGTTTGTTTGGGTAGTAGTGTGTGGGGTCATCATATGTTTACTGTTGGATTAGATGTTAAGACTGCTGTTTTTTTC
AGTTCAGTTACTATGATAATAGGTGTTCCTACGGGTATAAAGGTTTTTACTTGATTATATATGCTTTTAAAATCTCGTGTGAAT
AAGAGTGATCCTGTATTATGGTGAATTGTTTCTTTTATTATTTTGTTTACTTTTGGTGGTGTTACTGGTATAGTGTTATCTGCAT
GTGTTTTGGATAAAGTTCTTCATGATACTTGATTT";

        }

        else if("Taenia multiceps".equals(s))

        {
```

```
      q=
"GGTTTTGGTATAATTAGTCACATATGTTTAAGAATAAGCATGTGTCCAGATGCTTTTGGTTTTTATGGTTTATTATTTGCTATG
TTTTCAATAGTGTGTTTAGGGAGAAGTGTGTGAGGCCATCATATGTTTACAGTTGGGTTAGATGTTAAGACTGCTGTATTTTTT
AGTTCGGTTACTATGATAATAGGAGTGCCCACAGGAATAAAGGTTTTTACTTGGCTTTATATGCTTTTAAATTCTCGTGTAAA
CAAGAGTGATCCTATACTATGATGAATAGTTTCTTTTATAGTATTGTTTACTTTTGGTGGTGTAACTGGGATTGTATTGTCTGC
TTGTGTATTAGATAAAGTTTTACATGATACTTGATTT";

    }

    else if("Taenia mustelae".equals(s))

    {

      q=
"GGTTTTGGTATTATTGGTCATATATGTTTGAGTATAAGGATGTGTTCTGATGCTTTTGGGTTTTATGGATTGTTGTTTGCTATG
TTTTCTATTGTTTGTCTAGGTAGTAGAGTTTGAGGGCATCATATGTTTACTGTTGGTTTAGATGTTAAGACTGCTGTTTTTTTA
GTTCTGTTACTATGATTATAGGAGTTCCTACTGGTATAAAGGTGTTTACTTGGTTGTATATGTTACTGAATTCTAGTGTTAACA
AGAGGGATCCTGTGTTGTGATGAATAGTGTCATTTATATTTTTGTTTACTTTTGGTGGTGTTACTGGTATAGTTTTGTCTGCTT
GTGTATTAGATAATGTTTTGCATGATACTTGATTT";

    }

    else if("Taenia ovis".equals(s))

    {

      q=
"GGATTTGGTATAATTAGTCATATTTGTTTGAGGATTAGTATGTGTCCAGATGCTTTTGGTTTTTATGGCTTATTATTTGCTATG
TTTTCTATAGTATGTTTAGGAAGAAGTGTGTGGGGGCATCATATGTTTACTGTTGGGTTGGATGTTAAGACGGCTGTATTTTTT
AGTTCGGTTACTATGATCATAGGTGTGCCTACTGGTATAAAGGTTTTTACTTGGCTTTATATGCTTCTGAAATCTCGTGTGAAT
AAGAGTGATCCTATTTTGTGATGGATAGTTTCTTTTATAGTATTATTTACTTTTGGAGGTGTGACTGGTATTGTTTTATCTGCTT
GTGTATTGGATAAAGTTCTTCATGATACTTGATTT";

    }

    else if("Taenia parva".equals(s))

    {

      q=
"GGTTTGGGATTATAAGACATATATGTTTAAGAATTAGTATGTGTGATGATGCTTTTGGTTTTTATGGTTTGTTATTTGCTATGT
TTTCTATTGTGTGTTTAGGAAGAAGTGTATGAGGCCATCATATGTTTACTGTAGGTTTAGATGTGAAGACTGCTGTGTTTTTTA
GTTCAGTAACAATGATTATCGGGGTTCCTACTGGGATAAAGGTTTTTACTTGATTATATATGTTACTTAATTCTCGTATTAATA
AGGGTGATCCTGTAATTTGATGAATTGTTTCTTTCATAGTTTTATTTACGTTTGGTGGTGTCACTGGTATAGTTTTATCAGCTT
GTGTTTTAGATAAAGTTTTGCATGATACTTGATTT";

    }

    else if("Taenia pisiformis".equals(s))

    {

      q=
"GGGTTTGGTATAATTAGTCATATATGTTTAAGAATAAGTATGTGTTCAGATGCGTTTGGTTTTTATGGTTTATTGTTTGCAAT
GTTTTCTATAGTTTGTTTAGGTAGAAGTGTATGAGGTCATCATATGTTTACTGTTGGATTAGATGTAAAGACCGCTGTGTTTTT
TAGTTCAGTAACAATGATAATTGGAGTACCTACTGGAATTAAGGTCTTTACATGACTTTATATGCTTTTAAATTCTCGTGTCA
AAAAGAGTGATCCTGTGTTGTGGTGAATAATTTCTTTTATAGTCTTATTTACTTTTGGAGGTGTAACTGGTATAGTATTATCTG
CTTGTGTTTTAGATAAAGTT-TTACATGATACTTGATTT";

    }

    else if("Taenia saginata".equals(s))

    {

      q=
"GGTTTTGGTATGATTAGTCATATATGTTTAAGAATAAGTATGTGTCCAGATGCTTTTGGTTTTTATGGTTTGTTGTTTGCTATG
TTTTCAATAGTGTGTTTGGGGAGAAGTGTGTGGGGTCATCATATGTTTACGGTTGGGTTAGATGTTAAGACTGCTGTGTTTTTT
AGTTCGGTTACTATGATAATAGGAGTACCAACAGGAATAAAGGTTTTTACTTGACTTTATATGCTTTTAAATTCTCGTGTAAA
```

```
TAAGAGTGATCCTATATTGTGGTGAATAGTTTCTTTTATAGTGTTGTTTACTTTTGGTGGTGTGACTGGTATTGTGTTGTCTGC
TTGCGTATTGGATAAAGTTTTGCATGATACTTGATTT";

      }

      else if("Taenia serialis".equals(s))

      {

         q=
"GGTTTTGGTATAATTAGTCATATATGTTTAAGAATAAGTATGTGTCCAGACGCTTTTGGTTTTTATGGTTTGTTATTTGCTATG
TTCTCAATAGTGTGTTTAGGAAGGAGTGTATGGGGTCATCATATGTTTACAGTTGGGTTAGATGTTAAGACTGCTGTATTTTTT
AGCTCAGTTACTATGGTAATAGGAGTACCAACAGGAATAAAGGTTTTTACTTGGCTTTATATGTTATTAAATTCTCGTGTGAA
TAAGAGTGATCCTATATTGTGGTGGATAGTTTCTTTTATAGTATTGTTTACTTTTGGTGGTGTAACTGGGATTGTATTGTCTGC
TTGTGTGTTGGATAAAGTTTTACATGATACTTGGTTT";

      }

      else if("Taenia solium".equals(s))

      {

         q=
"GGGTTTGGTATAATTAGTCATATATGTTTGAGTATAAGTATGTGTTCTGATGCTTTTGGCTTTTATGGGTTATTGTTTGCTATG
TTTTCAATAGTATGTTTAGGAAGAAGTGTGTGAGGACATCATATGTTTACGGTTGGGTTAGATGTTAAGACGGCTGTATTTTTT
TAGTTCTGTTACTATGATAATTGGAGTGCCTACGGGGATTAAGGTTTTTACTTGGCTTTATATGCTTTTAAAATCTCGTGTTAA
TAAGAGTGATCCGGTTTTATGATGAATAATTTCGTTTATAGTATTGTTTACATTTGGTGGTGTAACCGGTATTATTCTATCTGC
TTGTGTATTAGATAAAGTTCTTCATGATACTTGGTTT";

      }

      else if("Taenia twitchelli".equals(s))

      {

         q=
"GGTTTTGGTATAATTAGACATATTTGTTTAAATGTAAGTATGAATTATGATTCTTTTGGATTTTATGGTTTGTTATTTGCTATG
TTTTCTATAGTTTGTTTAGGTAGAAGTGTATGAGGTCATCATATGTTTACTGTTGGATTAGATGTTAAGACTGCTGTTTTTTTT
AGTTCTGTTACTATGATTATAGGGGTTCCTACAGGTATAAAGGTGTTTACTTGGTTATATATGCTTTTAAAATCTCGTGTAAAT
AAGAGTGATCCTGTTTTATGATGAATTGTGTCTTTTATTATTTTGTTTACTTTTGGTGGTGTTACTGGTATAGTGTTATCTGCGT
GTGTTTTGGATAAAGTTCTTCATGATACTTGGTTT";

      }

      else if("Taeniopygia guttata".equals(s))

      {

         q=
"GGTTTCGGCATCATCTCCCACGTCGTAACCTACTATTCAGGTAAAAAAGAACCATTCGGATATATAGGAATAGTATGAGCTA
TGCTATCCATCGGATTCCTAGGATTCATCGTATGAGCCCACCACATGTTTACAGTAGGAATGGACGTAGACACCCGAGCATA
CTTTACATCCGCCACTATAATCATCGCCATCCCAACCGGCATCAAAGTATTCAGCTGACTAGCAACACTCCACGGAGGCACA
ATCAAGTGAGACCCACCAATACTATGAGCTCTAGGATTTATCTTCCTATTCACCATCGGAGGCCTAACCGGAATCGTCCTGGC
CAACTCCTCACTAGACATCGCCCTACACGACACCTACTACGTAGTAGCCCACTTCCACTACGTCCTATCAATAGGAGCAGTGT
TTGCAATCCTAGCAGGATTCACCCACTGATT";

      }



    double AA=0,AC=0,AG=0,AT=0,CA=0,CC=0,CG=0,CT=0,GA=0,GC=0,GG=0,GT=0,TA=0,TC=0,TG=0,TT=0;

    double length = q.length();

    for(int i=0;i<length-1;i++)

    {
```

```
if(q.charAt(i)=='A')
{
   if(q.charAt(i+1)=='A')
   {
                  AA++;
   }
   else if(q.charAt(i+1)=='C')
   {
                  AC++;
   }
   else if(q.charAt(i+1)=='G')
   {
                  AG++;
   }
   else if(q.charAt(i+1)=='T')
   {
                  AT++;
   }
}
else if(q.charAt(i)=='C')
{
      if(q.charAt(i+1)=='A')
      {
         CA++;
      }
      else if(q.charAt(i+1)=='C')
      {
         CC++;
      }
      else if(q.charAt(i+1)=='G')
      {
         CG++;
      }
      else if(q.charAt(i+1)=='T')
      {
            CT++;
        }
}
else if(q.charAt(i)=='G')
{
        if(q.charAt(i+1)=='A')
        {
            GA++;
        }
        else if(q.charAt(i+1)=='C')
        {
            GC++;
        }
        else if(q.charAt(i+1)=='G')
        {
            GG++;
        }
        else if(q.charAt(i+1)=='T')
        {
            GT++;
        }
}
else if(q.charAt(i)=='T')
{
        if(q.charAt(i+1)=='A')
        {
            TA++;
        }
        else if(q.charAt(i+1)=='C')
        {
            TC++;
        }
          else if(q.charAt(i+1)=='G')
        {
            TG++;
            }
            else if(q.charAt(i+1)=='T')
            {
                TT++;
            }
    }
}

AA=AA/length;
AC=AC/length;
AG=AG/length;
AT=AT/length;
CA=CA/length;
CC=CC/length;
CG=CG/length;
CT=CT/length;
GA=GA/length;
GC=GC/length;
GT=GT/length;
GG=GG/length;
TA=TA/length;
TC=TC/length;
TG=TG/length;
TT=TT/length;

AA=AA*10000;
AA=Math.round(AA);
AA=AA/10000;
AC=AC*10000;
AC=Math.round(AC);
AC=AC/10000;
AG=AG*10000;
AG=Math.round(AG);
AG=AG/10000;
AT=AT*10000;
```

```
AT=Math.round(AT);

AT=AT/10000;

CA=CA*10000;

CA=Math.round(CA);

CA=CA/10000;

CC=CC*10000;

CC=Math.round(CC);

CC=CC/10000;

CG=CG*10000;

CG=Math.round(CG);

CG=CG/10000;

CT=CT*10000;

CT=Math.round(CT);

CT=CT/10000;

GA=GA*10000;

GA=Math.round(GA);

GA=GA/10000;

GC=GC*10000;

GC=Math.round(GC);

GC=GC/10000;

GG=GG*10000;

GG=Math.round(GG);

GG=GG/10000;

GT=GT*10000;

GT=Math.round(GT);

GT=GT/10000;

TA=TA*10000;

TA=Math.round(TA);

TA=TA/10000;

TC=TC*10000;

TC=Math.round(TC);

TC=TC/10000;

TG=TG*10000;

TG=Math.round(TG);

TG=TG/10000;
```

```
TT=TT*10000;
TT=Math.round(TT);
TT=TT/10000;
   JFrame res= new JFrame("Dinucleotide frequency"+" "+s);
  res.setSize(420,200);
  res.setLayout(null);
  res.setLocationRelativeTo(null);
  JLabel f =new JLabel();
  f.setText(" AA");
  f.setSize(50,50);
  f.setFont(f.getFont().deriveFont(14.0f));
  JLabel f1 =new JLabel();
  f1.setText(" AC");
  //f1.setSize(50,50);
  f1.setBounds(0,30,50,50);
  f1.setFont(f1.getFont().deriveFont(14.0f));
  JLabel f2 =new JLabel();
  f2.setText(" AG");
  f2.setBounds(0,60,50,50);
  f2.setSize(50,50);
  f2.setFont(f2.getFont().deriveFont(14.0f));
  JLabel f3 =new JLabel();
  f3.setText(" AT");
  f3.setSize(50,50);
  f3.setBounds(0,90,50,50);
  f3.setFont(f3.getFont().deriveFont(14.0f));
  JLabel f4 =new JLabel();
  f4.setText(" CA");
  f4.setBounds(200,0,50,50);
  f4.setSize(50,50);
  f4.setFont(f4.getFont().deriveFont(14.0f));
  JLabel f5 =new JLabel();
  f5.setText(" CC");
  f5.setBounds(200,30,50,50);
  f5.setSize(50,50);
```

```
f5.setFont(f5.getFont().deriveFont(14.0f));

JLabel f6 =new JLabel();

f6.setBounds(200,60,50,50);

f6.setText(" CG");

f6.setSize(50,50);

f6.setFont(f6.getFont().deriveFont(14.0f));

JLabel f7 =new JLabel();

f7.setText(" CT");

f7.setBounds(200,90,50,50);

f7.setSize(50,50);

f7.setFont(f7.getFont().deriveFont(14.0f));

JLabel f8 =new JLabel();

f8.setText(" GA");

f8.setBounds(100,0,50,50);

f8.setSize(50,50);

f8.setFont(f8.getFont().deriveFont(14.0f));

JLabel f9 =new JLabel();

f9.setText(" GC");

f9.setBounds(100,30,50,50);

f9.setSize(50,50);

f9.setFont(f9.getFont().deriveFont(14.0f));

JLabel f10 =new JLabel();

f10.setText(" GG");

f10.setBounds(100,60,50,50);

f10.setSize(50,50);

f10.setFont(f10.getFont().deriveFont(14.0f));

JLabel f11 =new JLabel();

f11.setText(" GT");

f11.setBounds(100,90,50,50);

f11.setSize(50,50);

f11.setFont(f11.getFont().deriveFont(14.0f));

JLabel f12 =new JLabel();

f12.setText(" TA");

f12.setBounds(300,0,50,50);

f12.setSize(50,50);
```

```
 f12.setFont(f12.getFont().deriveFont(14.0f));
 JLabel f13 =new JLabel();
 f13.setText(" TC");
 f13.setBounds(300,30,50,50);
 f13.setSize(50,50);
 f13.setFont(f13.getFont().deriveFont(14.0f));
 JLabel f14 =new JLabel();
 f14.setText(" TG");
 f14.setBounds(300,60,50,50);
 f14.setSize(50,50);
 f14.setFont(f14.getFont().deriveFont(14.0f));
 JLabel f15 =new JLabel();
 f15.setText(" TT");
 f15.setBounds(300,90,50,50);
 f15.setSize(50,50);
 f15.setFont(f15.getFont().deriveFont(14.0f));
  String s1 = String.valueOf(AA);
 String s2 = String.valueOf(AC);
 String s3 = String.valueOf(AG);
 String s4 = String.valueOf(AT);
 String s5 = String.valueOf(CA);
 String s6 = String.valueOf(CC);
 String s7 = String.valueOf(CG);
 String s8 = String.valueOf(CT);
 String s9 = String.valueOf(GA);
 String s10 = String.valueOf(GC);
 String s11 = String.valueOf(GG);
 String s12 = String.valueOf(GT);
 String s13 = String.valueOf(TA);
 String s14 = String.valueOf(TC);
 String s15 = String.valueOf(TG);
 String s16 = String.valueOf(TT);
JTextField A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13,A14,A15,A16;
 A1 = new JTextField(s1);
 A1.setBounds(40,15,50,25);
```

```
A1.setEditable(false);

res.add(A1);

A2 = new JTextField(s2);

A2.setBounds(40,45,50,25);

A2.setEditable(false);

res.add(A2);

A3 = new JTextField(s3);

A3.setBounds(40,75,50,25);

A3.setEditable(false);

res.add(A3);

A4 = new JTextField(s4);

A4.setBounds(40,105,50,25);

A4.setEditable(false);

res.add(A4);

A5 = new JTextField(s5);

A5.setBounds(240,15,50,25);

A5.setEditable(false);

res.add(A5);

A6 = new JTextField(s6);

A6.setBounds(240,45,50,25);

A6.setEditable(false);

res.add(A6);

A7 = new JTextField(s7);

A7.setBounds(240,75,50,25);

A7.setEditable(false);

res.add(A7);

A8 = new JTextField(s8);

A8.setBounds(240,105,50,25);

A8.setEditable(false);

res.add(A8);

A9 = new JTextField(s9);

A9.setBounds(140,15,50,25);

A9.setEditable(false);

res.add(A9);

A10 = new JTextField(s10);
```

```
A10.setBounds(140,45,50,25);

A10.setEditable(false);

res.add(A10);

A11 = new JTextField(s11);

A11.setBounds(140,75,50,25);

A11.setEditable(false);

res.add(A11);

A12 = new JTextField(s12);

A12.setBounds(140,105,50,25);

A12.setEditable(false);

res.add(A12);

A13 = new JTextField(s13);

A13.setBounds(340,15,50,25);

A13.setEditable(false);

res.add(A13);

A14 = new JTextField(s14);

A14.setBounds(340,45,50,25);

A14.setEditable(false);

res.add(A14);

A15 = new JTextField(s15);

A15.setBounds(340,75,50,25);

A15.setEditable(false);

res.add(A15);

A16 = new JTextField(s16);

A16.setBounds(340,105,50,25);

A16.setEditable(false);

res.add(A16);

res.add(f);

res.add(f1);

res.add(f2);

res.add(f3);

res.add(f4);

res.add(f5);

res.add(f6);

res.add(f7);
```

```
        res.add(f8);

        res.add(f9);

        res.add(f10);

        res.add(f11);

        res.add(f12);

        res.add(f13);

        res.add(f14);

        res.add(f15);

        res.setVisible(true);

    }


    private void jComboBox1ActionPerformed(java.awt.event.ActionEvent evt) {

        // TODO add your handling code here:

        s = (String) jComboBox1.getSelectedItem();


    }


    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

        // TODO add your handling code here:

        String q="";


        if("Angiostrongylus cantonensis".equals(s))

        {

            q=
"GCTTTTGGGATTGTTAGACAGTCTACTTTATATTTAACGGGTAAAAAAGAGGTTTTTGGTTATTTGGGTATGGTTTATGCTAT
TTTAAGAATTGGTTTGATTGGTTGTGTGGTTTGGGCTCATCATATATATACGGTTGGTATGGATTTAGATTCTCGTGCTTATTT
TACTGCGGCTACTATAGTTATTGCGGTTCCTACGGGAGTGAAGGTTTTTAGGTGGTTGGCAACTTTATTTGGTATAAAGATAT
TGTTTCAACCTATTTTATTGTGGGTTTTGGGTTTTATTTTTTTATTTACTATCGGTGGGTTAACGGGGGTTATATTGTCTAATTC
TAGTTTGGATATTATTTTACATGATACTTATTATGTAGTTAGGCATTTTCATTATGTTT";

        }

        else if("Angiostrongylus costaricensis".equals(s))

        {

           q=
"GCTTTTGGGATTATTAGTCAATCTGCTTTGTATTTGTCAGGGAAGAAAGAGGTTTTTGGTTATTTAGGGATGGTTTATGCGAT
TTTAAGAATTGGGTTGATTGGGTGTGTAGTTTGAGCTCATCATATGTATACTGTTGGTATGGATTTGGATTCTCGTGCTTACTT
TACTGCAGCTACAATAGTTATTGCGGTTCCTACTGGGGTTAAAGTGTTTAGTTGGTTGGCTACACTTTATGGGATGAAAATGA
TGTTTCAGCCGATTTTGTTGTGGGTTATGGGGTTTATTTTTTTGTTTACTATTGGGGGTTTGACCGGGGTTATGTTATCTAATTC
AAGTTTGGATATTATTTTGCATGATACTTATTATGTGGTT";

        }

        else if("Angiostrongylus vasorum".equals(s))
    {

        q=
"GCTTTTGGGATTGTTAGTCAGTCGACTTTATATTTGACTGGGAAGAAGGAGGTGTTTGGTTATTTGGGGATGGTTTATGCGAT
TTTAAGGATTGGTTTGATTGGTTGTGTGGTGTGAGCTCATCATATGTATACTGTTGGTATAGATTTAGATTCTCGTGCTTATTT
TACTGCGGCTACTATGGTGATTGCGGTGCCGACTGGAGTGAAGGTTTTTAGTTGGTTGGCTACTTTGTTTGGGATGAAGTATA
GTATTTCAGCCTATTTTGTTGTGGGTTATAGGATTTATTTTTTTTTATTTACTATTGGGGGTTTGACGGGTGTGATATTGTCAAA
TTCGAGATTGGATATTATTTTACATGATACGTATTATGTGGTAAGTCATTTTCATTATGTGAGGTTGTTTCATGATACTTGGTT
T";

    }

    else if("Diplogonoporus balaenopterae".equals(s))

    {

        q=
"GGGTTTGGAATGATTAGACATGTTTGTAGTAACTTAGGTTGTTCATATGATACTTTTGGATTTTATGGTTTATTATTTGCTATG
TTTTCTATCGTTTGTCTTGGTAGGGTTGTATGGGGTCATCATATGTTTACGGTAGGTTTAGATGTGAAGACAGCTGTTTTTTTT
AGGTCAGTTACTATGATTATAGGGGTGCCTACTGGAATAAAGGTGTTTTCTTGGCTGTATATGATTTTAAAAAGTCGTGTCTC
TTTGCGTGAACCTATATTTTGGTGGGTGTTATCTTTTATCGTGCTGTTTACAATAGGGGGTGTTACTGGTATTATACTTTCTGC
TTGTGTTCTTGATAATATTTTGCATGATACTTGATTT";

    }

    else if("Diplogonoporus grandis".equals(s))

    {

        q=
"GGGTTTGGAATGATTAGACATGTTTGTAGTAACTTAGGTTGTTCATATGATACTTTTGGATTTTATGGTTTATTATTTGCTATG
TTTTCTATCGTTTGTCTTGGTAGGGTTGTATGGGGTCATCATATGTTTACGGTAGGTTTAGATGTGAAGACAGCTGTTTTTTTT
AGGTCAGTTACTATGATTATAGGGGTGCCTACTGGAATAAAGGTGTTTTCTTGGCTGTATATGATTTTAAAAAGTCGTGTCTC
TTTGCGTGAACCTATATTTTGGTGGGTGTTATCTTTTATCGTGCTGTTTACAATAGGGGGTGTTACTGGTATTATACTTTCTGC
TTGTGTTCTTGATAATATTTTGCATGATACTTGATTT";

    }

    else if("Aelurostrongylus abstrusus".equals(s))

    {

        q=
"GCTTTTGGTATTGTTAGTCAGTCTACTTTGTATTTGACGGGGAAGAAGGAAGTTTTTGGTTATTTAGGGATAGTTTATGCTAT
TATAAGTATTGGTTTAATTGGTTGTGTTGTTTGGGCTCATCATATATATACTGTTGGTATAGATTTGGATTCTCGTGCTTATTTT
ACGGCGGCTACGATGGTTATTGCTGTGCCAACGGGTGTTAAGGTTTTTAGTTGGTTGGCTACTTTATTTGGAATGAAGATAGT
GTTTCAGCCGGTTTTGTTGTGGGTTTTGGGTTTTATTTTTTTGTTTACTATTGGGGGGTTAACTGGGGTCATGCTTTCGAATTCT
AGTTTGGATATTATTTTGCATGATACTTATTATGTGGTTAGTCATTTTCATTATGTGTTGAGTTT";

    }

    else if("Dictyocaulus eckerti".equals(s))

    {

        q=
"GCTTTTGGTATTATTAGTCAGTCAACTTTGTATTTAACTGGTAAAAAGGAGGTTTTTGGTTCTTTGGGTATGGTTTATGCTATT
TTAAGTATTGGTTTAATTGGTTGTGTAGTATGAGCACATCATATATATACTGTTGGAATAGATTTGGATTCGCGTGCTTATTTT
ACTGCTGCTACTATGGTAATTGCTGTTCCTACGGGTGTAAAAGTTTTTAGTTGGTTGGCTACTTTGTATGGTTTAAAAATAGTA
TATAATCCTTTGTTGTTATGGGTTTTGGGTTTTATTTTTTTATTTACTATTGGGGGGTTAACTGGAGTTATTTTGTCAAATTCTA
GTTTAGATATTTTGTTACATGATACTTATTATGTTGTAAGGCATTT";

    }

    else if("Dictyocaulus viviparus".equals(s))

    {
          q=
"GCTTTTGGTATTATTAGACAATCTACTTTGTATTTAACTGGTAAAAAAGAGGTTTTTGGTTCTTTGGGTATGGTTTATGCTATT
TTAAGTATTGGTTTAATTGGTTGTGTTGTGTGGGCACATCATATGTATACTGTTGGGATGGATTTGGATTCGCGTGCTTATTTT
ACTGCTGCTACTATAGTAATTGCTGTTCCTACTGGAGTTAAGGTTTTTAGATGATTGGCTACTTTATATGGATTGAAAATGGTT
TATAATCCTTTGTTGTTGTGAGTTTTAGGTTTTATTTTTTTGTTTACTATTGGTGGTTTAACTGGTGTTATTTTGTCAAATTCTA
GTCTTGATATTTTGTTGCATGATACTTATTAT";

        }

        else if("Ascaris lumbricoides".equals(s))

        {

          q=
"GCTTTTGGTATTATTAGTCAGAGTAGTTTGTATTTGACTGGTAAAAAGGAGGTTTTTGGGTCTTTGGGTATGGTTTATGCTAT
TTTAAGTATTGGTTTGATTGGTTGTGTTGTTTGAGCTCATCATATGTATACTGTTGGTATGGATCTTGATTCTCGGGCTTATTTT
ACTGCTGCAACTATGGTTATTGCTGTTCCTACTGGTGTTAAGGTTTTTAGTTGGTTGGCTACCTTGTTTGGTATAAAAATGGTT
TTTCAGCCTTTACTTTTATGAGTTATGGGTTTTATTTTTTTGTTTACTATTGGTGGGTTAACCGGGGTTATACTTTCTAATTCTA
GTTTGGATATTATCTTGCATGATACTTATTATGTTGTTAGTCATTTTCATTATGTTCTTAGTTT";

        }

        else if("Ascaris suum".equals(s))

        {

          q=
"GCTTTTGGTATTATTAGTCAGAGTAGTTTGTATCTGACTGGTAAAAAGGAGGTTTTTGGGTCTTTGGGTATGGTTTATGCTAT
TTTGAGTATTGGTTTAATTGGTTGTGTTGTTTGAGCTCATCATATGTATACTGTTGGTATGGATCTTGACTCTCGGGCTTATTTT
ACTGCTGCAACTATGGTTATTGCTGTTCCTACTGGTGTTAAGGTTTTTAGTTGGTTGGCTACCTTGTTTGGTATAAAGATGGTT
TTTCAACCTTTACTTTTATGAGTTATGGGTTTTATTTTTTTGTTTACTATTGGTGGGTTAACCGGGGTTATGCTTTCTAATTCTA
GTTTGGATATTATCTTGCATGATACTTA-TTATGTTGTTAGTCATTTTCATTATGTCCTTAGTTT";

        }

        else if("Baylisascaris ailuri".equals(s))

        {

          q=
"GCTTTTGGTATTATTAGCCAGAGTAGGTTGTATTTAACTGGTAAAAAGGAAGTTTTTGGTTCTTTGGGTATGGTTTATGCTAT
TTTGAGTATTGGTTTAATTGGTTGTGTTGTTTGGGCTCATCATATGTATACTGTAGGTATGGATTTGGATTCTCGTGCTTATTTT
ACTGCGGCTACTATGGTTATTGCGGTCCCTACGGGTGTTAAGGTTTTTAGTTGGTTGGCTACTTTGTTTGGTATGAAGATGGTG
TTCCAGCCTTTACTTTTGTGAGTTATGGGTTTTATTTTTTTATTTACTATTGGCGGATTGACTGGGGTGATGCTTTCTAATTCTA
GTTTGGATATTATTTTGCATGATACTTA-TTATGTTGTTAGACATTTTCATTATGTTCTTAGTTT";

        }

        else if("Baylisascaris procyonis".equals(s))

        {

          q=
"GCTTTTGGTATTATTAGCCAAAGTAGGTTGTATTTAACTGGTAAGAAGGAAGTTTTTGGTTCTTTGGGAATGGTTTATGCTAT
TTTGAGTATTGGTTTGATTGGATGTGTGGTTTGGGCTCATCATATGTATACTGTGGGTATGGATTTGGATTCTCGGGCTTATTT
TACTGCGGCTACTATGGTTATTGCGGTTCCTACGGGAGTTAAGGTTTTTAGTTGGTTGGCCACTTTATTTGGTATGAAGATAGT
GTTTCAGCCTTTGCTTTTGTGGGTTATGGGGTTTATTTTTTTGTTTACTATTGGTGGTTTGACTGGGGTTATGCTTTCTAATTCT
AGTTTGGATATTATTTTGCATGATACTTATTATGTTGTTAGGCATTTTCATTATGTTCTTAGTTT";

        }

        else if("Baylisascaris schroederi".equals(s))

        {

          q=
"GCTTTTGGTATTATTAGTCAGAGTAGGTTGTATCTGACTGGTAAGAAGGAGGTTTTTGGTTCTTTGGGTATGGTTTACGCAAT
TTTGAGTATTGGTTTGATTGGTTGTGTTGTTTGAGCTCATCATATGTATACTGTGGGTATAGATTTGGATTCTCGTGCTTATTTT
ACTGCGGCTACTATGGTTATCGCAGTTCCTACGGGTGTTAAGGTTTTTAGTTGGTTGGCTACTTTGTTTGGTATGAAGATGGTG
```

```
TTTCAGCCTTTGCTTTTGTGGGTTATAGGATTTATTTTTTTGTTTACTATTGGTGGATTGACTGGGGTGATGCTTTCTAATTCTA
GTTTGGATATTATTTTGCATGATACTTA-TTATGTTGTTAGGCATTTTCATTATGTTCTTAGTTT";

        }

        else if("Baylisascaris transfuga".equals(s))

        {

            q=
"GCTTTTGGTATTATTAGTCAGAGTAGATTGTATTTAACTGGTAAAAAGGAGGTTTTTGGTTCTTTGGGTATGGTTTATGCTAT
TTTGAGTATTGGTTTAATTGGTTGTGTTGTTTGGGCTCATCATATGTATACTGTAGGTATGGATTTGGATTCTCGTGCTTATTTT
ACTGCGGCTACTATGGTTATTGCGGTTCCTACAGGTGTTAAGGTTTTTAGTTGGTTGGCCACTTTGTTTGGTATGAAGATGGTG
TTTCAGCCTTTACTTTTGTGGGTTATGGGGTTTATTTTTTTATTTACTATTGGTGGATTGACTGGGGTGATGCTTTCTAATTCTA
GTTTGGATATTATTTTGCATGATACTTATTATGTTGTTAGGCATTTTCATTATGTTCTTAGTTT";

        }

        else if("Toxocara canis".equals(s))

        {

            q=
"GCTTTTGGTATTATTAGGCAAAGTAGTTTGTATTTGACTGGTAAAAAGGAGGTTTTTGGTTCTTTAGGCATGGTTTATGCTAT
TTTAAGTATTGGTCTGATTGGCTGTGTAGTTTGGGCTCACCATATGTATACGGTGGGCATGGATTTGGATTCTCGTGCTTATTT
TACTGCGGCAACGATGGTTATTGCTGTGCCTACGGGGGTTAAGGTTTTTAGTTGGTTAGCCACTCTTTTTGGTATGAAGATGG
TGTTTCAACCTTTGCTTTTGTGGGTGCTGGGTTTTATTTTTTTATTTACTATCGGGGGGTTGACTGGTGTTATGTTATCTAATTC
TAGGTTGGACATTATCTTGCATGATACTTATTATGTTGTTAGTCATTTTCATTATGTTT";

        }

        else if("Toxocara cati".equals(s))

        {

            q=
"GATTTTTTGGGCATCCTGAGGTTTATATTTTGATTTTACCTGCCTTTGGTATTATTAGTCAAAGTAGTTTATATTTAACTGGTA
AGAAGGAGGTTTTTGGTTCTTTGGGCATGGTCTATGCTATTTTGAGTATTGGTTTGATTGGTTGTGTGGTGTGAGCTCACCACA
TGTATACTGTTGGTATAGACTTGGATTCTCGGGCTTATTTTACTGCGGCTACTATGGTTATCGCTGTGCCTACGGGTGTTAAGG
TTTTTAGTTGGTTGGCTACTCTTTTTGGTATAAAAATGGTTTTTCAACCTTTGCTTTTGTGAGTGTTGGGTTTTATTTTTTTGTTT
ACTATTGGTGGGCTTACTGGAGTTATGCTTTCTAATTCTAGTTTGGATATTATTTTGCATGACACCTATTATGTTGTGAGGCAT
TTCCACTATGTTT";

        }else if("Toxocara malaysiensis".equals(s))

        {

            q=
"GCTTTTGGTATTATTAGTCAGAGTAGTTTGTATTTAACTGGTAAGAAGGAAGTTTTTGGTTCGTTGGGGATGGTTTATGCTAT
TTTAAGTATTGGTTTGATTGGCTGTGTGGTTTGGGCTCATCATATGTATACCGTGGGTATAGATTTGGATTCTCGGGCTTATTT
TACTGCGGCGACTATGGTTATTGCTGTGCCTACTGGTGTTAAGGTTTTTAGTTGGTTGGCTACTCTTTTTGGTATGAAAATGGT
TTTTCAGCCTTTACTTTTATGGGTGTTAGGTTTTATTTTCTTGTTTACTATTGGGGGCCTTACTGGTGTGATGCTTTCTAATTCT
AGCCTTGATATTATTTTGCATGATACCTATTATGTTGTTAGACATTTTCATTATGTTT";

        }

        else if("Taenia asiatica".equals(s))

        {

            q=
"GGTTTTGGTATGATTAGTCATATATGTTTAAGAATAAGTATGTGTCCGGATGCTTTTGGTTTTTATGGTTTGTTATTTGCTATG
TTTTCAATAGTATGTTTGGGGAGAAGTGTGTGGGGTCATGATATGTTTACGGTTGGATTAGTTGTTAAGACTACTGTGTTTTTT
AGTTCGGTTACTATGATAATAGGAGTACCAACAGGAATAAAGGTTTTTACTTGACTTTATATGCTTTTAAATTCTCGTGTAAA
TAAGAGGGATCCTATATTGTGGTGGATAGTTTCTTTTATAGTGTTGTTTACCTTTGGTGGTGTGACTGGTATTGTGTTGTCTGC
TTGTGTATTGGATAAAGTTTTGCATGATACTTGATTT";

        }
        else if("Taenia crassiceps".equals(s))

        {

            q=
"GGTTTTGGAATTATTAGACATATTTGTTTGAAAATAAGTATGAATTGTGATTCTTTTGGTTTTTATGGATTGTTATTTGCTATG
TTTTCAATAGTTTGTTTAGGTAGGAGTGTTTGGGGTCATCATATGTTTACGGTTGGTTTAGATGTTAAGACTGCTGTTTTTTTT
AGTTCTGTTACTATGATTATAGGAGTACCTACAGGTATAAAGGTGTTTACTTGATTGTATATGCTTTTAAATTCGCGTGTGAA
CAAGAGTGATCCTATATTGTGGTGAATTGTTTCTTTTATAGTTTTATTTACGTTTGGTGGTGTTACTGGAATAGTATTGTCTGC
TTGTGTATTAGATAAAGTTCTTCATGATACTTGATTT";

        }

        else if("Taenia hydatigena".equals(s))

        {

            q=
"GGATTTGGAATTATTAGTCATATATGTTTGAGAATAAGTATGAGTCCTGATGCTTTTGGGTTCTATGGATTATTATTTGCTAT
GTTTTCAATAGTCTGTTTGGGTAGAAGTGTGTGGGGTCATCATATGTTTACTGTTGGGTTAGATGTTAAGACTGCTGTTTTTTT
TAGTTCTGTGACTATGATTATAGGTGTGCCTACTGGTATAAAGGTGTTTACTTGGTTATATATGCTTTTAAACTCTCATGTGAA
TAAGAGTGATCCTGTTGTTTGATGAATTGTTTCTTTTATAGTTTTGTTTACTTTTGGTGGGGTTACTGGTATTGTGTTGTCAGCA
TGTGTATTAGATAAAGTTCTTCATGATACCTGATTT";

        }

        else if("Taenia krepkogorski".equals(s))

        {

            q=
"GGATTTGGTATAATTAGTCATATATGTTTAAGAATAAGTATGTCTTCTGATGTGTTTGGGTTTTATGGTTTATTATTTGCTATG
TTTTCTATAGTTTGTTTAGGAAGAAGAGTGTGAGGTCATCACATGTTTACTGTTGGTTTAGATGTTAAGACTGCTGTGTTTTTT
AGCTCAATTACTATGATTATTGGTGTGCCTACTGGGATTAAGGTTTTTACATGATTATATATGTTATTAAATGCTCGAGTAAA
AAAGAGTGATCCTGTGTTGTGGTGAATTGTTTCATTTATAATATTGTTTACATTTGGTGGAGTTACTGGTATAGTATTGTCTGC
TTGTGTTTTAGATAAAGTGTTACATGATACTTGGTTT";

        }

        else if("Taenia laticollis".equals(s))

        {

            q=
"GGATTTGGTATAATTAGACATATATGTTTAAGTATTAGTATGTGTTCGGATGCTTTCGGTTTTTATGGTTTATTATTTGCTATG
TTTTCTATTGTTTGTTTAGGGAGAAGAGTTTGGGGTCATCATATGTTTACAGTTGGGTTAGATGTTAAGACGGCTGTATTTTTT
AGTTCTGTAACTATGATTATTGGTGTACCTACAGGTATAAAGGTTTTTACATGATTATATATGCTTTTAAATTCTCGGGTTAAA
AAGAGTGATCCTGTATTATGGTGGATAGTTTCTTTTATAGTTTTGTTTACGTTTGGTGGTGTTACAGGAATAGTGTTGTCTGCT
TGCGTATTAGATAAAGTATTACATGATACTTGATTT";

        }

        else if("Taenia madoquae".equals(s))

        {

            q=
"GGTTTTGGGATAATTAGTCATATATGTTTGAGGATTAGTATGTGTCCTGATGCTTTTGGTTTTTATGGTTTGTTATTTGCTATG
TTCTCAATAGTGTGTTTGGGAAGGAGTGTATGAGGTCATCACATGTTTACGGTTGGATTAGATGTTAAGACTGCTGTATTTTTT
TAGTTCGGTTACTATGATAATAGGAGTACCTACAGGAATAAAGGTTTTTACTTGACTTTATATGCTTTTAAATTCTCGTGTGA
ATAAGAGTGATCCTGTGTTATGATGGATAGTTTCTTTTATAGTATTGTTTACTTTTGGTGGTGTAACTGGTATTGTATTATCTG
CTTGTGTATTGGATAATGTTTTACATGATACTTGATTT";

        }

        else if("Taenia martis".equals(s))

        {
```

```
        q=
"GGTTTTGGTATAATTAGACATATTTGTCTAAATATAAGTATGAATTATGATTCTTTTGGTTTTTATGGTTTATTATTTGCTATG
TTTTCTATAGTTTGTTTGGGTAGTAGTGTGTGGGGTCATCATATGTTTACTGTTGGATTAGATGTTAAGACTGCTGTTTTTTTC
AGTTCAGTTACTATGATAATAGGTGTTCCTACGGGTATAAAGGTTTTTACTTGATTATATATGCTTTTAAAATCTCGTGTGAAT
AAGAGTGATCCTGTATTATGGTGAATTGTTTCTTTTATTATTTTGTTTACTTTTGGTGGTGTTACTGGTATAGTGTTATCTGCAT
GTGTTTTGGATAAAGTTCTTCATGATACTTGATTT";

      }

      else if("Taenia multiceps".equals(s))

      {

        q=
"GGTTTTGGTATAATTAGTCACATATGTTTAAGAATAAGCATGTGTCCAGATGCTTTTGGTTTTTATGGTTTATTATTTGCTATG
TTTTCAATAGTGTGTTTAGGGAGAAGTGTGTGAGGCCATCATATGTTTACAGTTGGGTTAGATGTTAAGACTGCTGTATTTTTT
AGTTCGGTTACTATGATAATAGGAGTGCCCACAGGAATAAAGGTTTTTACTTGGCTTTATATGCTTTTAAATTCTCGTGTAAA
CAAGAGTGATCCTATACTATGATGAATAGTTTCTTTTATAGTATTGTTTACTTTTGGTGGTGTAACTGGGATTGTATTGTCTGC
TTGTGTATTAGATAAAGTTTTACATGATACTTGATTT";

      }

      else if("Taenia mustelae".equals(s))

      {

        q=
"GGTTTTGGTATTATTGGTCATATATGTTTGAGTATAAGGATGTGTTCTGATGCTTTTGGGTTTTATGGATTGTTGTTTGCTATG
TTTTCTATTGTTTGTCTAGGTAGTAGAGTTTGAGGGCATCATATGTTTACTGTTGGTTTAGATGTTAAGACTGCTGTTTTTTTA
GTTCTGTTACTATGATTATAGGAGTTCCTACTGGTATAAAGGTGTTTACTTGGTTGTATATGTTACTGAATTCTAGTGTTAACA
AGAGGGATCCTGTGTTGTGATGAATAGTGTCATTTATATTTTTGTTTACTTTTGGTGGTGTTACTGGTATAGTTTTGTCTGCTT
GTGTATTAGATAATGTTTTGCATGATACTTGATTT";

      }

      else if("Taenia ovis".equals(s))

      {

        q=
"GGATTTGGTATAATTAGTCATATTTGTTTGAGGATTAGTATGTGTCCAGATGCTTTTGGTTTTTATGGCTTATTATTTGCTATG
TTTTCTATAGTATGTTTAGGAAGAAGTGTGTGGGGGCATCATATGTTTACTGTTGGGTTGGATGTTAAGACGGCTGTATTTTTT
AGTTCGGTTACTATGATCATAGGTGTGCCTACTGGTATAAAGGTTTTTACTTGGCTTTATATGCTTCTGAAATCTCGTGTGAAT
AAGAGTGATCCTATTTTGTGATGGATAGTTTCTTTTATAGTATTATTTACTTTTGGAGGTGTGACTGGTATTGTTTTATCTGCTT
GTGTATTGGATAAAGTTCTTCATGATACTTGATTT";

      }

      else if("Taenia parva".equals(s))

      {

        q=
"GGTTTGGGATTATAAGACATATATGTTTAAGAATTAGTATGTGTGATGATGCTTTTGGTTTTTATGGTTTGTTATTTGCTATGT
TTTCTATTGTGTGTTTAGGAAGAAGTGTATGAGGCCATCATATGTTTACTGTAGGTTTAGATGTGAAGACTGCTGTGTTTTTTA
GTTCAGTAACAATGATTATCGGGGTTCCTACTGGGATAAAGGTTTTTACTTGATTATATATGTTACTTAATTCTCGTATTAATA
AGGGTGATCCTGTAATTTGATGAATTGTTTCTTTCATAGTTTTATTTACGTTTGGTGGTGTCACTGGTATAGTTTTATCAGCTT
GTGTTTTAGATAAAGTTTTGCATGATACTTGATTT";

      }

      else if("Taenia pisiformis".equals(s))

      {

        q=
"GGGTTTGGTATAATTAGTCATATATGTTTAAGAATAAGTATGTGTTCAGATGCGTTTGGTTTTTATGGTTTATTGTTTGCAAT
GTTTTCTATAGTTTGTTTAGGTAGAAGTGTATGAGGTCATCATATGTTTACTGTTGGATTAGATGTAAAGACCGCTGTGTTTTT
TAGTTCAGTAACAATGATAATTGGAGTACCTACTGGAATTAAGGTCTTTACATGACTTTATATGCTTTTAAATTCTCGTGTCA
```

```
AAAAGAGTGATCCTGTGTTGTGGTGAATAATTTCTTTTATAGTCTTATTTACTTTTGGAGGTGTAACTGGTATAGTATTATCTG
CTTGTGTTTTAGATAAAGTT-TTACATGATACTTGATTT";

    }

    else if("Taenia saginata".equals(s))

    {

       q=
"GGTTTTGGTATGATTAGTCATATATGTTTAAGAATAAGTATGTGTCCAGATGCTTTTGGTTTTTATGGTTTGTTGTTTGCTATG
TTTTCAATAGTGTGTTTGGGGAGAAGTGTGTGGGGTCATCATATGTTTACGGTTGGGTTAGATGTTAAGACTGCTGTGTTTTTT
AGTTCGGTTACTATGATAATAGGAGTACCAACAGGAATAAAGGTTTTTACTTGACTTTATATGCTTTTAAATTCTCGTGTAAA
TAAGAGTGATCCTATATTGTGGTGAATAGTTTCTTTTATAGTGTTGTTTACTTTTGGTGGTGTGACTGGTATTGTGTTGTCTGC
TTGCGTATTGGATAAAGTTTTGCATGATACTTGATTT";

    }

    else if("Taenia serialis".equals(s))

    {

       q=
"GGTTTTGGTATAATTAGTCATATATGTTTAAGAATAAGTATGTGTCCAGACGCTTTTGGTTTTTATGGTTTGTTATTTGCTATG
TTCTCAATAGTGTGTTTAGGAAGGAGTGTATGGGGTCATCATATGTTTACAGTTGGGTTAGATGTTAAGACTGCTGTATTTTTT
AGCTCAGTTACTATGGTAATAGGAGTACCAACAGGAATAAAGGTTTTTACTTGGCTTTATATGTTATTAAATTCTCGTGTGAA
TAAGAGTGATCCTATATTGTGGTGGATAGTTTCTTTTATAGTATTGTTTACTTTTGGTGGTGTAACTGGGATTGTATTGTCTGC
TTGTGTGTTGGATAAAGTTTTACATGATACTTGGTTT";

    }

    else if("Taenia solium".equals(s))

    {

       q=
"GGGTTTGGTATAATTAGTCATATATGTTTGAGTATAAGTATGTGTTCTGATGCTTTTGGCTTTTATGGGTTATTGTTTGCTATG
TTTTCAATAGTATGTTTAGGAAGAAGTGTGTGAGGACATCATATGTTTACGGTTGGGTTAGATGTTAAGACGGCTGTATTTTT
TAGTTCTGTTACTATGATAATTGGAGTGCCTACGGGGATTAAGGTTTTTACTTGGCTTTATATGCTTTTAAAATCTCGTGTTAA
TAAGAGTGATCCGGTTTTATGATGAATAATTTCGTTTATAGTATTGTTTACATTTGGTGGTGTAACCGGTATTATTCTATCTGC
TTGTGTATTAGATAAAGTTCTTCATGATACTTGGTTT";

    }

    else if("Taenia twitchelli".equals(s))

    {

       q=
"GGTTTTGGTATAATTAGACATATTTGTTTAAATGTAAGTATGAATTATGATTCTTTTGGATTTTATGGTTTGTTATTTGCTATG
TTTTCTATAGTTTGTTTAGGTAGAAGTGTATGAGGTCATCATATGTTTACTGTTGGATTAGATGTTAAGACTGCTGTTTTTTTT
AGTTCTGTTACTATGATTATAGGGGTTCCTACAGGTATAAAGGTGTTTACTTGGTTATATATGCTTTTAAAATCTCGTGTAAAT
AAGAGTGATCCTGTTTTATGATGAATTGTGTCTTTTATTATTTTGTTTACTTTTGGTGGTGTTACTGGTATAGTGTTATCTGCGT
GTGTTTTGGATAAAGTTCTTCATGATACTTGGTTT";

    }

    else if("Taeniopygia guttata".equals(s))

    {

       q=
"GGTTTCGGCATCATCTCCCACGTCGTAACCTACTATTCAGGTAAAAAAGAACCATTCGGATATATAGGAATAGTATGAGCTA
TGCTATCCATCGGATTCCTAGGATTCATCGTATGAGCCCACCACATGTTTACAGTAGGAATGGACGTAGACACCCGAGCATA
CTTTACATCCGCCACTATAATCATCGCCATCCCAACCGGCATCAAAGTATTCAGCTGACTAGCAACACTCCACGGAGGCACA
ATCAAGTGAGACCCACCAATACTATGAGCTCTAGGATTTATCTTCCTATTCACCATCGGAGGCCTAACCGGAATCGTCCTGGC
CAACTCCTCACTAGACATCGCCCTACACGACACCTACTACGTAGTAGCCCACTTCCACTACGTCCTATCAATAGGAGCAGTGT
TTGCAATCCTAGCAGGATTCACCCACTGATT";
```

```
        }



double
AAA=0,AAC=0,AAG=0,AAT=0,ACA=0,ACC=0,ACG=0,ACT=0,AGA=0,AGC=0,AGG=0,AGT=0,ATA=0,ATC=0,ATG=0,ATT=0,CA
A=0,CAC=0,CAG=0,CAT=0,CCA=0,CCC=0,CCG=0,CCT=0,CGA=0,CGC=0,CGG=0,CGT=0,CTA=0,CTC=0,CTG=0,CTT=0,GAA=0,G
AC=0,GAG=0,GAT=0,GCA=0,GCC=0,GCG=0,GCT=0,GGA=0,GGC=0,GGG=0,GGT=0,GTA=0,GTC=0,GTG=0,GTT=0,TAA=0,TAC=
0,TAG=0,TAT=0,TCA=0,TCC=0,TCG=0,TCT=0,TGA=0,TGC=0,TGG=0,TGT=0,TTA=0,TTC=0,TTG=0,TTT=0;

double length = q.length();

for(int i=0;i<length-2;i++)

{

if(q.charAt(i)=='A')

{

        if(q.charAt(i+1)=='A')

        {

                if(q.charAt(i+2)=='A')

                AAA++;

                else if(q.charAt(i+2)=='C')

                AAC++;

                else if(q.charAt(i+2)=='G')

                AAG++;

                else if(q.charAt(i+2)=='T')

                AAT++;

        }

        else if(q.charAt(i+1)=='C')

        {

                if(q.charAt(i+2)=='A')

                ACA++;

                else if(q.charAt(i+2)=='C')

                ACC++;

                else if(q.charAt(i+2)=='G')

                ACG++;

                else if(q.charAt(i+2)=='T')

                ACT++;



        }

        else if(q.charAt(i+1)=='G')
```

```
          {
                    if(q.charAt(i+2)=='A')
                    AGA++;
                    else if(q.charAt(i+2)=='C')
                    AGC++;
                    else if(q.charAt(i+2)=='G')
                    AGG++;
                    else if(q.charAt(i+2)=='T')
                    AGT++;
          }
          else if(q.charAt(i+1)=='T')
          {
                    if(q.charAt(i+2)=='A')
                    ATA++;
                    else if(q.charAt(i+2)=='C')
                    ATC++;
                    else if(q.charAt(i+2)=='G')
                    ATG++;
                    else if(q.charAt(i+2)=='T')
                    ATT++;
          }
}
else if(q.charAt(i)=='C')
{
          if(q.charAt(i+1)=='A')
          {
                    if(q.charAt(i+2)=='A')
                    CAA++;
                    else if(q.charAt(i+2)=='C')
                    CAC++;
                    else if(q.charAt(i+2)=='G')
                    CAG++;
                    else if(q.charAt(i+2)=='T')
                    CAT++;
          }
```

```
        else if(q.charAt(i+1)=='C')
        {
                if(q.charAt(i+2)=='A')
                CCA++;
                else if(q.charAt(i+2)=='C')
                CCC++;
                else if(q.charAt(i+2)=='G')
                CCG++;
                else if(q.charAt(i+2)=='T')
                CCT++;
        }
        else if(q.charAt(i+1)=='G')
        {
                if(q.charAt(i+2)=='A')
                CGA++;
                else if(q.charAt(i+2)=='C')
                CGC++;
                else if(q.charAt(i+2)=='G')
                CGG++;
                else if(q.charAt(i+2)=='T')
                CGT++;
        }
        else if(q.charAt(i+1)=='T')
        {
                if(q.charAt(i+2)=='A')
                CTA++;
                else if(q.charAt(i+2)=='C')
                CTC++;
                else if(q.charAt(i+2)=='G')
                CTG++;
                else if(q.charAt(i+2)=='T')
                CTT++;
        }
}
else if(q.charAt(i)=='G')
```

```
{
        if(q.charAt(i+1)=='A')
        {
                if(q.charAt(i+2)=='A')
                GAA++;
                else if(q.charAt(i+2)=='C')
                GAC++;
                else if(q.charAt(i+2)=='G')
                GAG++;
                else if(q.charAt(i+2)=='T')
                GAT++;
        }
        else if(q.charAt(i+1)=='C')
        {
                if(q.charAt(i+2)=='A')
                GCA++;
                else if(q.charAt(i+2)=='C')
                GCC++;
                else if(q.charAt(i+2)=='G')
                GCG++;
                else if(q.charAt(i+2)=='T')
                GCT++;
        }
        else if(q.charAt(i+1)=='G')
        {
                if(q.charAt(i+2)=='A')
                GGA++;
                else if(q.charAt(i+2)=='C')
                GGC++;
                else if(q.charAt(i+2)=='G')
                GGG++;
                else if(q.charAt(i+2)=='T')
                GGT++;
        }
        else if(q.charAt(i+1)=='T')
```

```
                {

                            if(q.charAt(i+2)=='A')

                            GTA++;

                            else if(q.charAt(i+2)=='C')

                            GTC++;

                            else if(q.charAt(i+2)=='G')

                            GTG++;

                            else if(q.charAt(i+2)=='T')

                            GTT++;

                }

        }

        else if(q.charAt(i)=='T')

        {

            if(q.charAt(i+1)=='A')

            {

                            if(q.charAt(i+2)=='A')

                            TAA++;

                            else if(q.charAt(i+2)=='C')

                            TAC++;

                            else if(q.charAt(i+2)=='G')

                            TAG++;

                            else if(q.charAt(i+2)=='T')

                            TAT++;

                }

                else if(q.charAt(i+1)=='C')

                {

                            if(q.charAt(i+2)=='A')

                            TCA++;

                            else if(q.charAt(i+2)=='C')

                            TCC++;

                            else if(q.charAt(i+2)=='G')

                            TCG++;

                            else if(q.charAt(i+2)=='T')

                            TCT++;

                }
                else if(q.charAt(i+1)=='G')
                {
                        if(q.charAt(i+2)=='A')
                        TGA++;
                        else if(q.charAt(i+2)=='C')
                        TGC++;
                        else if(q.charAt(i+2)=='G')
                        TGG++;
                        else if(q.charAt(i+2)=='T')
                        TGT++;
                }
                else if(q.charAt(i+1)=='T')
                {
                        if(q.charAt(i+2)=='A')
                        TTA++;
                        else if(q.charAt(i+2)=='C')
                        TTC++;
                        else if(q.charAt(i+2)=='G')
                        TTG++;
                        else if(q.charAt(i+2)=='T')
                        TTT++;
                }
        }
      }
        AAA=AAA/length;
        AAC=AAC/length;
        AAG=AAG/length;
        AAT=AAT/length;
        ACA=ACA/length;
        ACC=ACC/length;
        ACG=ACG/length;
        ACT=ACT/length;
        AGA=AGA/length;
        AGC=AGC/length;
        AGG=AGG/length;
```

```
AGT=AGT/length;
ATA=ATA/length;
ATC=ATC/length;
ATG=ATG/length;
ATT=ATT/length;

CAA=CAA/length;
CAC=CAC/length;
CAG=CAG/length;
CAT=CAT/length;

CCA=CCA/length;
CCC=CCC/length;
CCG=CCG/length;
CCT=CCT/length;

CGA=CGA/length;
CGC=CGC/length;
CGG=CGG/length;
CGT=CGT/length;

CTA=CTA/length;
CTC=CTC/length;
CTG=CTG/length;
CTT=CTT/length;

GAA=GAA/length;
GAC=GAC/length;
GAG=GAG/length;
GAT=GAT/length;

GCA=GCA/length;
GCC=GCC/length;
GCG=GCG/length;
GCT=GCT/length;
```

```
GTA=GTA/length;

GTC=GTC/length;

GTG=GTG/length;

GTT=GTT/length;


GGA=GGA/length;

GGC=GGC/length;

GGG=GGG/length;

GGT=GGT/length;


TAA=TAA/length;

TAC=TAC/length;

TAG=TAG/length;

TAT=TAT/length;


TCA=TCA/length;

TCC=TCC/length;

TCG=TCG/length;

TCT=TCT/length;


TGA=TGA/length;

TGC=TGC/length;

TGG=TGG/length;

TGT=TGT/length;


TTA=TTA/length;

TTC=TTC/length;

TTG=TTG/length;

TTT=TTT/length;


AAA=AAA*10000;

AAA=Math.round(AAA);

AAA=AAA/10000;

ACA=ACA*10000;
```

```
ACA=Math.round(ACA);

ACA=ACA/10000;

AGA=AGA*10000;

AGA=Math.round(AGA);

AGA=AGA/10000;

ATA=ATA*10000;

ATA=Math.round(ATA);

ATA=ATA/10000;

CAA=CAA*10000;

CAA=Math.round(CAA);

CAA=CAA/10000;

CCA=CCA*10000;

CCA=Math.round(CCA);

CCA=CCA/10000;

CGA=CGA*10000;

CGA=Math.round(CGA);

CGA=CGA/10000;

CTA=CTA*10000;

CTA=Math.round(CTA);

CTA=CTA/10000;

GAA=GAA*10000;

GAA=Math.round(GAA);

GAA=GAA/10000;

GCA=GCA*10000;

GCA=Math.round(GCA);

GCA=GCA/10000;

GGA=GGA*10000;

GGA=Math.round(GGA);

GGA=GGA/10000;

GTA=GTA*10000;

GTA=Math.round(GTA);

GTA=GTA/10000;

TAA=TAA*10000;

TAA=Math.round(TAA);

TAA=TAA/10000;
```

```
TCA=TCA*10000;

TCA=Math.round(TCA);

TCA=TCA/10000;

TGA=TGA*10000;

TGA=Math.round(TGA);

TGA=TGA/10000;

TTA=TTA*10000;

TTA=Math.round(TTA);

TTA=TTA/10000;


AAC=AAC*10000;

AAC=Math.round(AAC);

AAC=AAC/10000;

ACC=ACC*10000;

ACC=Math.round(ACC);

ACC=ACC/10000;

AGC=AGC*10000;

AGC=Math.round(AGC);

AGC=AGC/10000;

ATC=ATC*10000;

ATC=Math.round(ATC);

ATC=ATC/10000;

CAC=CAC*10000;

CAC=Math.round(CAC);

CAC=CAC/10000;

CCC=CCC*10000;

CCC=Math.round(CCC);

CCC=CCC/10000;

CGC=CGC*10000;

CGC=Math.round(CGC);

CGC=CGC/10000;

CTC=CTC*10000;

CTC=Math.round(CTC);

CTC=CTC/10000;

GAC=GAC*10000;
```

```
GAC=Math.round(GAC);

GAC=GAC/10000;

GCC=GCC*10000;

GCC=Math.round(GCC);

GCC=GCC/10000;

GGC=GGC*10000;

GGC=Math.round(GGC);

GGC=GGC/10000;

GTC=GTC*10000;

GTC=Math.round(GTC);

GTC=GTC/10000;

TAC=TAC*10000;

TAC=Math.round(TAC);

TAC=TAC/10000;

TCC=TCC*10000;

TCC=Math.round(TCC);

TCC=TCC/10000;

TGC=TGC*10000;

TGC=Math.round(TGC);

TGC=TGC/10000;

TTC=TTC*10000;

TTC=Math.round(TTC);

TTC=TTC/10000;


AAG=AAG*10000;

AAG=Math.round(AAG);

AAG=AAG/10000;

ACG=ACG*10000;

ACG=Math.round(ACG);

ACG=ACG/10000;

AGG=AGG*10000;

AGG=Math.round(AGG);

AGG=AGG/10000;

ATG=ATG*10000;

ATG=Math.round(ATG);
```

```
ATG=ATG/10000;

CAG=CAG*10000;

CAG=Math.round(CAG);

CAG=CAG/10000;

CCG=CCG*10000;

CCG=Math.round(CCG);

CCG=CCG/10000;

CGG=CGG*10000;

CGG=Math.round(CGG);

CGG=CGG/10000;

CTG=CTG*10000;

CTG=Math.round(CTG);

CTG=CTG/10000;

GAG=GAG*10000;

GAG=Math.round(GAG);

GAG=GAG/10000;

GCG=GCG*10000;

GCG=Math.round(GCG);

GCG=GCG/10000;

GGG=GGG*10000;

GGG=Math.round(GGG);

GGG=GGG/10000;

GTG=GTG*10000;

GTG=Math.round(GTG);

GTG=GTG/10000;

TAG=TAG*10000;

TAG=Math.round(TAG);

TAG=TAG/10000;

TCG=TCG*10000;

TCG=Math.round(TCG);

TCG=TCG/10000;

TGG=TGG*10000;

TGG=Math.round(TGG);

TGG=TGG/10000;

TTG=TTG*10000;
```

```
TTG=Math.round(TTG);

TTG=TTG/10000;


AAT=AAT*10000;

AAT=Math.round(AAT);

AAT=AAT/10000;

ACT=ACT*10000;

ACT=Math.round(ACT);

ACT=ACT/10000;

AGT=AGT*10000;

AGT=Math.round(AGT);

AGT=AGT/10000;

ATT=ATT*10000;

ATT=Math.round(ATT);

ATT=ATT/10000;

CAT=CAT*10000;

CAT=Math.round(CAT);

CAT=CAT/10000;

CCT=CCT*10000;

CCT=Math.round(CCT);

CCT=CCT/10000;

CGT=CGT*10000;

CGT=Math.round(CGT);

CGT=CGT/10000;

CTT=CTT*10000;

CTT=Math.round(CTT);

CTT=CTT/10000;

GAT=GAT*10000;

GAT=Math.round(GAT);

GAT=GAT/10000;

GCT=GCT*10000;

GCT=Math.round(GCT);

GCT=GCT/10000;

GGT=GGT*10000;

GGT=Math.round(GGT);
```

```
GGT=GGT/10000;

GTT=GTT*10000;

GTT=Math.round(GTT);

GTT=GTT/10000;

TAT=TAT*10000;

TAT=Math.round(TAT);

TAT=TAT/10000;

TCT=TCT*10000;

TCT=Math.round(TCT);

TCT=TCT/10000;

TGT=TGT*10000;

TGT=Math.round(TGT);

TGT=TGT/10000;

TTT=TTT*10000;

TTT=Math.round(TTT);

TTT=TTT/10000;

            JFrame res= new JFrame("Trinucleotide frequency"+"   "+s);

   res.setSize(500,600);

   res.setLayout(null);


            JLabel f =new JLabel();

   f.setText(" AAA");

   f.setSize(50,50);

   f.setFont(f.getFont().deriveFont(14.0f));


            JLabel f1 =new JLabel();

            f1.setText(" AAC");

            f1.setBounds(0,30,50,50);

   f1.setSize(50,50);

   f1.setFont(f1.getFont().deriveFont(14.0f));


            JLabel f2 =new JLabel();

            f2.setText(" AAG");

   f2.setSize(50,50);

   f2.setFont(f1.getFont().deriveFont(14.0f));
```

```
        f2.setBounds(0,60,50,50);


        JLabel f15 =new JLabel();
f15.setText(" AAT");
f15.setSize(50,50);
f15.setFont(f15.getFont().deriveFont(14.0f));
f15.setBounds(0,90,50,50);


        JLabel f3 =new JLabel();
f3.setText(" ACA");
f3.setSize(50,50);
f3.setFont(f3.getFont().deriveFont(14.0f));
f3.setBounds(0,120,50,50);


        JLabel f4 =new JLabel();
f4.setText(" ACC");
f4.setSize(50,50);
f4.setFont(f4.getFont().deriveFont(14.0f));
f4.setBounds(0,150,50,50);


        JLabel f5 =new JLabel();
f5.setText(" ACG");
f5.setSize(50,50);
f5.setFont(f5.getFont().deriveFont(14.0f));
f5.setBounds(0,180,50,50);




        JLabel f6 =new JLabel();
f6.setText(" ACT");
f6.setSize(50,50);
f6.setFont(f6.getFont().deriveFont(14.0f));
f6.setBounds(0,210,50,50);


        JLabel f7 =new JLabel();
f7.setText(" AGA");
```

```
f7.setSize(50,50);
f7.setFont(f7.getFont().deriveFont(14.0f));
f7.setBounds(0,240,50,50);

        JLabel f8 =new JLabel();
f8.setText(" AGC");
f8.setSize(50,50);
f8.setFont(f8.getFont().deriveFont(14.0f));
f8.setBounds(0,270,50,50);

        JLabel f9 =new JLabel();
f9.setText(" AGG");
f9.setSize(50,50);
f9.setFont(f9.getFont().deriveFont(14.0f));
f9.setBounds(0,300,50,50);

        JLabel f10 =new JLabel();
f10.setText(" AGT");
f10.setSize(50,50);
f10.setFont(f10.getFont().deriveFont(14.0f));
f10.setBounds(0,330,50,50);

        JLabel f11 =new JLabel();
f11.setText(" ATA");
f11.setSize(50,50);
f11.setFont(f11.getFont().deriveFont(14.0f));
f11.setBounds(0,360,50,50);

        JLabel f12 =new JLabel();
f12.setText(" ATC");
f12.setSize(50,50);
f12.setFont(f12.getFont().deriveFont(14.0f));
f12.setBounds(0,390,50,50);

        JLabel f13 =new JLabel();
```

```
f13.setText(" ATG");

f13.setSize(50,50);

f13.setFont(f13.getFont().deriveFont(14.0f));

f13.setBounds(0,420,50,50);


        JLabel f14 =new JLabel();

f14.setText(" ATT");

f14.setSize(50,50);

f14.setFont(f14.getFont().deriveFont(14.0f));

f14.setBounds(0,450,50,50);


        JLabel g =new JLabel();

g.setText(" CAA");

g.setSize(50,50);

g.setFont(g.getFont().deriveFont(14.0f));

g.setBounds(100,0,50,50);


        JLabel g1 =new JLabel();

        g1.setText(" CAC");

        g1.setBounds(100,30,50,50);

g1.setSize(50,50);

g1.setFont(g1.getFont().deriveFont(14.0f));


        JLabel g2 =new JLabel();

        g2.setText(" CAG");

g2.setSize(50,50);

g2.setFont(g1.getFont().deriveFont(14.0f));

        g2.setBounds(100,60,50,50);


        JLabel g15 =new JLabel();

g15.setText(" CAT");

g15.setSize(50,50);

g15.setFont(g15.getFont().deriveFont(14.0f));

g15.setBounds(100,90,50,50);
```

```
        JLabel g3 =new JLabel();
g3.setText(" CCA");
g3.setSize(50,50);
g3.setFont(g3.getFont().deriveFont(14.0f));
g3.setBounds(100,120,50,50);

        JLabel g4 =new JLabel();
g4.setText(" CCC");
g4.setSize(50,50);
g4.setFont(g4.getFont().deriveFont(14.0f));
g4.setBounds(100,150,50,50);

        JLabel g5 =new JLabel();
g5.setText(" CCG");
g5.setSize(50,50);
g5.setFont(g5.getFont().deriveFont(14.0f));
g5.setBounds(100,180,50,50);



        JLabel g6 =new JLabel();
g6.setText(" CCT");
g6.setSize(50,50);
g6.setFont(g6.getFont().deriveFont(14.0f));
g6.setBounds(100,210,50,50);

        JLabel g7 =new JLabel();
g7.setText(" CGA");
g7.setSize(50,50);
g7.setFont(g7.getFont().deriveFont(14.0f));
g7.setBounds(100,240,50,50);

        JLabel g8 =new JLabel();
g8.setText(" CGC");
g8.setSize(50,50);
g8.setFont(g8.getFont().deriveFont(14.0f));
```

```
g8.setBounds(100,270,50,50);


        JLabel g9 =new JLabel();
g9.setText(" CGG");
g9.setSize(50,50);
g9.setFont(g9.getFont().deriveFont(14.0f));
g9.setBounds(100,300,50,50);


        JLabel g10 =new JLabel();
g10.setText(" CGT");
g10.setSize(50,50);
g10.setFont(g10.getFont().deriveFont(14.0f));
g10.setBounds(100,330,50,50);


        JLabel g11 =new JLabel();
g11.setText(" CTA");
g11.setSize(50,50);
g11.setFont(g11.getFont().deriveFont(14.0f));
g11.setBounds(100,360,50,50);


        JLabel g12 =new JLabel();
g12.setText(" CTC");
g12.setSize(50,50);
g12.setFont(g12.getFont().deriveFont(14.0f));
g12.setBounds(100,390,50,50);


        JLabel g13 =new JLabel();
g13.setText(" CTG");
g13.setSize(50,50);
g13.setFont(g13.getFont().deriveFont(14.0f));
g13.setBounds(100,420,50,50);


        JLabel g14 =new JLabel();
g14.setText(" CTT");
g14.setSize(50,50);
```

```
g14.setFont(g14.getFont().deriveFont(14.0f));
g14.setBounds(100,450,50,50);


        JLabel h =new JLabel();
h.setText(" GAA");
h.setSize(50,50);
h.setFont(h.getFont().deriveFont(14.0f));
        h.setBounds(200,0,50,50);
        JLabel h1 =new JLabel();
        h1.setText(" GAC");
        h1.setBounds(200,30,50,50);
h1.setSize(50,50);
h1.setFont(h1.getFont().deriveFont(14.0f));


        JLabel h2 =new JLabel();
        h2.setText(" GAG");
h2.setSize(50,50);
h2.setFont(h1.getFont().deriveFont(14.0f));
        h2.setBounds(200,60,50,50);


        JLabel h15 =new JLabel();
h15.setText(" GAT");
h15.setSize(50,50);
h15.setFont(h15.getFont().deriveFont(14.0f));
h15.setBounds(200,90,50,50);


        JLabel h3 =new JLabel();
h3.setText(" GCA");
h3.setSize(50,50);
h3.setFont(h3.getFont().deriveFont(14.0f));
h3.setBounds(200,120,50,50);


        JLabel h4 =new JLabel();
h4.setText(" GCC");
h4.setSize(50,50);
```

```
h4.setFont(h4.getFont().deriveFont(14.0f));
h4.setBounds(200,150,50,50);

        JLabel h5 =new JLabel();
h5.setText(" GCG");
h5.setSize(50,50);
h5.setFont(h5.getFont().deriveFont(14.0f));
h5.setBounds(200,180,50,50);



        JLabel h6 =new JLabel();
h6.setText(" GCT");
h6.setSize(50,50);
h6.setFont(h6.getFont().deriveFont(14.0f));
h6.setBounds(200,210,50,50);

        JLabel h7 =new JLabel();
h7.setText(" GGA");
h7.setSize(50,50);
h7.setFont(h7.getFont().deriveFont(14.0f));
h7.setBounds(200,240,50,50);

        JLabel h8 =new JLabel();
h8.setText(" GGC");
h8.setSize(50,50);
h8.setFont(h8.getFont().deriveFont(14.0f));
h8.setBounds(200,270,50,50);

        JLabel h9 =new JLabel();
h9.setText(" GGG");
h9.setSize(50,50);
h9.setFont(h9.getFont().deriveFont(14.0f));
h9.setBounds(200,300,50,50);

        JLabel h10 =new JLabel();
```

```
h10.setText(" GGT");
h10.setSize(50,50);
h10.setFont(h10.getFont().deriveFont(14.0f));
h10.setBounds(200,330,50,50);

        JLabel h11 =new JLabel();
h11.setText(" GTA");
h11.setSize(50,50);
h11.setFont(h11.getFont().deriveFont(14.0f));
h11.setBounds(200,360,50,50);

        JLabel h12 =new JLabel();
h12.setText(" GTC");
h12.setSize(50,50);
h12.setFont(h12.getFont().deriveFont(14.0f));
h12.setBounds(200,390,50,50);

        JLabel h13 =new JLabel();
h13.setText(" GTG");
h13.setSize(50,50);
h13.setFont(h13.getFont().deriveFont(14.0f));
h13.setBounds(200,420,50,50);

        JLabel h14 =new JLabel();
h14.setText(" GTT");
h14.setSize(50,50);
h14.setFont(h14.getFont().deriveFont(14.0f));
h14.setBounds(200,450,50,50);


        JLabel l =new JLabel();
l.setText(" TAA");
l.setSize(50,50);
l.setFont(l.getFont().deriveFont(14.0f));
        l.setBounds(300,0,50,50);
```

```
          JLabel l1 =new JLabel();
          l1.setText(" TAC");
          l1.setBounds(300,30,50,50);
l1.setSize(50,50);
l1.setFont(l1.getFont().deriveFont(14.0f));
          JLabel l2 =new JLabel();
          l2.setText(" TAG");
l2.setSize(50,50);
l2.setFont(l1.getFont().deriveFont(14.0f));
          l2.setBounds(300,60,50,50);
          JLabel l15 =new JLabel();
l15.setText(" TAT");
l15.setSize(50,50);
l15.setFont(l15.getFont().deriveFont(14.0f));
l15.setBounds(300,90,50,50);
          JLabel l3 =new JLabel();
l3.setText(" TCA");
l3.setSize(50,50);
l3.setFont(l3.getFont().deriveFont(14.0f));
l3.setBounds(300,120,50,50);
          JLabel l4 =new JLabel();
l4.setText(" TCC");
l4.setSize(50,50);
l4.setFont(l4.getFont().deriveFont(14.0f));
l4.setBounds(300,150,50,50);
          JLabel l5 =new JLabel();
l5.setText(" TCG");
l5.setSize(50,50);
l5.setFont(l5.getFont().deriveFont(14.0f));
l5.setBounds(300,180,50,50);
          JLabel l6 =new JLabel();
l6.setText(" TCT");
l6.setSize(50,50);
l6.setFont(l6.getFont().deriveFont(14.0f));
l6.setBounds(300,210,50,50);
```

```
        JLabel l7 =new JLabel();
l7.setText(" TGA");
l7.setSize(50,50);
l7.setFont(l7.getFont().deriveFont(14.0f));
l7.setBounds(300,240,50,50);
        JLabel l8 =new JLabel();
l8.setText(" TGC");
l8.setSize(50,50);
l8.setFont(l8.getFont().deriveFont(14.0f));
l8.setBounds(300,270,50,50);
        JLabel l9 =new JLabel();
l9.setText(" TGG");
l9.setSize(50,50);
l9.setFont(l9.getFont().deriveFont(14.0f));
l9.setBounds(300,300,50,50);
        JLabel l10 =new JLabel();
l10.setText(" TGT");
l10.setSize(50,50);
l10.setFont(l10.getFont().deriveFont(14.0f));
l10.setBounds(300,330,50,50);
        JLabel l11 =new JLabel();
l11.setText(" TTA");
l11.setSize(50,50);
l11.setFont(l11.getFont().deriveFont(14.0f));
l11.setBounds(300,360,50,50);
        JLabel l12 =new JLabel();
l12.setText(" TTC");
l12.setSize(50,50);
l12.setFont(l12.getFont().deriveFont(14.0f));
l12.setBounds(300,390,50,50);
        JLabel l13 =new JLabel();
l13.setText(" TTG");
l13.setSize(50,50);
l13.setFont(l13.getFont().deriveFont(14.0f));
l13.setBounds(300,420,50,50);
```

```
        JLabel l14 =new JLabel();
l14.setText(" TTT");
l14.setSize(50,50);
l14.setFont(l14.getFont().deriveFont(14.0f));
l14.setBounds(300,450,50,50);

String s1 = String.valueOf(AAA);
String s2 = String.valueOf(AAC);
String s3 = String.valueOf(AAG);
String s4 = String.valueOf(AAT);
String s5 = String.valueOf(ACA);
String s6 = String.valueOf(ACC);
String s7 = String.valueOf(ACG);
String s8 = String.valueOf(ACT);
String s9 = String.valueOf(AGA);
String s10 = String.valueOf(AGC);
String s11 = String.valueOf(AGG);
String s12 = String.valueOf(AGT);
String s13 = String.valueOf(ATA);
String s14 = String.valueOf(ATC);
String s15 = String.valueOf(ATG);
String s16 = String.valueOf(ATT);

        String s17 = String.valueOf(CAA);
String s18 = String.valueOf(CAC);
String s19 = String.valueOf(CAG);
String s20 = String.valueOf(CAT);
        String s21 = String.valueOf(CCA);
String s22 = String.valueOf(CCC);
String s23 = String.valueOf(CCG);
String s24 = String.valueOf(CCT);
String s25 = String.valueOf(CGA);
String s26 = String.valueOf(CGC);
String s27 = String.valueOf(CGG);
String s28 = String.valueOf(CGT);
String s29 = String.valueOf(CTA);
        String s30 = String.valueOf(CTC);
        String s31 = String.valueOf(CTG);
String s32 = String.valueOf(CTT);

        String s33 = String.valueOf(GAA);
String s34 = String.valueOf(GAC);
String s35 = String.valueOf(GAG);
String s36 = String.valueOf(GAT);
String s37 = String.valueOf(GCA);
String s38 = String.valueOf(GCC);
String s39 = String.valueOf(GCG);
String s40 = String.valueOf(GCT);
        String s41 = String.valueOf(GGA);
String s42 = String.valueOf(GGC);
String s43 = String.valueOf(GGG);
String s44 = String.valueOf(GGT);
String s45 = String.valueOf(GTA);
String s46 = String.valueOf(GTC);
String s47 = String.valueOf(GTG);
String s48 = String.valueOf(GTT);

        String s49 = String.valueOf(TAA);
String s50 = String.valueOf(TAC);
        String s51 = String.valueOf(TAG);
String s52 = String.valueOf(TAT);
String s53 = String.valueOf(TCA);
String s54 = String.valueOf(TCC);
String s55 = String.valueOf(TCG);
String s56 = String.valueOf(TCT);
String s57 = String.valueOf(TGA);
String s58 = String.valueOf(TGC);
String s59 = String.valueOf(TGG);
String s60 = String.valueOf(TGT);
String s61 = String.valueOf(TTA);
String s62 = String.valueOf(TTC);

String s63 = String.valueOf(TTG);

String s64 = String.valueOf(TTT);

JTextField A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13,A14,A15,A16,

                B1,B2,B3,B4,B5,B6,B7,B8,B9,B10,B11,B12,B13,B14,B15,B16,

                C1,C2,C3,C4,C5,C6,C7,C8,C9,C10,C11,C12,C13,C14,C15,C16,

    D1,D2,D3,D4,D5,D6,D7,D8,D9,D10,D11,D12,D13,D14,D15,D16;



A1 = new JTextField(s1);

A1.setBounds(40,15,50,25);

A1.setEditable(false);

res.add(A1);

A2 = new JTextField(s2);

A2.setBounds(40,45,50,25);

A2.setEditable(false);

res.add(A2);

A3 = new JTextField(s3);

A3.setBounds(40,75,50,25);

A3.setEditable(false);

res.add(A3);

A4 = new JTextField(s4);

A4.setBounds(40,105,50,25);

A4.setEditable(false);

res.add(A4);

A5 = new JTextField(s5);

A5.setBounds(40,135,50,25);

A5.setEditable(false);

res.add(A5);

A6 = new JTextField(s6);

A6.setBounds(40,165,50,25);

A6.setEditable(false);

res.add(A6);

A7 = new JTextField(s7);

A7.setBounds(40,195,50,25);
```

```
A7.setEditable(false);
res.add(A7);
A8 = new JTextField(s8);
A8.setBounds(40,225,50,25);
A8.setEditable(false);
res.add(A8);
A9 = new JTextField(s9);
A9.setBounds(40,255,50,25);
A9.setEditable(false);
res.add(A9);
A10 = new JTextField(s10);
A10.setBounds(40,285,50,25);
A10.setEditable(false);
res.add(A10);
A11 = new JTextField(s11);
A11.setBounds(40,315,50,25);
A11.setEditable(false);
res.add(A11);
A12 = new JTextField(s12);
A12.setBounds(40,345,50,25);
A12.setEditable(false);
res.add(A12);
A13 = new JTextField(s13);
A13.setBounds(40,375,50,25);
A13.setEditable(false);
res.add(A13);
A14 = new JTextField(s14);
A14.setBounds(40,405,50,25);
A14.setEditable(false);
res.add(A14);
A15 = new JTextField(s15);
A15.setBounds(40,435,50,25);
A15.setEditable(false);
res.add(A15);
A16 = new JTextField(s16);
```

```
A16.setBounds(40,465,50,25);

A16.setEditable(false);

res.add(A16);


        B1 = new JTextField(s17);

B1.setBounds(140,15,50,25);

B1.setEditable(false);

res.add(B1);

B2 = new JTextField(s18);

B2.setBounds(140,45,50,25);

B2.setEditable(false);

res.add(B2);

B3 = new JTextField(s19);

B3.setBounds(140,75,50,25);

B3.setEditable(false);

res.add(B3);

B4 = new JTextField(s20);

B4.setBounds(140,105,50,25);

B4.setEditable(false);

res.add(B4);

B5 = new JTextField(s21);

B5.setBounds(140,135,50,25);

B5.setEditable(false);

res.add(B5);

B6 = new JTextField(s22);

B6.setBounds(140,165,50,25);

B6.setEditable(false);

res.add(B6);

B7 = new JTextField(s23);

B7.setBounds(140,195,50,25);

B7.setEditable(false);

res.add(B7);

B8 = new JTextField(s24);

B8.setBounds(140,225,50,25);

B8.setEditable(false);
```

```
res.add(B8);

B9 = new JTextField(s25);

B9.setBounds(140,255,50,25);

B9.setEditable(false);

res.add(B9);

B10 = new JTextField(s26);

B10.setBounds(140,285,50,25);

B10.setEditable(false);

res.add(B10);

B11 = new JTextField(s27);

B11.setBounds(140,315,50,25);

B11.setEditable(false);

res.add(B11);

B12 = new JTextField(s28);

B12.setBounds(140,345,50,25);

B12.setEditable(false);

res.add(B12);

B13 = new JTextField(s29);

B13.setBounds(140,375,50,25);

B13.setEditable(false);

res.add(B13);

B14 = new JTextField(s50);

B14.setBounds(140,405,50,25);

B14.setEditable(false);

res.add(B14);

B15 = new JTextField(s31);

B15.setBounds(140,435,50,25);

B15.setEditable(false);

res.add(B15);

B16 = new JTextField(s32);

B16.setBounds(140,465,50,25);

B16.setEditable(false);

res.add(B16);


     C1 = new JTextField(s33);
```

```
C1.setBounds(240,15,50,25);

C1.setEditable(false);

res.add(C1);

C2 = new JTextField(s34);

C2.setBounds(240,45,50,25);

C2.setEditable(false);

res.add(C2);

C3 = new JTextField(s35);

C3.setBounds(240,75,50,25);

C3.setEditable(false);

res.add(C3);

C4 = new JTextField(s36);

C4.setBounds(240,105,50,25);

C4.setEditable(false);

res.add(C4);

C5 = new JTextField(s37);

C5.setBounds(240,135,50,25);

C5.setEditable(false);

res.add(C5);

C6 = new JTextField(s38);

C6.setBounds(240,165,50,25);

C6.setEditable(false);

res.add(C6);

C7 = new JTextField(s39);

C7.setBounds(240,195,50,25);

C7.setEditable(false);

res.add(C7);

C8 = new JTextField(s40);

C8.setBounds(240,225,50,25);

C8.setEditable(false);

res.add(C8);

C9 = new JTextField(s41);

C9.setBounds(240,255,50,25);

C9.setEditable(false);

res.add(C9);
```

```
C10 = new JTextField(s42);
C10.setBounds(240,285,50,25);
C10.setEditable(false);
res.add(C10);
C11 = new JTextField(s43);
C11.setBounds(240,315,50,25);
C11.setEditable(false);
res.add(C11);
C12 = new JTextField(s44);
C12.setBounds(240,345,50,25);
C12.setEditable(false);
res.add(C12);
C13 = new JTextField(s45);
C13.setBounds(240,375,50,25);
C13.setEditable(false);
res.add(C13);
C14 = new JTextField(s46);
C14.setBounds(240,405,50,25);
C14.setEditable(false);
res.add(C14);
C15 = new JTextField(s47);
C15.setBounds(240,435,50,25);
C15.setEditable(false);
res.add(C15);
C16 = new JTextField(s48);
C16.setBounds(240,465,50,25);
C16.setEditable(false);
res.add(C16);

        D1 = new JTextField(s49);
D1.setBounds(340,15,50,25);
D1.setEditable(false);
res.add(D1);
D2 = new JTextField(s50);
D2.setBounds(340,45,50,25);
```

```
D2.setEditable(false);

res.add(D2);

D3 = new JTextField(s51);

D3.setBounds(340,75,50,25);

D3.setEditable(false);

res.add(D3);

D4 = new JTextField(s52);

D4.setBounds(340,105,50,25);

D4.setEditable(false);

res.add(D4);

D5 = new JTextField(s53);

D5.setBounds(340,135,50,25);

D5.setEditable(false);

res.add(D5);

D6 = new JTextField(s54);

D6.setBounds(340,165,50,25);

D6.setEditable(false);

res.add(D6);

D7 = new JTextField(s55);

D7.setBounds(340,195,50,25);

D7.setEditable(false);

res.add(D7);

D8 = new JTextField(s56);

D8.setBounds(340,225,50,25);

D8.setEditable(false);

res.add(D8);

D9 = new JTextField(s57);

D9.setBounds(340,255,50,25);

D9.setEditable(false);

res.add(D9);

D10 = new JTextField(s58);

D10.setBounds(340,285,50,25);

D10.setEditable(false);

res.add(D10);

D11 = new JTextField(s59);
```

```
D11.setBounds(340,315,50,25);

D11.setEditable(false);

res.add(D11);

D12 = new JTextField(s60);

D12.setBounds(340,345,50,25);

D12.setEditable(false);

res.add(D12);

D13 = new JTextField(s61);

D13.setBounds(340,375,50,25);

D13.setEditable(false);

res.add(D13);

D14 = new JTextField(s62);

D14.setBounds(340,405,50,25);

D14.setEditable(false);

res.add(D14);

D15 = new JTextField(s63);

D15.setBounds(340,435,50,25);

D15.setEditable(false);

res.add(D15);

D16 = new JTextField(s64);

D16.setBounds(340,465,50,25);

D16.setEditable(false);

res.add(D16);

res.add(f);

res.add(f1);

res.add(f2);

res.add(f3);

res.add(f4);

res.add(f5);

res.add(f6);

res.add(f7);

res.add(f8);

res.add(f9);

res.add(f10);

res.add(f11);
res.add(f12);
res.add(f13);
res.add(f14);
res.add(f15);
        res.add(g);
res.add(g1);
res.add(g2);
res.add(g3);
res.add(g4);
res.add(g5);
res.add(g6);
res.add(g7);
res.add(g8);
res.add(g9);
res.add(g10);
res.add(g11);
res.add(g12);
res.add(g13);
res.add(g14);
res.add(g15);
        res.add(h);
res.add(h1);
res.add(h2);
res.add(h3);
res.add(h4);
res.add(h5);
res.add(h6);
res.add(h7);
res.add(h8);
res.add(h9);
res.add(h10);
res.add(h11);
res.add(h12);
res.add(h13);
res.add(h14);
```

```
        res.add(h15);

        res.add(l);

        res.add(l1);

        res.add(l2);

        res.add(l3);

        res.add(l4);

        res.add(l5);

        res.add(l6);

        res.add(l7);

        res.add(l8);

        res.add(l9);

        res.add(l10);

        res.add(l11);

        res.add(l12);

        res.add(l13);

        res.add(l14);

        res.add(l15);

        res.setLocationRelativeTo(null);

        res.setVisible(true);

    }


    private void jComboBox2ActionPerformed(java.awt.event.ActionEvent evt) {

        // TODO add your handling code here:

        t = (String) jComboBox2.getSelectedItem();

    }


    @SuppressWarnings("empty-statement")


private void jButton3ActionPerformed(java.awt.event.ActionEvent evt)

{




// TODO add your handling code here:



ammy diff;
```

```
diff = new ammy(s);

double [] x1 = diff.calcDifreq(s);

double [] x2 = diff.calcTrifreq(s);

double max1= x1[0];

double max2= x2[0];

double min1= x1[0];

double min2= x2[0];

for (int i=0;i<x1.length;i++)

{

        if(x1[i]>max1)

        max1=x1[i];

        else;

}

for (int i=0;i<x1.length;i++)

{

        if(x1[i]<min1)

        min1=x1[i];
```

```
                else;

        }

        for (int i=0;i<x2.length;i++)

        {

                if(x2[i]>max2)

                max2=x2[i];

                else;

        }

        for (int i=0;i<x2.length;i++)

        {

                if(x2[i]<min2)

                min2=x2[i];

                else;

        }

        double diffe1= max1-min1;

        double diffe2= max2-min2;

        diffe1=diffe1*10000;
            diffe1=Math.round(diffe1);
            diffe1=diffe1/10000;
            diffe2=diffe2*10000;

            diffe2=Math.round(diffe2);

            diffe2=diffe2/10000;

            JFrame res= new JFrame("Discrimination for "+s);

            res.setSize(420,200);

            res.setLayout(null);

            res.setLocationRelativeTo(null);

            JLabel f =new JLabel();

            f.setText(" Diff in Dinucleotide freq");

            f.setSize(300,50);

            f.setFont(f.getFont().deriveFont(14.0f));

            JLabel f1 =new JLabel();

            f1.setText(" Diff in Trinucleotide freq");

            //f1.setSize(300,50);

            f1.setBounds(0,30,300,50);

            f1.setFont(f1.getFont().deriveFont(14.0f));

            res.add(f);

            res.add(f1);


            String s1 = String.valueOf(diffe1);

            String s2 = String.valueOf(diffe2);

            JTextField A1,A2;

            A1 = new JTextField(s1);

            A1.setBounds(200,20,50,25);

            A1.setEditable(false);

            res.add(A1);

            A2 = new JTextField(s2);

            A2.setBounds(200,50,50,25);

            A2.setEditable(false);

            res.add(A2);

            res.setVisible(true);

        }



        private void jButton5ActionPerformed(java.awt.event.ActionEvent evt)
```

```
{
        // TODO add your handling code here:

ammy diff;

diff = new ammy(s);

double [] x1 = diff.calcDifreq(s);

double [] x2 = diff.calcTrifreq(s);

double [] m1= new double[x1.length];

double [] m2 =new double[x2.length];

double sum1 = 0;

double sum2 = 0;

double avg1;

double avg2;

for(int i=0;i<x1.length;i++)

{

                  sum1+=x1[i];

}

avg1=sum1/x1.length;

for(int i=0;i<x2.length;i++)
```

```
{
sum2+=x2[i];
}
avg2=sum2/x2.length;
for(int i=0;i<x1.length;i++)
{
m1[i]=x1[i]-avg1;
}
for(int i=0;i<x2.length;i++)
{
m2[i]=x2[i]-avg2;
}
for(int i=0;i<x1.length;i++)
{
m1[i]*=m1[i];
}
for(int i=0;i<x2.length;i++)
{
```

```
m2[i]*=m2[i];

}

double summate1=0;

double summate2=0;

for(int i=0;i<m1.length;i++)

{

summate1+=m1[i];

}

for(int i=0;i<m2.length;i++)

{

summate2+=m2[i];

}

double var1=summate1/m1.length;

double var2=summate2/m2.length;

var1=var1*10000;

var1=Math.round(var1);

var1=var1/10000;
```

```
var2=var2*10000;


var2=Math.round(var2);


var2=var2/10000;

System.out.println(var1+",");

    JFrame res= new JFrame("Variance for "+s);

    res.setSize(420,200);

    res.setLayout(null);

    res.setLocationRelativeTo(null);

    JLabel f =new JLabel();

    f.setText(" Variance for Dinucleotide freq");

    f.setSize(300,50);

    f.setFont(f.getFont().deriveFont(14.0f));

    JLabel f1 =new JLabel();

    f1.setText(" Variance for Trinucleotide freq");

    //f1.setSize(300,50);

    f1.setBounds(0,30,300,50);

    f1.setFont(f1.getFont().deriveFont(14.0f));

    res.add(f);

    res.add(f1);


    String s1 = String.valueOf(var1);

    String s2 = String.valueOf(var2);

    JTextField A1,A2;

    A1 = new JTextField(s1);

    A1.setBounds(200,20,50,25);

    A1.setEditable(false);

    res.add(A1);

    A2 = new JTextField(s2);

    A2.setBounds(200,50,50,25);

    A2.setEditable(false);

    res.add(A2);

    res.setVisible(true);
    }



private void jButton7ActionPerformed(java.awt.event.ActionEvent evt)
{
    // TODO add your handling code here:

ammy diff;

diff = new ammy(s,t);

double [] x1 = diff.calcDifreq(s);

double [] x2 = diff.calcTrifreq(s);

double [] y1 = diff.calcDifreq(t);

double [] y2 = diff.calcTrifreq(t);

for(int i=0;i<x1.length;i++)

{

                x1[i]=x1[i]-y1[i];

                x1[i]*=x1[i];

}



for(int i=0;i<x2.length;i++)

{

                x2[i]=x2[i]-y2[i];
```

```
        x2[i]*=x2[i];

}

double su1=0;

for(int i=0;i<x1.length;i++)

{

        su1+=x1[i];

}

double su2=0;

for(int i=0;i<x2.length;i++)

{

        su2+=x2[i];

}

su1=sqrt(su1);

su2=sqrt(su2);



su1=su1*10000;


su1=Math.round(su1);
    su1=su1/10000;
    su2=su2*10000;

    su2=Math.round(su2);

    su2=su2/10000;

    JFrame res= new JFrame("D for "+s+" "+t);

    res.setSize(420,200);

    res.setLayout(null);

    res.setLocationRelativeTo(null);

    JLabel f =new JLabel();

    f.setText(" D for di ");

    f.setSize(300,50);

    f.setFont(f.getFont().deriveFont(14.0f));

    JLabel f1 =new JLabel();

    f1.setText(" D for tri ");

    //f1.setSize(300,50);

    f1.setBounds(0,30,300,50);

    f1.setFont(f1.getFont().deriveFont(14.0f));

    res.add(f);

    res.add(f1);


    String s1 = String.valueOf(su1);

    String s2 = String.valueOf(su2);

    JTextField A1,A2;

    A1 = new JTextField(s1);

    A1.setBounds(200,20,50,25);

    A1.setEditable(false);

    res.add(A1);

    A2 = new JTextField(s2);

    A2.setBounds(200,50,50,25);

    A2.setEditable(false);

    res.add(A2);

    res.setVisible(true);


  }


  /**
```

```
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
     * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
     */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {
        java.util.logging.Logger.getLogger(aJFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (InstantiationException ex) {
        java.util.logging.Logger.getLogger(aJFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (IllegalAccessException ex) {
        java.util.logging.Logger.getLogger(aJFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {
        java.util.logging.Logger.getLogger(aJFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    }
    //</editor-fold>

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            aJFrame f=new aJFrame();
            f.setTitle("PRATIBHA");
            f.setVisible(true);
            f.setLocationRelativeTo(null);
            f.setBackground(Color.green);
        }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton2;
    private javax.swing.JButton jButton3;
    private javax.swing.JButton jButton4;
    private javax.swing.JButton jButton5;
    private javax.swing.JButton jButton6;
    private javax.swing.JButton jButton7;
    private javax.swing.JColorChooser jColorChooser1;
    private javax.swing.JComboBox jComboBox1;
    private javax.swing.JComboBox jComboBox2;
    private javax.swing.JFrame jFrame1;
    private javax.swing.JFrame jFrame2;
    private javax.swing.JLabel jLabel2;
    // End of variables declaration
}




class ammy {
    String a="",b="";
    ammy(String a)
    {
        this.a=a;
    }
    ammy(String a, String b)
    {
    this.a=a;
    this.b=b;
    }
    double[] calcDifreq(String s)
```

```
  {

    String q="";



    if("Angiostrongylus cantonensis".equals(s))

    {

        q=
"GCTTTTGGGATTGTTAGACAGTCTACTTTATATTTAACGGGTAAAAAAGAGGTTTTTGGTTATTTGGGTATGGTTTATGCTAT
TTTAAGAATTGGTTTGATTGGTTGTGTGGTTTGGGCTCATCATATATATACGGTTGGTATGGATTTAGATTCTCGTGCTTATTT
TACTGCGGCTACTATAGTTATTGCGGTTCCTACGGGAGTGAAGGTTTTTAGGTGGTTGGCAACTTTATTTGGTATAAAGATAT
TGTTTCAACCTATTTTATTGTGGGTTTTGGGTTTTATTTTTTTATTTACTATCGGTGGGTTAACGGGGGTTATATTGTCTAATTC
TAGTTTGGATATTATTTTACATGATACTTATTATGTAGTTAGGCATTTTCATTATGTTT";

    }

    else if("Angiostrongylus costaricensis".equals(s))

    {

       q=
"GCTTTTGGGATTATTAGTCAATCTGCTTTGTATTTGTCAGGGAAGAAAGAGGTTTTTGGTTATTTAGGGATGGTTTATGCGAT
TTTAAGAATTGGGTTGATTGGGTGTGTAGTTTGAGCTCATCATATGTATACTGTTGGTATGGATTTGGATTCTCGTGCTTACTT
TACTGCAGCTACAATAGTTATTGCGGTTCCTACTGGGGTTAAAGTGTTTAGTTGGTTGGCTACACTTTATGGGATGAAAATGA
TGTTTCAGCCGATTTTGTTGTGGGTTATGGGGTTTATTTTTTTGTTTACTATTGGGGGTTTGACCGGGGTTATGTTATCTAATTC
AAGTTTGGATATTATTTTGCATGATACTTATTATGTGGTT";

    }

    else if("Angiostrongylus vasorum".equals(s))

    {

       q=
"GCTTTTGGGATTGTTAGTCAGTCGACTTTATATTTGACTGGGAAGAAGGAGGTGTTTGGTTATTTGGGGATGGTTTATGCGAT
TTTAAGGATTGGTTTGATTGGTTGTGTGGTGTGAGCTCATCATATGTATACTGTTGGTATAGATTTAGATTCTCGTGCTTATTT
TACTGCGGCTACTATGGTGATTGCGGTGCCGACTGGAGTGAAGGTTTTTAGTTGGTTGGCTACTTTGTTTGGGATGAAGTATA
GTATTTCAGCCTATTTTGTTGTGGGTTATAGGATTTATTTTTTTTTATTTACTATTGGGGGTTTGACGGGTGTGATATTGTCAAA
TTCGAGATTGGATATTATTTTACATGATACGTATTATGTGGTAAGTCATTTTCATTATGTGAGGTTGTTTCATGATACTTGGTT
T";

    }

    else if("Diplogonoporus balaenopterae".equals(s))

    {

       q=
"GGGTTTGGAATGATTAGACATGTTTGTAGTAACTTAGGTTGTTCATATGATACTTTTGGATTTTATGGTTTATTATTTGCTATG
TTTTCTATCGTTTGTCTTGGTAGGGTTGTATGGGGTCATCATATGTTTACGGTAGGTTTAGATGTGAAGACAGCTGTTTTTTTT
AGGTCAGTTACTATGATTATAGGGGTGCCTACTGGAATAAAGGTGTTTTCTTGGCTGTATATGATTTTAAAAAGTCGTGTCTC
TTTGCGTGAACCTATATTTTGGTGGGTGTTATCTTTTATCGTGCTGTTTACAATAGGGGGTGTTACTGGTATTATACTTTCTGC
TTGTGTTCTTGATAATATTTTGCATGATACTTGATTT";

    }

    else if("Diplogonoporus grandis".equals(s))

    {

       q=
"GGGTTTGGAATGATTAGACATGTTTGTAGTAACTTAGGTTGTTCATATGATACTTTTGGATTTTATGGTTTATTATTTGCTATG
TTTTCTATCGTTTGTCTTGGTAGGGTTGTATGGGGTCATCATATGTTTACGGTAGGTTTAGATGTGAAGACAGCTGTTTTTTTT
AGGTCAGTTACTATGATTATAGGGGTGCCTACTGGAATAAAGGTGTTTTCTTGGCTGTATATGATTTTAAAAAGTCGTGTCTC
TTTGCGTGAACCTATATTTTGGTGGGTGTTATCTTTTATCGTGCTGTTTACAATAGGGGGTGTTACTGGTATTATACTTTCTGC
TTGTGTTCTTGATAATATTTTGCATGATACTTGATTT";
```

```
    }

    else if("Aelurostrongylus abstrusus".equals(s))

    {

        q=
"GCTTTTGGTATTGTTAGTCAGTCTACTTTGTATTTGACGGGGAAGAAGGAAGTTTTTGGTTATTTAGGGATAGTTTATGCTAT
TATAAGTATTGGTTTAATTGGTTGTGTTGTTTGGGCTCATCATATATATACTGTTGGTATAGATTTGGATTCTCGTGCTTATTTT
ACGGCGGCTACGATGGTTATTGCTGTGCCAACGGGTGTTAAGGTTTTTAGTTGGTTGGCTACTTTATTTGGAATGAAGATAGT
GTTTCAGCCGGTTTTGTTGTGGGTTTTGGGTTTTATTTTTTTGTTTACTATTGGGGGGTTAACTGGGGTCATGCTTTCGAATTCT
AGTTTGGATATTATTTTGCATGATACTTATTATGTGGTTAGTCATTTTCATTATGTGTTGAGTTT";

    }

    else if("Dictyocaulus eckerti".equals(s))

    {

        q=
"GCTTTTGGTATTATTAGTCAGTCAACTTTGTATTTAACTGGTAAAAAGGAGGTTTTTGGTTCTTTGGGTATGGTTTATGCTATT
TTAAGTATTGGTTTAATTGGTTGTGTAGTATGAGCACATCATATATATACTGTTGGAATAGATTTGGATTCGCGTGCTTATTTT
ACTGCTGCTACTATGGTAATTGCTGTTCCTACGGGTGTAAAAGTTTTTAGTTGGTTGGCTACTTTGTATGGTTTAAAAATAGTA
TATAATCCTTTGTTGTTATGGGTTTTGGGTTTTATTTTTTTATTTACTATTGGGGGGTTAACTGGAGTTATTTTGTCAAATTCTA
GTTTAGATATTTTGTTACATGATACTTATTATGTTGTAAGGCATTT";

    }

    else if("Dictyocaulus viviparus".equals(s))

    {

        q=
"GCTTTTGGTATTATTAGACAATCTACTTTGTATTTAACTGGTAAAAAAGAGGTTTTTGGTTCTTTGGGTATGGTTTATGCTATT
TTAAGTATTGGTTTAATTGGTTGTGTTGTGTGGGCACATCATATGTATACTGTTGGGATGGATTTGGATTCGCGTGCTTATTTT
ACTGCTGCTACTATAGTAATTGCTGTTCCTACTGGAGTTAAGGTTTTTAGATGATTGGCTACTTTATATGGATTGAAAATGGTT
TATAATCCTTTGTTGTTGTGAGTTTTAGGTTTTATTTTTTTGTTTACTATTGGTGGTTTAACTGGTGTTATTTTGTCAAATTCTA
GTCTTGATATTTTGTTGCATGATACTTATTAT";

    }

    else if("Ascaris lumbricoides".equals(s))

    {

        q=
"GCTTTTGGTATTATTAGTCAGAGTAGTTTGTATTTGACTGGTAAAAAGGAGGTTTTTGGGTCTTTGGGTATGGTTTATGCTAT
TTTAAGTATTGGTTTGATTGGTTGTGTTGTTTGAGCTCATCATATGTATACTGTTGGTATGGATCTTGATTCTCGGGCTTATTTT
ACTGCTGCAACTATGGTTATTGCTGTTCCTACTGGTGTTAAGGTTTTTAGTTGGTTGGCTACCTTGTTTGGTATAAAAATGGTT
TTTCAGCCTTTACTTTTATGAGTTATGGGTTTTATTTTTTTGTTTACTATTGGTGGGTTAACCGGGGTTATACTTTCTAATTCTA
GTTTGGATATTATCTTGCATGATACTTATTATGTTGTTAGTCATTTTCATTATGTTCTTAGTTT";

    }

    else if("Ascaris suum".equals(s))

    {

        q=
"GCTTTTGGTATTATTAGTCAGAGTAGTTTGTATCTGACTGGTAAAAAGGAGGTTTTTGGGTCTTTGGGTATGGTTTATGCTAT
TTTGAGTATTGGTTTAATTGGTTGTGTTGTTTGAGCTCATCATATGTATACTGTTGGTATGGATCTTGACTCTCGGGCTTATTTT
ACTGCTGCAACTATGGTTATTGCTGTTCCTACTGGTGTTAAGGTTTTTAGTTGGTTGGCTACCTTGTTTGGTATAAAGATGGTT
TTTCAACCTTTACTTTTATGAGTTATGGGTTTTATTTTTTTGTTTACTATTGGTGGGTTAACCGGGGTTATGCTTTCTAATTCTA
GTTTGGATATTATCTTGCATGATACTTA-TTATGTTGTTAGTCATTTTCATTATGTCCTTAGTTT";

    }

    else if("Baylisascaris ailuri".equals(s))
```

```
        {

            q=
"GCTTTTGGTATTATTAGCCAGAGTAGGTTGTATTTAACTGGTAAAAAGGAAGTTTTTGGTTCTTTGGGTATGGTTTATGCTAT
TTTGAGTATTGGTTTAATTGGTTGTGTTGTTTGGGCTCATCATATGTATACTGTAGGTATGGATTTGGATTCTCGTGCTTATTTT
ACTGCGGCTACTATGGTTATTGCGGTCCCTACGGGTGTTAAGGTTTTTAGTTGGTTGGCTACTTTGTTTGGTATGAAGATGGTG
TTCCAGCCTTTACTTTTGTGAGTTATGGGTTTTATTTTTTTATTTACTATTGGCGGATTGACTGGGGTGATGCTTTCTAATTCTA
GTTTGGATATTATTTTGCATGATACTTA-TTATGTTGTTAGACATTTTCATTATGTTCTTAGTTT";

        }

        else if("Baylisascaris procyonis".equals(s))

        {

            q=
"GCTTTTGGTATTATTAGCCAAAGTAGGTTGTATTTAACTGGTAAGAAGGAAGTTTTTGGTTCTTTGGGAATGGTTTATGCTAT
TTTGAGTATTGGTTTGATTGGATGTGTGGTTTGGGCTCATCATATGTATACTGTGGGTATGGATTTGGATTCTCGGGCTTATTT
TACTGCGGCTACTATGGTTATTGCGGTTCCTACGGGAGTTAAGGTTTTTAGTTGGTTGGCCACTTTATTTGGTATGAAGATAGT
GTTTCAGCCTTTGCTTTTGTGGGTTATGGGGTTTATTTTTTTGTTTACTATTGGTGGTTTGACTGGGGTTATGCTTTCTAATTCT
AGTTTGGATATTATTTTGCATGATACTTATTATGTTGTTAGGCATTTTCATTATGTTCTTAGTTT";

        }

        else if("Baylisascaris schroederi".equals(s))

        {

            q=
"GCTTTTGGTATTATTAGTCAGAGTAGGTTGTATCTGACTGGTAAGAAGGAGGTTTTTGGTTCTTTGGGTATGGTTTACGCAAT
TTTGAGTATTGGTTTGATTGGTTGTGTTGTTTGAGCTCATCATATGTATACTGTGGGTATAGATTTGGATTCTCGTGCTTATTTT
ACTGCGGCTACTATGGTTATCGCAGTTCCTACGGGTGTTAAGGTTTTTAGTTGGTTGGCTACTTTGTTTGGTATGAAGATGGTG
TTTCAGCCTTTGCTTTTGTGGGTTATAGGATTTATTTTTTTGTTTACTATTGGTGGATTGACTGGGGTGATGCTTTCTAATTCTA
GTTTGGATATTATTTTGCATGATACTTA-TTATGTTGTTAGGCATTTTCATTATGTTCTTAGTTT";

        }

        else if("Baylisascaris transfuga".equals(s))

        {

            q=
"GCTTTTGGTATTATTAGTCAGAGTAGATTGTATTTAACTGGTAAAAAGGAGGTTTTTGGTTCTTTGGGTATGGTTTATGCTAT
TTTGAGTATTGGTTTAATTGGTTGTGTTGTTTGGGCTCATCATATGTATACTGTAGGTATGGATTTGGATTCTCGTGCTTATTTT
ACTGCGGCTACTATGGTTATTGCGGTTCCTACAGGTGTTAAGGTTTTTAGTTGGTTGGCCACTTTGTTTGGTATGAAGATGGTG
TTTCAGCCTTTACTTTTGTGGGTTATGGGGTTTATTTTTTTATTTACTATTGGTGGATTGACTGGGGTGATGCTTTCTAATTCTA
GTTTGGATATTATTTTGCATGATACTTATTATGTTGTTAGGCATTTTCATTATGTTCTTAGTTT";

        }

        else if("Toxocara canis".equals(s))

        {

            q=
"GCTTTTGGTATTATTAGGCAAAGTAGTTTGTATTTGACTGGTAAAAAGGAGGTTTTTGGTTCTTTAGGCATGGTTTATGCTAT
TTTAAGTATTGGTCTGATTGGCTGTGTAGTTTGGGCTCACCATATGTATACGGTGGGCATGGATTTGGATTCTCGTGCTTATTT
TACTGCGGCAACGATGGTTATTGCTGTGCCTACGGGGGTTAAGGTTTTTAGTTGGTTAGCCACTCTTTTTGGTATGAAGATGG
TGTTTCAACCTTTGCTTTTGTGGGTGCTGGGTTTTATTTTTTTATTTACTATCGGGGGGTTGACTGGTGTTATGTTATCTAATTC
TAGGTTGGACATTATCTTGCATGATACTTATTATGTTGTTAGTCATTTTCATTATGTTT";

        }

        else if("Toxocara cati".equals(s))

        {

            q=
"GATTTTTTGGGCATCCTGAGGTTTATATTTTGATTTTACCTGCCTTTGGTATTATTAGTCAAAGTAGTTTATATTTAACTGGTA
```

```
AGAAGGAGGTTTTTGGTTCTTTGGGCATGGTCTATGCTATTTTGAGTATTGGTTTGATTGGTTGTGTGGTGTGAGCTCACCACA
TGTATACTGTTGGTATAGACTTGGATTCTCGGGCTTATTTACTGCGGCTACTATGGTTATCGCTGTGCCTACGGGTGTTAAGG
TTTTTAGTTGGTTGGCTACTCTTTTTGGTATAAAAATGGTTTTTCAACCTTTGCTTTTGTGAGTGTTGGGTTTTATTTTTTTGTTT
ACTATTGGTGGGCTTACTGGAGTTATGCTTTCTAATTCTAGTTTGGATATTATTTTGCATGACACCTATTATGTTGTGAGGCAT
TTCCACTATGTTT";

        }else if("Toxocara malaysiensis".equals(s))

        {

          q=
"GCTTTTGGTATTATTAGTCAGAGTAGTTTGTATTTAACTGGTAAGAAGGAAGTTTTTGGTTCGTTGGGGATGGTTTATGCTAT
TTTAAGTATTGGTTTGATTGGCTGTGTGGTTTGGGCTCATCATATGTATACCGTGGGTATAGATTTGGATTCTCGGGCTTATTT
TACTGCGGCGACTATGGTTATTGCTGTGCCTACTGGTGTTAAGGTTTTTAGTTGGTTGGCTACTCTTTTTGGTATGAAAATGGT
TTTTCAGCCTTTACTTTTATGGGTGTTAGGTTTTATTTTCTTGTTTACTATTGGGGGCCTTACTGGTGTGATGCTTTCTAATTCT
AGCCTTGATATTATTTTGCATGATACCTATTATGTTGTTAGACATTTTCATTATGTTT";

        }

        else if("Taenia asiatica".equals(s))

        {

          q=
"GGTTTTGGTATGATTAGTCATATATGTTTAAGAATAAGTATGTGTCCGGATGCTTTTGGTTTTTATGGTTTGTTATTTGCTATG
TTTTCAATAGTATGTTTGGGGAGAAGTGTGTGGGGTCATGATATGTTTACGGTTGGATTAGTTGTTAAGACTACTGTGTTTTTT
AGTTCGGTTACTATGATAATAGGAGTACCAACAGGAATAAAGGTTTTTACTTGACTTTATATGCTTTTAAATTCTCGTGTAAA
TAAGAGGGATCCTATATTGTGGTGGATAGTTTCTTTTATAGTGTTGTTTACCTTTGGTGGTGTGACTGGTATTGTGTTGTCTGC
TTGTGTATTGGATAAAGTTTTGCATGATACTTGATTT";

        }

        else if("Taenia crassiceps".equals(s))

        {

          q=
"GGTTTTGGAATTATTAGACATATTTGTTTGAAAATAAGTATGAATTGTGATTCTTTTGGTTTTTATGGATTGTTATTTGCTATG
TTTTCAATAGTTTGTTTAGGTAGGAGTGTTTGGGGTCATCATATGTTTACGGTTGGTTTAGATGTTAAGACTGCTGTTTTTTTT
AGTTCTGTTACTATGATTATAGGAGTACCTACAGGTATAAAGGTGTTTACTTGATTGTATATGCTTTTAAATTCGCGTGTGAA
CAAGAGTGATCCTATATTGTGGTGAATTGTTTCTTTTATAGTTTTATTTACGTTTGGTGGTGTTACTGGAATAGTATTGTCTGC
TTGTGTATTAGATAAAGTTCTTCATGATACTTGATTT";

        }

        else if("Taenia hydatigena".equals(s))

        {

          q=
"GGATTTGGAATTATTAGTCATATATGTTTGAGAATAAGTATGAGTCCTGATGCTTTTGGGTTCTATGGATTATTATTTGCTAT
GTTTTCAATAGTCTGTTTGGGTAGAAGTGTGTGGGGTCATCATATGTTTACTGTTGGGTTAGATGTTAAGACTGCTGTTTTTTT
TAGTTCTGTGACTATGATTATAGGTGTGCCTACTGGTATAAAGGTGTTTACTTGGTTATATATGCTTTTAAACTCTCATGTGAA
TAAGAGTGATCCTGTTGTTTGATGAATTGTTTCTTTTATAGTTTTGTTTACTTTTGGTGGGGTTACTGGTATTGTGTTGTCAGCA
TGTGTATTAGATAAAGTTCTTCATGATACCTGATTT";

        }

        else if("Taenia krepkogorski".equals(s))

        {

          q=
"GGATTTGGTATAATTAGTCATATATGTTTAAGAATAAGTATGTCTTCTGATGTGTTTGGGTTTTATGGTTTATTATTTGCTATG
TTTTCTATAGTTTGTTTAGGAAGAAGAGTGTGAGGTCATCACATGTTTACTGTTGGTTTAGATGTTAAGACTGCTGTGTTTTTT
AGCTCAATTACTATGATTATTGGTGTGCCTACTGGGATTAAGGTTTTTACATGATTATATATGTTATTAAATGCTCGAGTAAA
AAAGAGTGATCCTGTGTTGTGGTGAATTGTTTCATTTATAATATTGTTTACATTTGGTGGAGTTACTGGTATAGTATTGTCTGC
TTGTGTTTTAGATAAAGTGTTACATGATACTTGGTTT";
```

```
        }

        else if("Taenia laticollis".equals(s))

        {

           q=
"GGATTTGGTATAATTAGACATATATGTTTAAGTATTAGTATGTGTTCGGATGCTTTCGGTTTTTATGGTTTATTATTTGCTATG
TTTTCTATTGTTTGTTTAGGGAGAAGAGTTTGGGGTCATCATATGTTTACAGTTGGGTTAGATGTTAAGACGGCTGTATTTTTT
AGTTCTGTAACTATGATTATTGGTGTACCTACAGGTATAAAGGTTTTTACATGATTATATATGCTTTTAAATTCTCGGGTTAAA
AAGAGTGATCCTGTATTATGGTGGATAGTTTCTTTTATAGTTTTGTTTACGTTTGGTGGTGTTACAGGAATAGTGTTGTCTGCT
TGCGTATTAGATAAAGTATTACATGATACTTGATTT";

        }

        else if("Taenia madoquae".equals(s))

        {

           q=
"GGTTTTGGGATAATTAGTCATATATGTTTGAGGATTAGTATGTGTCCTGATGCTTTTGGTTTTTATGGTTTGTTATTTGCTATG
TTCTCAATAGTGTGTTTGGGAAGGAGTGTATGAGGTCATCACATGTTTACGGTTGGATTAGATGTTAAGACTGCTGTATTTTT
TAGTTCGGTTACTATGATAATAGGAGTACCTACAGGAATAAAGGTTTTTACTTGACTTTATATGCTTTTAAATTCTCGTGTGA
ATAAGAGTGATCCTGTGTTATGATGGATAGTTTCTTTTATAGTATTGTTTACTTTTGGTGGTGTAACTGGTATTGTATTATCTG
CTTGTGTATTGGATAATGTTTTACATGATACTTGATTT";

        }

        else if("Taenia martis".equals(s))

        {

           q=
"GGTTTTGGTATAATTAGACATATTTGTCTAAATATAAGTATGAATTATGATTCTTTTGGTTTTTATGGTTTATTATTTGCTATG
TTTTCTATAGTTTGTTTGGGTAGTAGTGTGTGGGGTCATCATATGTTTACTGTTGGATTAGATGTTAAGACTGCTGTTTTTTTC
AGTTCAGTTACTATGATAATAGGTGTTCCTACGGGTATAAAGGTTTTTACTTGATTATATATGCTTTTAAAATCTCGTGTGAAT
AAGAGTGATCCTGTATTATGGTGAATTGTTTCTTTTATTATTTTGTTTACTTTTGGTGGTGTTACTGGTATAGTGTTATCTGCAT
GTGTTTTGGATAAAGTTCTTCATGATACTTGATTT";

        }

        else if("Taenia multiceps".equals(s))

        {

           q=
"GGTTTTGGTATAATTAGTCACATATGTTTAAGAATAAGCATGTGTCCAGATGCTTTTGGTTTTTATGGTTTATTATTTGCTATG
TTTTCAATAGTGTGTTTAGGGAGAAGTGTGTGAGGCCATCATATGTTTACAGTTGGGTTAGATGTTAAGACTGCTGTATTTTTT
AGTTCGGTTACTATGATAATAGGAGTGCCCACAGGAATAAAGGTTTTTACTTGGCTTTATATGCTTTTAAATTCTCGTGTAAA
CAAGAGTGATCCTATACTATGATGAATAGTTTCTTTTATAGTATTGTTTACTTTTGGTGGTGTAACTGGGATTGTATTGTCTGC
TTGTGTATTAGATAAAGTTTTACATGATACTTGATTT";

        }

        else if("Taenia mustelae".equals(s))

        {

           q=
"GGTTTTGGTATTATTGGTCATATATGTTTGAGTATAAGGATGTGTTCTGATGCTTTTGGGTTTTATGGATTGTTGTTTGCTATG
TTTTCTATTGTTTGTCTAGGTAGTAGAGTTTGAGGGCATCATATGTTTACTGTTGGTTTAGATGTTAAGACTGCTGTTTTTTTTA
GTTCTGTTACTATGATTATAGGAGTTCCTACTGGTATAAAGGTGTTTACTTGGTTGTATATGTTACTGAATTCTAGTGTTAACA
AGAGGGATCCTGTGTTGTGATGAATAGTGTCATTTATATTTTTGTTTACTTTTGGTGGTGTTACTGGTATAGTTTTGTCTGCTT
GTGTATTAGATAATGTTTTGCATGATACTTGATTT";

        }

        else if("Taenia ovis".equals(s))
```

```
    {

        q=
"GGATTTGGTATAATTAGTCATATTTGTTTGAGGATTAGTATGTGTCCAGATGCTTTTGGTTTTTATGGCTTATTATTTGCTATG
TTTTCTATAGTATGTTTAGGAAGAAGTGTGTGGGGGCATCATATGTTTACTGTTGGGTTGGATGTTAAGACGGCTGTATTTTTT
AGTTCGGTTACTATGATCATAGGTGTGCCTACTGGTATAAAGGTTTTTACTTGGCTTTATATGCTTCTGAAATCTCGTGTGAAT
AAGAGTGATCCTATTTTGTGATGGATAGTTTCTTTTATAGTATTATTTACTTTTGGAGGTGTGACTGGTATTGTTTTATCTGCTT
GTGTATTGGATAAAGTTCTTCATGATACTTGATTT";

    }

    else if("Taenia parva".equals(s))

    {

        q=
"GGTTTGGGATTATAAGACATATATGTTTAAGAATTAGTATGTGTGATGATGCTTTTGGTTTTTATGGTTTGTTATTTGCTATGT
TTTCTATTGTGTGTTTAGGAAGAAGTGTATGAGGCCATCATATGTTTACTGTAGGTTTAGATGTGAAGACTGCTGTGTTTTTTA
GTTCAGTAACAATGATTATCGGGGTTCCTACTGGGATAAAGGTTTTTACTTGATTATATATGTTACTTAATTCTCGTATTAATA
AGGGTGATCCTGTAATTTGATGAATTGTTTCTTTCATAGTTTTATTTACGTTTGGTGGTGTCACTGGTATAGTTTTATCAGCTT
GTGTTTTAGATAAAGTTTTGCATGATACTTGATTT";

    }

    else if("Taenia pisiformis".equals(s))

    {

        q=
"GGGTTTGGTATAATTAGTCATATATGTTTAAGAATAAGTATGTGTTCAGATGCGTTTGGTTTTTATGGTTTATTGTTTGCAAT
GTTTTCTATAGTTTGTTTAGGTAGAAGTGTATGAGGTCATCATATGTTTACTGTTGGATTAGATGTAAAGACCGCTGTGTTTTT
TAGTTCAGTAACAATGATAATTGGAGTACCTACTGGAATTAAGGTCTTTACATGACTTTATATGCTTTTAAATTCTCGTGTCA
AAAAGAGTGATCCTGTGTTGTGGTGAATAATTTCTTTTATAGTCTTATTTACTTTTGGAGGTGTAACTGGTATAGTATTATCTG
CTTGTGTTTTAGATAAAGTT-TTACATGATACTTGATTT";

    }

    else if("Taenia saginata".equals(s))

    {

        q=
"GGTTTTGGTATGATTAGTCATATATGTTTAAGAATAAGTATGTGTCCAGATGCTTTTGGTTTTTATGGTTTGTTGTTTGCTATG
TTTTCAATAGTGTGTTTGGGGAGAAGTGTGTGGGGTCATCATATGTTTACGGTTGGGTTAGATGTTAAGACTGCTGTGTTTTTT
AGTTCGGTTACTATGATAATAGGAGTACCAACAGGAATAAAGGTTTTTACTTGACTTTATATGCTTTTAAATTCTCGTGTAAA
TAAGAGTGATCCTATATTGTGGTGAATAGTTTCTTTTATAGTGTTGTTTACTTTTGGTGGTGTGACTGGTATTGTGTTGTCTGC
TTGCGTATTGGATAAAGTTTTGCATGATACTTGATTT";

    }

    else if("Taenia serialis".equals(s))

    {

        q=
"GGTTTTGGTATAATTAGTCATATATGTTTAAGAATAAGTATGTGTCCAGACGCTTTTGGTTTTTATGGTTTGTTATTTGCTATG
TTCTCAATAGTGTGTTTAGGAAGGAGTGTATGGGGTCATCATATGTTTACAGTTGGGTTAGATGTTAAGACTGCTGTATTTTTT
AGCTCAGTTACTATGGTAATAGGAGTACCAACAGGAATAAAGGTTTTTACTTGGCTTTATATGTTATTAAATTCTCGTGTGAA
TAAGAGTGATCCTATATTGTGGTGGATAGTTTCTTTTATAGTATTGTTTACTTTTGGTGGTGTAACTGGGATTGTATTGTCTGC
TTGTGTGTTGGATAAAGTTTTACATGATACTTGGTTT";

    }

    else if("Taenia solium".equals(s))

    {

        q=
"GGGTTTGGTATAATTAGTCATATATGTTTGAGTATAAGTATGTGTTCTGATGCTTTTGGCTTTTATGGGTTATTGTTTGCTATG
```

```
TTTTCAATAGTATGTTTAGGAAGAAGTGTGTGAGGACATCATATGTTTACGGTTGGGTTAGATGTTAAGACGGCTGTATTTTT
TAGTTCTGTTACTATGATAATTGGAGTGCCTACGGGGATTAAGGTTTTTACTTGGCTTTATATGCTTTTAAAATCTCGTGTTAA
TAAGAGTGATCCGGTTTTATGATGAATAATTTCGTTTATAGTATTGTTTACATTTGGTGGTGTAACCGGTATTATTCTATCTGC
TTGTGTATTAGATAAAGTTCTTCATGATACTTGGTTT";

        }

        else if("Taenia twitchelli".equals(s))

        {

            q=
"GGTTTTGGTATAATTAGACATATTTGTTTAAATGTAAGTATGAATTATGATTCTTTTGGATTTTATGGTTTGTTATTTGCTATG
TTTTCTATAGTTTGTTTAGGTAGAAGTGTATGAGGTCATCATATGTTTACTGTTGGATTAGATGTTAAGACTGCTGTTTTTTTT
AGTTCTGTTACTATGATTATAGGGGTTCCTACAGGTATAAAGGTGTTTACTTGGTTATATATGCTTTTAAAATCTCGTGTAAAT
AAGAGTGATCCTGTTTTATGATGAATTGTGTCTTTTATTATTTTGTTTACTTTTGGTGGTGTTACTGGTATAGTGTTATCTGCGT
GTGTTTTGGATAAAGTTCTTCATGATACTTGGTTT";

        }

        else if("Taeniopygia guttata".equals(s))

        {

            q=
"GGTTTCGGCATCATCTCCCACGTCGTAACCTACTATTCAGGTAAAAAAGAACCATTCGGATATATAGGAATAGTATGAGCTA
TGCTATCCATCGGATTCCTAGGATTCATCGTATGAGCCCACCACATGTTTACAGTAGGAATGGACGTAGACACCCGAGCATA
CTTTACATCCGCCACTATAATCATCGCCATCCCAACCGGCATCAAAGTATTCAGCTGACTAGCAACACTCCACGGAGGCACA
ATCAAGTGAGACCCACCAATACTATGAGCTCTAGGATTTATCTTCCTATTCACCATCGGAGGCCTAACCGGAATCGTCCTGGC
CAACTCCTCACTAGACATCGCCCTACACGACACCTACTACGTAGTAGCCCACTTCCACTACGTCCTATCAATAGGAGCAGTGT
TTGCAATCCTAGCAGGATTCACCCACTGATT";

        }




     double AA=0,AC=0,AG=0,AT=0,CA=0,CC=0,CG=0,CT=0,GA=0,GC=0,GG=0,GT=0,TA=0,TC=0,TG=0,TT=0;

     int length = q.length();

     for(int i=0;i<length-1;i++)

     {

        if(q.charAt(i)=='A')

        {

            if(q.charAt(i+1)=='A')

            {

                        AA++;

            }

            else if(q.charAt(i+1)=='C')

            {

                        AC++;

            }

            else if(q.charAt(i+1)=='G')
```

```
   {
                  AG++;
   }
   else if(q.charAt(i+1)=='T')
   {
                  AT++;
   }
}
else if(q.charAt(i)=='C')
{
      if(q.charAt(i+1)=='A')
      {
         CA++;
      }
      else if(q.charAt(i+1)=='C')
      {
         CC++;
      }
      else if(q.charAt(i+1)=='G')
      {
         CG++;
      }
      else if(q.charAt(i+1)=='T')
      {
         CT++;
      }
}
else if(q.charAt(i)=='G')
{
      if(q.charAt(i+1)=='A')
      {
         GA++;
      }
      else if(q.charAt(i+1)=='C')
      {
```

```
                GC++;
            }
            else if(q.charAt(i+1)=='G')
            {
                GG++;
            }
            else if(q.charAt(i+1)=='T')
            {
                GT++;
            }
    }
    else if(q.charAt(i)=='T')
    {
            if(q.charAt(i+1)=='A')
            {
                TA++;
            }
            else if(q.charAt(i+1)=='C')
            {
                TC++;
            }
              else if(q.charAt(i+1)=='G')
            {
                TG++;
            }
            else if(q.charAt(i+1)=='T')
            {
                TT++;
            }
    }
}
AA=AA/length;
AC=AC/length;
AG=AG/length;
AT=AT/length;
```

```
CA=CA/length;

CC=CC/length;

CG=CG/length;

CT=CT/length;

GA=GA/length;

GC=GC/length;

GT=GT/length;

GG=GG/length;

TA=TA/length;

TC=TC/length;

TG=TG/length;

TT=TT/length;


AA=AA*10000;

AA=Math.round(AA);

AA=AA/10000;

AC=AC*10000;

AC=Math.round(AC);

AC=AC/10000;

AG=AG*10000;

AG=Math.round(AG);

AG=AG/10000;

AT=AT*10000;

AT=Math.round(AT);

AT=AT/10000;

CA=CA*10000;

CA=Math.round(CA);

CA=CA/10000;

CC=CC*10000;

CC=Math.round(CC);

CC=CC/10000;

CG=CG*10000;

CG=Math.round(CG);

CG=CG/10000;

CT=CT*10000;
```

```
CT=Math.round(CT);
CT=CT/10000;
GA=GA*10000;
GA=Math.round(GA);
GA=GA/10000;
GC=GC*10000;
GC=Math.round(GC);
GC=GC/10000;
GG=GG*10000;
GG=Math.round(GG);
GG=GG/10000;
GT=GT*10000;
GT=Math.round(GT);
GT=GT/10000;
TA=TA*10000;
TA=Math.round(TA);
TA=TA/10000;
TC=TC*10000;
TC=Math.round(TC);
TC=TC/10000;
TG=TG*10000;
TG=Math.round(TG);
TG=TG/10000;
TT=TT*10000;
TT=Math.round(TT);
TT=TT/10000;

double[] c={AA,AC,AG,AT,CA,CC,CG,CT,GA,GC,GG,GT,TA,TC,TG,TT};


   return c;
}
double[] calcTrifreq(String s)


{
String q="";
```

```
        if("Angiostrongylus cantonensis".equals(s))

        {

            q=
"GCTTTTGGGATTGTTAGACAGTCTACTTTATATTTAACGGGTAAAAAAGAGGTTTTTGGTTATTTGGGTATGGTTTATGCTAT
TTTAAGAATTGGTTTGATTGGTTGTGTGGTTTGGGCTCATCATATATATACGGTTGGTATGGATTTAGATTCTCGTGCTTATTT
TACTGCGGCTACTATAGTTATTGCGGTTCCTACGGGAGTGAAGGTTTTTAGGTGGTTGGCAACTTTATTTGGTATAAAGATAT
TGTTTCAACCTATTTTATTGTGGGTTTTGGGTTTTATTTTTTTATTTACTATCGGTGGGTTAACGGGGGTTATATTGTCTAATTC
TAGTTTGGATATTATTTTACATGATACTTATTATGTAGTTAGGCATTTTCATTATGTTT";

        }

        else if("Angiostrongylus costaricensis".equals(s))

        {

            q=
"GCTTTTGGGATTATTAGTCAATCTGCTTTGTATTTGTCAGGGAAGAAAGAGGTTTTTGGTTATTTAGGGATGGTTTATGCGAT
TTTAAGAATTGGGTTGATTGGGTGTGTAGTTTGAGCTCATCATATGTATACTGTTGGTATGGATTTGGATTCTCGTGCTTACTT
TACTGCAGCTACAATAGTTATTGCGGTTCCTACTGGGGTTAAAGTGTTTAGTTGGTTGGCTACACTTTATGGGATGAAAATGA
TGTTTCAGCCGATTTTGTTGTGGGTTATGGGGTTTATTTTTTTGTTTACTATTGGGGGTTTGACCGGGGTTATGTTATCTAATTC
AAGTTTGGATATTATTTTGCATGATACTTATTATGTGGTT";

        }

        else if("Angiostrongylus vasorum".equals(s))

        {

            q=
"GCTTTTGGGATTGTTAGTCAGTCGACTTTATATTTGACTGGGAAGAAGGAGGTGTTTGGTTATTTGGGGATGGTTTATGCGAT
TTTAAGGATTGGTTTGATTGGTTGTGTGGTGTGAGCTCATCATATGTATACTGTTGGTATAGATTTAGATTCTCGTGCTTATTT
TACTGCGGCTACTATGGTGATTGCGGTGCCGACTGGAGTGAAGGTTTTTAGTTGGTTGGCTACTTTGTTTGGGATGAAGTATA
GTATTTCAGCCTATTTTGTTGTGGGTTATAGGATTTATTTTTTTTATTTACTATTGGGGGTTTGACGGGTGTGATATTGTCAAA
TTCGAGATTGGATATTATTTTACATGATACGTATTATGTGGTAAGTCATTTTCATTATGTGAGGTTGTTTCATGATACTTGGTT
T";

        }

        else if("Diplogonoporus balaenopterae".equals(s))

        {

            q=
"GGGTTTGGAATGATTAGACATGTTTGTAGTAACTTAGGTTGTTCATATGATACTTTTGGATTTTATGGTTTATTATTTGCTATG
TTTTCTATCGTTTGTCTTGGTAGGGTTGTATGGGGTCATCATATGTTTACGGTAGGTTTAGATGTGAAGACAGCTGTTTTTTTT
AGGTCAGTTACTATGATTATAGGGGTGCCTACTGGAATAAAGGTGTTTTCTTGGCTGTATATGATTTTAAAAAGTCGTGTCTC
TTTGCGTGAACCTATATTTTGGTGGGTGTTATCTTTTATCGTGCTGTTTACAATAGGGGGTGTTACTGGTATTATACTTTCTGC
TTGTGTTCTTGATAATATTTTGCATGATACTTGATTT";

        }

        else if("Diplogonoporus grandis".equals(s))

        {

            q=
"GGGTTTGGAATGATTAGACATGTTTGTAGTAACTTAGGTTGTTCATATGATACTTTTGGATTTTATGGTTTATTATTTGCTATG
TTTTCTATCGTTTGTCTTGGTAGGGTTGTATGGGGTCATCATATGTTTACGGTAGGTTTAGATGTGAAGACAGCTGTTTTTTTT
AGGTCAGTTACTATGATTATAGGGGTGCCTACTGGAATAAAGGTGTTTTCTTGGCTGTATATGATTTTAAAAAGTCGTGTCTC
TTTGCGTGAACCTATATTTTGGTGGGTGTTATCTTTTATCGTGCTGTTTACAATAGGGGGTGTTACTGGTATTATACTTTCTGC
TTGTGTTCTTGATAATATTTTGCATGATACTTGATTT";

        }

        else if("Aelurostrongylus abstrusus".equals(s))
        {

            q=
"GCTTTTGGTATTGTTAGTCAGTCTACTTTGTATTTGACGGGGAAGAAGGAAGTTTTTGGTTATTTAGGGATAGTTTATGCTAT
TATAAGTATTGGTTTAATTGGTTGTGTTGTTTGGGCTCATCATATATATACTGTTGGTATAGATTTGGATTCTCGTGCTTATTTT
ACGGCGGCTACGATGGTTATTGCTGTGCCAACGGGTGTTAAGGTTTTTAGTTGGTTGGCTACTTTATTTGGAATGAAGATAGT
GTTTCAGCCGGTTTTGTTGTGGGTTTTGGGTTTTATTTTTTTGTTTACTATTGGGGGGTTAACTGGGGTCATGCTTTCGAATTCT
AGTTTGGATATTATTTTGCATGATACTTATTATGTGGTTAGTCATTTTCATTATGTGTTGAGTTT";

        }

        else if("Dictyocaulus eckerti".equals(s))

        {

            q=
"GCTTTTGGTATTATTAGTCAGTCAACTTTGTATTTAACTGGTAAAAAGGAGGTTTTTGGTTCTTTGGGTATGGTTTATGCTATT
TTAAGTATTGGTTTAATTGGTTGTGTAGTATGAGCACATCATATATATACTGTTGGAATAGATTTGGATTCGCGTGCTTATTTT
ACTGCTGCTACTATGGTAATTGCTGTTCCTACGGGTGTAAAAGTTTTTAGTTGGTTGGCTACTTTGTATGGTTTAAAAATAGTA
TATAATCCTTTGTTGTTATGGGTTTTGGGTTTTATTTTTTTATTTACTATTGGGGGGTTAACTGGAGTTATTTTGTCAAATTCTA
GTTTAGATATTTTGTTACATGATACTTATTATGTTGTAAGGCATTT";

        }

        else if("Dictyocaulus viviparus".equals(s))

        {

            q=
"GCTTTTGGTATTATTAGACAATCTACTTTGTATTTAACTGGTAAAAAAGAGGTTTTTGGTTCTTTGGGTATGGTTTATGCTATT
TTAAGTATTGGTTTAATTGGTTGTGTTGTGTGGGCACATCATATGTATACTGTTGGGATGGATTTGGATTCGCGTGCTTATTTT
ACTGCTGCTACTATAGTAATTGCTGTTCCTACTGGAGTTAAGGTTTTTAGATGATTGGCTACTTTATATGGATTGAAAATGGTT
TATAATCCTTTGTTGTTGTGAGTTTTAGGTTTTATTTTTTTGTTTACTATTGGTGGTTTAACTGGTGTTATTTTGTCAAATTCTA
GTCTTGATATTTTGTTGCATGATACTTATTAT";

        }

        else if("Ascaris lumbricoides".equals(s))

        {

            q=
"GCTTTTGGTATTATTAGTCAGAGTAGTTTGTATTTGACTGGTAAAAAGGAGGTTTTTGGGTCTTTGGGTATGGTTTATGCTAT
TTTAAGTATTGGTTTGATTGGTTGTGTTGTTTGAGCTCATCATATGTATACTGTTGGTATGGATCTTGATTCTCGGGCTTATTTT
ACTGCTGCAACTATGGTTATTGCTGTTCCTACTGGTGTTAAGGTTTTTAGTTGGTTGGCTACCTTGTTTGGTATAAAAATGGTT
TTTCAGCCTTTACTTTTATGAGTTATGGGTTTTATTTTTTTGTTTACTATTGGTGGGTTAACCGGGGTTATACTTTCTAATTCTA
GTTTGGATATTATCTTGCATGATACTTATTATGTTGTTAGTCATTTTCATTATGTTCTTAGTTT";

        }

        else if("Ascaris suum".equals(s))

        {

            q=
"GCTTTTGGTATTATTAGTCAGAGTAGTTTGTATCTGACTGGTAAAAAGGAGGTTTTTGGGTCTTTGGGTATGGTTTATGCTAT
TTTGAGTATTGGTTTAATTGGTTGTGTTGTTTGAGCTCATCATATGTATACTGTTGGTATGGATCTTGACTCTCGGGCTTATTTT
ACTGCTGCAACTATGGTTATTGCTGTTCCTACTGGTGTTAAGGTTTTTAGTTGGTTGGCTACCTTGTTTGGTATAAAGATGGTT
TTTCAACCTTTACTTTTATGAGTTATGGGTTTTATTTTTTTGTTTACTATTGGTGGGTTAACCGGGGTTATGCTTTCTAATTCTA
GTTTGGATATTATCTTGCATGATACTTA-TTATGTTGTTAGTCATTTTCATTATGTCCTTAGTTT";

        }

        else if("Baylisascaris ailuri".equals(s))

        {

            q=
"GCTTTTGGTATTATTAGCCAGAGTAGGTTGTATTTAACTGGTAAAAAGGAAGTTTTTGGTTCTTTGGGTATGGTTTATGCTAT
```

```
TTTGAGTATTGGTTTAATTGGTTGTGTTGTTTGGGCTCATCATATGTATACTGTAGGTATGGATTTGGATTCTCGTGCTTATTTT
ACTGCGGCTACTATGGTTATTGCGGTCCCTACGGGTGTTAAGGTTTTTAGTTGGTTGGCTACTTTGTTTGGTATGAAGATGGTG
TTCCAGCCTTTACTTTTGTGAGTTATGGGTTTTATTTTTTTATTTACTATTGGCGGATTGACTGGGGTGATGCTTTCTAATTCTA
GTTTGGATATTATTTTGCATGATACTTA-TTATGTTGTTAGACATTTTCATTATGTTCTTAGTTT";

        }

        else if("Baylisascaris procyonis".equals(s))

        {

           q=
"GCTTTTGGTATTATTAGCCAAAGTAGGTTGTATTTAACTGGTAAGAAGGAAGTTTTTGGTTCTTTGGGAATGGTTTATGCTAT
TTTGAGTATTGGTTTGATTGGATGTGTGGTTTGGGCTCATCATATGTATACTGTGGGTATGGATTTGGATTCTCGGGCTTATTT
TACTGCGGCTACTATGGTTATTGCGGTTCCTACGGGAGTTAAGGTTTTTAGTTGGTTGGCCACTTTATTTGGTATGAAGATAGT
GTTTCAGCCTTTGCTTTTGTGGGTTATGGGGTTTATTTTTTTGTTTACTATTGGTGGTTTGACTGGGGTTATGCTTTCTAATTCT
AGTTTGGATATTATTTTGCATGATACTTATTATGTTGTTAGGCATTTTCATTATGTTCTTAGTTT";

        }

        else if("Baylisascaris schroederi".equals(s))

        {

           q=
"GCTTTTGGTATTATTAGTCAGAGTAGGTTGTATCTGACTGGTAAGAAGGAGGTTTTTGGTTCTTTGGGTATGGTTTACGCAAT
TTTGAGTATTGGTTTGATTGGTTGTGTTGTTTGAGCTCATCATATGTATACTGTGGGTATAGATTTGGATTCTCGTGCTTATTTT
ACTGCGGCTACTATGGTTATCGCAGTTCCTACGGGTGTTAAGGTTTTTAGTTGGTTGGCTACTTTGTTTGGTATGAAGATGGTG
TTTCAGCCTTTGCTTTTGTGGGTTATAGGATTTATTTTTTTGTTTACTATTGGTGGATTGACTGGGGTGATGCTTTCTAATTCTA
GTTTGGATATTATTTTGCATGATACTTA-TTATGTTGTTAGGCATTTTCATTATGTTCTTAGTTT";

        }

        else if("Baylisascaris transfuga".equals(s))

        {

           q=
"GCTTTTGGTATTATTAGTCAGAGTAGATTGTATTTAACTGGTAAAAAGGAGGTTTTTGGTTCTTTGGGTATGGTTTATGCTAT
TTTGAGTATTGGTTTAATTGGTTGTGTTGTTTGGGCTCATCATATGTATACTGTAGGTATGGATTTGGATTCTCGTGCTTATTTT
ACTGCGGCTACTATGGTTATTGCGGTTCCTACAGGTGTTAAGGTTTTTAGTTGGTTGGCCACTTTGTTTGGTATGAAGATGGTG
TTTCAGCCTTTACTTTTGTGGGTTATGGGGTTTATTTTTTTATTTACTATTGGTGGATTGACTGGGGTGATGCTTTCTAATTCTA
GTTTGGATATTATTTTGCATGATACTTATTATGTTGTTAGGCATTTTCATTATGTTCTTAGTTT";

        }

        else if("Toxocara canis".equals(s))

        {

           q=
"GCTTTTGGTATTATTAGGCAAAGTAGTTTGTATTTGACTGGTAAAAAGGAGGTTTTTGGTTCTTTAGGCATGGTTTATGCTAT
TTTAAGTATTGGTCTGATTGGCTGTGTAGTTTGGGCTCACCATATGTATACGGTGGGCATGGATTTGGATTCTCGTGCTTATTT
TACTGCGGCAACGATGGTTATTGCTGTGCCTACGGGGGTTAAGGTTTTTAGTTGGTTAGCCACTCTTTTTGGTATGAAGATGG
TGTTTCAACCTTTGCTTTTGTGGGTGCTGGGTTTTATTTTTTTATTTACTATCGGGGGGTTGACTGGTGTTATGTTATCTAATTC
TAGGTTGGACATTATCTTGCATGATACTTATTATGTTGTTAGTCATTTTCATTATGTTT";

        }

        else if("Toxocara cati".equals(s))

        {

           q=
"GATTTTTTGGGCATCCTGAGGTTTATATTTTGATTTTACCTGCCTTTGGTATTATTAGTCAAAGTAGTTTATATTTAACTGGTA
AGAAGGAGGTTTTTGGTTCTTTGGGCATGGTCTATGCTATTTTGAGTATTGGTTTGATTGGTTGTGTGGTGTGAGCTCACCACA
TGTATACTGTTGGTATAGACTTGGATTCTCGGGCTTATTTTACTGCGGCTACTATGGTTATCGCTGTGCCTACGGGTGTTAAGG
TTTTTAGTTGGTTGGCTACTCTTTTTGGTATAAAAATGGTTTTTCAACCTTTGCTTTTGTGAGTGTTGGGTTTTATTTTTTTGTTT
```

```
ACTATTGGTGGGCTTACTGGAGTTATGCTTTCTAATTCTAGTTTGGATATTATTTTGCATGACACCTATTATGTTGTGAGGCAT
TTCCACTATGTTT";

    }else if("Toxocara malaysiensis".equals(s))

    {

        q=
"GCTTTTGGTATTATTAGTCAGAGTAGTTTGTATTTAACTGGTAAGAAGGAAGTTTTTGGTTCGTTGGGGATGGTTTATGCTAT
TTTAAGTATTGGTTTGATTGGCTGTGTGGTTTGGGCTCATCATATGTATACCGTGGGTATAGATTTGGATTCTCGGGCTTATTT
TACTGCGGCGACTATGGTTATTGCTGTGCCTACTGGTGTTAAGGTTTTTAGTTGGTTGGCTACTCTTTTTGGTATGAAAATGGT
TTTTCAGCCTTTACTTTTATGGGTGTTAGGTTTTATTTTCTTGTTTACTATTGGGGGCCTTACTGGTGTGATGCTTTCTAATTCT
AGCCTTGATATTATTTTGCATGATACCTATTATGTTGTTAGACATTTTCATTATGTTT";

    }

    else if("Taenia asiatica".equals(s))

    {

        q=
"GGTTTTGGTATGATTAGTCATATATGTTTAAGAATAAGTATGTGTCCGGATGCTTTTGGTTTTTATGGTTTGTTATTTGCTATG
TTTTCAATAGTATGTTTGGGGAGAAGTGTGTGGGGTCATGATATGTTTACGGTTGGATTAGTTGTTAAGACTACTGTGTTTTTT
AGTTCGGTTACTATGATAATAGGAGTACCAACAGGAATAAAGGTTTTTACTTGACTTTATATGCTTTTAAATTCTCGTGTAAA
TAAGAGGGATCCTATATTGTGGTGGATAGTTTCTTTTATAGTGTTGTTTACCTTTGGTGGTGTGACTGGTATTGTGTTGTCTGC
TTGTGTATTGGATAAAGTTTTGCATGATACTTGATTT";

    }

    else if("Taenia crassiceps".equals(s))

    {

        q=
"GGTTTTGGAATTATTAGACATATTTGTTTGAAAATAAGTATGAATTGTGATTCTTTTGGTTTTTATGGATTGTTATTTGCTATG
TTTTCAATAGTTTGTTTAGGTAGGAGTGTTTGGGGTCATCATATGTTTACGGTTGGTTTAGATGTTAAGACTGCTGTTTTTTTT
AGTTCTGTTACTATGATTATAGGAGTACCTACAGGTATAAAGGTGTTTACTTGATTGTATATGCTTTTAAATTCGCGTGTGAA
CAAGAGTGATCCTATATTGTGGTGAATTGTTTCTTTTATAGTTTTATTTACGTTTGGTGGTGTTACTGGAATAGTATTGTCTGC
TTGTGTATTAGATAAAGTTCTTCATGATACTTGATTT";

    }

    else if("Taenia hydatigena".equals(s))

    {

        q=
"GGATTTGGAATTATTAGTCATATATGTTTGAGAATAAGTATGAGTCCTGATGCTTTTGGGTTCTATGGATTATTATTTGCTAT
GTTTTCAATAGTCTGTTTGGGTAGAAGTGTGTGGGGTCATCATATGTTTACTGTTGGGTTAGATGTTAAGACTGCTGTTTTTTT
TAGTTCTGTGACTATGATTATAGGTGTGCCTACTGGTATAAAGGTGTTTACTTGGTTATATATGCTTTTAAACTCTCATGTGAA
TAAGAGTGATCCTGTTGTTTGATGAATTGTTTCTTTTATAGTTTTGTTTACTTTTGGTGGGGTTACTGGTATTGTGTTGTCAGCA
TGTGTATTAGATAAAGTTCTTCATGATACCTGATTT";

    }

    else if("Taenia krepkogorski".equals(s))

    {

        q=
"GGATTTGGTATAATTAGTCATATATGTTTAAGAATAAGTATGTCTTCTGATGTGTTTGGGTTTTATGGTTTATTATTTGCTATG
TTTTCTATAGTTTGTTTAGGAAGAAGAGTGTGAGGTCATCACATGTTTACTGTTGGTTTAGATGTTAAGACTGCTGTGTTTTTT
AGCTCAATTACTATGATTATTGGTGTGCCTACTGGGATTAAGGTTTTTACATGATTATATATGTTATTAAATGCTCGAGTAAA
AAAGAGTGATCCTGTGTTGTGGTGAATTGTTTCATTTATAATATTGTTTACATTTGGTGGAGTTACTGGTATAGTATTGTCTGC
TTGTGTTTTAGATAAAGTGTTACATGATACTTGGTTT";

    }

    else if("Taenia laticollis".equals(s))
```

```
        {

           q=
"GGATTTGGTATAATTAGACATATATGTTTAAGTATTAGTATGTGTTCGGATGCTTTCGGTTTTTATGGTTTATTATTTGCTATG
TTTTCTATTGTTTGTTTAGGGAGAAGAGTTTGGGGTCATCATATGTTTACAGTTGGGTTAGATGTTAAGACGGCTGTATTTTTT
AGTTCTGTAACTATGATTATTGGTGTACCTACAGGTATAAAGGTTTTTACATGATTATATATGCTTTTAAATTCTCGGGTTAAA
AAGAGTGATCCTGTATTATGGTGGATAGTTTCTTTTATAGTTTTGTTTACGTTTGGTGGTGTTACAGGAATAGTGTTGTCTGCT
TGCGTATTAGATAAAGTATTACATGATACTTGATTT";

        }

        else if("Taenia madoquae".equals(s))

        {

           q=
"GGTTTTGGGATAATTAGTCATATATGTTTGAGGATTAGTATGTGTCCTGATGCTTTTGGTTTTTATGGTTTGTTATTTGCTATG
TTCTCAATAGTGTGTTTGGGAAGGAGTGTATGAGGTCATCACATGTTTACGGTTGGATTAGATGTTAAGACTGCTGTATTTTT
TAGTTCGGTTACTATGATAATAGGAGTACCTACAGGAATAAAGGTTTTTACTTGACTTTATATGCTTTTAAATTCTCGTGTGA
ATAAGAGTGATCCTGTGTTATGATGGATAGTTTCTTTTATAGTATTGTTTACTTTTGGTGGTGTAACTGGTATTGTATTATCTG
CTTGTGTATTGGATAATGTTTTACATGATACTTGATTT";

        }

        else if("Taenia martis".equals(s))

        {

           q=
"GGTTTTGGTATAATTAGACATATTTGTCTAAATATAAGTATGAATTATGATTCTTTTGGTTTTTATGGTTTATTATTTGCTATG
TTTTCTATAGTTTGTTTGGGTAGTAGTGTGTGGGGTCATCATATGTTTACTGTTGGATTAGATGTTAAGACTGCTGTTTTTTTC
AGTTCAGTTACTATGATAATAGGTGTTCCTACGGGTATAAAGGTTTTTACTTGATTATATATGCTTTTAAAATCTCGTGTGAAT
AAGAGTGATCCTGTATTATGGTGAATTGTTTCTTTTATTATTTTGTTTACTTTTGGTGGTGTTACTGGTATAGTGTTATCTGCAT
GTGTTTTGGATAAAGTTCTTCATGATACTTGATTT";

        }

        else if("Taenia multiceps".equals(s))

        {

           q=
"GGTTTTGGTATAATTAGTCACATATGTTTAAGAATAAGCATGTGTCCAGATGCTTTTGGTTTTTATGGTTTATTATTTGCTATG
TTTTCAATAGTGTGTTTAGGGAGAAGTGTGTGAGGCCATCATATGTTTACAGTTGGGTTAGATGTTAAGACTGCTGTATTTTTT
AGTTCGGTTACTATGATAATAGGAGTGCCCACAGGAATAAAGGTTTTTACTTGGCTTTATATGCTTTTAAATTCTCGTGTAAA
CAAGAGTGATCCTATACTATGATGAATAGTTTCTTTTATAGTATTGTTTACTTTTGGTGGTGTAACTGGGATTGTATTGTCTGC
TTGTGTATTAGATAAAGTTTTACATGATACTTGATTT";

        }

        else if("Taenia mustelae".equals(s))

        {

           q=
"GGTTTTGGTATTATTGGTCATATATGTTTGAGTATAAGGATGTGTTCTGATGCTTTTGGGTTTTATGGATTGTTGTTTGCTATG
TTTTCTATTGTTTGTCTAGGTAGTAGAGTTTGAGGGCATCATATGTTTACTGTTGGTTTAGATGTTAAGACTGCTGTTTTTTTTA
GTTCTGTTACTATGATTATAGGAGTTCCTACTGGTATAAAGGTGTTTACTTGGTTGTATATGTTACTGAATTCTAGTGTTAACA
AGAGGGATCCTGTGTTGTGATGAATAGTGTCATTTATATTTTTGTTTACTTTTGGTGGTGTTACTGGTATAGTTTTGTCTGCTT
GTGTATTAGATAATGTTTTGCATGATACTTGATTT";

        }

        else if("Taenia ovis".equals(s))

        {

           q=
"GGATTTGGTATAATTAGTCATATTTGTTTGAGGATTAGTATGTGTCCAGATGCTTTTGGTTTTTATGGCTTATTATTTGCTATG
```

```
TTTTCTATAGTATGTTTAGGAAGAAGTGTGTGGGGGCATCATATGTTTACTGTTGGGTTGGATGTTAAGACGGCTGTATTTTTT
AGTTCGGTTACTATGATCATAGGTGTGCCTACTGGTATAAAGGTTTTTACTTGGCTTTATATGCTTCTGAAATCTCGTGTGAAT
AAGAGTGATCCTATTTTGTGATGGATAGTTTCTTTTATAGTATTATTTACTTTTGGAGGTGTGACTGGTATTGTTTTATCTGCTT
GTGTATTGGATAAAGTTCTTCATGATACTTGATTT";

        }

        else if("Taenia parva".equals(s))

        {

           q=
"GGTTTGGGATTATAAGACATATATGTTTAAGAATTAGTATGTGTGATGATGCTTTTGGTTTTTATGGTTTGTTATTTGCTATGT
TTTCTATTGTGTGTTTAGGAAGAAGTGTATGAGGCCATCATATGTTTACTGTAGGTTTAGATGTGAAGACTGCTGTGTTTTTTA
GTTCAGTAACAATGATTATCGGGGTTCCTACTGGGATAAAGGTTTTTACTTGATTATATATGTTACTTAATTCTCGTATTAATA
AGGGTGATCCTGTAATTTGATGAATTGTTTCTTTCATAGTTTTATTTACGTTTGGTGGTGTCACTGGTATAGTTTTATCAGCTT
GTGTTTTAGATAAAGTTTTGCATGATACTTGATTT";

        }

        else if("Taenia pisiformis".equals(s))

        {

           q=
"GGGTTTGGTATAATTAGTCATATATGTTTAAGAATAAGTATGTGTTCAGATGCGTTTGGTTTTTATGGTTTATTGTTTGCAAT
GTTTTCTATAGTTTGTTTAGGTAGAAGTGTATGAGGTCATCATATGTTTACTGTTGGATTAGATGTAAAGACCGCTGTGTTTTT
TAGTTCAGTAACAATGATAATTGGAGTACCTACTGGAATTAAGGTCTTTACATGACTTTATATGCTTTTAAATTCTCGTGTCA
AAAAGAGTGATCCTGTGTTGTGGTGAATAATTTCTTTTATAGTCTTATTTACTTTTGGAGGTGTAACTGGTATAGTATTATCTG
CTTGTGTTTTAGATAAAGTT-TTACATGATACTTGATTT";

        }

        else if("Taenia saginata".equals(s))

        {

           q=
"GGTTTTGGTATGATTAGTCATATATGTTTAAGAATAAGTATGTGTCCAGATGCTTTTGGTTTTTATGGTTTGTTGTTTGCTATG
TTTTCAATAGTGTGTTTGGGGAGAAGTGTGTGGGGTCATCATATGTTTACGGTTGGGTTAGATGTTAAGACTGCTGTGTTTTTT
AGTTCGGTTACTATGATAATAGGAGTACCAACAGGAATAAAGGTTTTTACTTGACTTTATATGCTTTTAAATTCTCGTGTAAA
TAAGAGTGATCCTATATTGTGGTGAATAGTTTCTTTTATAGTGTTGTTTACTTTTGGTGGTGTGACTGGTATTGTGTTGTCTGC
TTGCGTATTGGATAAAGTTTTGCATGATACTTGATTT";

        }

        else if("Taenia serialis".equals(s))

        {

           q=
"GGTTTTGGTATAATTAGTCATATATGTTTAAGAATAAGTATGTGTCCAGACGCTTTTGGTTTTTATGGTTTGTTATTTGCTATG
TTCTCAATAGTGTGTTTAGGAAGGAGTGTATGGGGTCATCATATGTTTACAGTTGGGTTAGATGTTAAGACTGCTGTATTTTTT
AGCTCAGTTACTATGGTAATAGGAGTACCAACAGGAATAAAGGTTTTTACTTGGCTTTATATGTTATTAAATTCTCGTGTGAA
TAAGAGTGATCCTATATTGTGGTGGATAGTTTCTTTTATAGTATTGTTTACTTTTGGTGGTGTAACTGGGATTGTATTGTCTGC
TTGTGTGTTGGATAAAGTTTTACATGATACTTGGTTT";

        }

        else if("Taenia solium".equals(s))

        {

           q=
"GGGTTTGGTATAATTAGTCATATATGTTTGAGTATAAGTATGTGTTCTGATGCTTTTGGCTTTTATGGGTTATTGTTTGCTATG
TTTTCAATAGTATGTTTAGGAAGAAGTGTGTGAGGACATCATATGTTTACGGTTGGGTTAGATGTTAAGACGGCTGTATTTTTT
TAGTTCTGTTACTATGATAATTGGAGTGCCTACGGGGATTAAGGTTTTTACTTGGCTTTATATGCTTTTAAAATCTCGTGTTAA
TAAGAGTGATCCGGTTTTATGATGAATAATTTCGTTTATAGTATTGTTTACATTTGGTGGTGTAACCGGTATTATTCTATCTGC
TTGTGTATTAGATAAAGTTCTTCATGATACTTGGTTT";
```

```
        }

        else if("Taenia twitchelli".equals(s))

        {

            q=
"GGTTTTGGTATAATTAGACATATTTGTTTAAATGTAAGTATGAATTATGATTCTTTTGGATTTTATGGTTTGTTATTTGCTATG
TTTTCTATAGTTTGTTTAGGTAGAAGTGTATGAGGTCATCATATGTTTACTGTTGGATTAGATGTTAAGACTGCTGTTTTTTTT
AGTTCTGTTACTATGATTATAGGGGTTCCTACAGGTATAAAGGTGTTTACTTGGTTATATATGCTTTTAAAATCTCGTGTAAAT
AAGAGTGATCCTGTTTTATGATGAATTGTGTCTTTTATTATTTTGTTTACTTTTGGTGGTGTTACTGGTATAGTGTTATCTGCGT
GTGTTTTGGATAAAGTTCTTCATGATACTTGGTTT";

        }

        else if("Taeniopygia guttata".equals(s))

        {

            q=
"GGTTTCGGCATCATCTCCCACGTCGTAACCTACTATTCAGGTAAAAAAGAACCATTCGGATATATAGGAATAGTATGAGCTA
TGCTATCCATCGGATTCCTAGGATTCATCGTATGAGCCCACCACATGTTTACAGTAGGAATGGACGTAGACACCCGAGCATA
CTTTACATCCGCCACTATAATCATCGCCATCCCAACCGGCATCAAAGTATTCAGCTGACTAGCAACACTCCACGGAGGCACA
ATCAAGTGAGACCCACCAATACTATGAGCTCTAGGATTTATCTTCCTATTCACCATCGGAGGCCTAACCGGAATCGTCCTGGC
CAACTCCTCACTAGACATCGCCCTACACGACACCTACTACGTAGTAGCCCACTTCCACTACGTCCTATCAATAGGAGCAGTGT
TTGCAATCCTAGCAGGATTCACCCACTGATT";

        }




double
AAA=0,AAC=0,AAG=0,AAT=0,ACA=0,ACC=0,ACG=0,ACT=0,AGA=0,AGC=0,AGG=0,AGT=0,ATA=0,ATC=0,ATG=0,ATT=0,CA
A=0,CAC=0,CAG=0,CAT=0,CCA=0,CCC=0,CCG=0,CCT=0,CGA=0,CGC=0,CGG=0,CGT=0,CTA=0,CTC=0,CTG=0,CTT=0,GAA=0,G
AC=0,GAG=0,GAT=0,GCA=0,GCC=0,GCG=0,GCT=0,GGA=0,GGC=0,GGG=0,GGT=0,GTA=0,GTC=0,GTG=0,GTT=0,TAA=0,TAC=
0,TAG=0,TAT=0,TCA=0,TCC=0,TCG=0,TCT=0,TGA=0,TGC=0,TGG=0,TGT=0,TTA=0,TTC=0,TTG=0,TTT=0;

int length = q.length();

for(int i=0;i<length-2;i++)

{

if(q.charAt(i)=='A')

{

                if(q.charAt(i+1)=='A')

                {

                                if(q.charAt(i+2)=='A')

                                AAA++;

                                else if(q.charAt(i+2)=='C')

                                AAC++;

                                else if(q.charAt(i+2)=='G')

                                AAG++;

                                else if(q.charAt(i+2)=='T')

                                AAT++;
                }
                else if(q.charAt(i+1)=='C')
                {
                                if(q.charAt(i+2)=='A')
                                ACA++;
                                else if(q.charAt(i+2)=='C')
                                ACC++;
                                else if(q.charAt(i+2)=='G')
                                ACG++;
                                else if(q.charAt(i+2)=='T')
                                ACT++;


                }
                else if(q.charAt(i+1)=='G')
                {
                                if(q.charAt(i+2)=='A')
                                AGA++;
                                else if(q.charAt(i+2)=='C')
                                AGC++;
                                else if(q.charAt(i+2)=='G')
                                AGG++;
                                else if(q.charAt(i+2)=='T')
                                AGT++;
                }
                else if(q.charAt(i+1)=='T')
                {
                                if(q.charAt(i+2)=='A')
                                ATA++;
                                else if(q.charAt(i+2)=='C')
                                ATC++;
                                else if(q.charAt(i+2)=='G')
                                ATG++;
                                else if(q.charAt(i+2)=='T')
                                ATT++;
                }
```

```
}
else if(q.charAt(i)=='C')
{
        if(q.charAt(i+1)=='A')
        {
                if(q.charAt(i+2)=='A')
                CAA++;
                else if(q.charAt(i+2)=='C')
                CAC++;
                else if(q.charAt(i+2)=='G')
                CAG++;
                else if(q.charAt(i+2)=='T')
                CAT++;
        }
        else if(q.charAt(i+1)=='C')
        {
                if(q.charAt(i+2)=='A')
                CCA++;
                else if(q.charAt(i+2)=='C')
                CCC++;
                else if(q.charAt(i+2)=='G')
                CCG++;
                else if(q.charAt(i+2)=='T')
                CCT++;
        }
        else if(q.charAt(i+1)=='G')
        {
                if(q.charAt(i+2)=='A')
                CGA++;
                else if(q.charAt(i+2)=='C')
                CGC++;
                else if(q.charAt(i+2)=='G')
                CGG++;
                else if(q.charAt(i+2)=='T')
                CGT++;
        }
        else if(q.charAt(i+1)=='T')
        {
                if(q.charAt(i+2)=='A')
                CTA++;
                else if(q.charAt(i+2)=='C')
                CTC++;
                else if(q.charAt(i+2)=='G')
                CTG++;
                else if(q.charAt(i+2)=='T')
                CTT++;
        }
}
else if(q.charAt(i)=='G')
{
        if(q.charAt(i+1)=='A')
        {
                if(q.charAt(i+2)=='A')
                GAA++;
                else if(q.charAt(i+2)=='C')
                GAC++;
                else if(q.charAt(i+2)=='G')
                GAG++;
                else if(q.charAt(i+2)=='T')
                GAT++;
        }
        else if(q.charAt(i+1)=='C')
        {
                if(q.charAt(i+2)=='A')
                GCA++;
                else if(q.charAt(i+2)=='C')
                GCC++;
                else if(q.charAt(i+2)=='G')
                GCG++;
                else if(q.charAt(i+2)=='T')
```

```
                    GCT++;
            }
            else if(q.charAt(i+1)=='G')
            {
                    if(q.charAt(i+2)=='A')
                    GGA++;
                    else if(q.charAt(i+2)=='C')
                    GGC++;
                    else if(q.charAt(i+2)=='G')
                    GGG++;
                    else if(q.charAt(i+2)=='T')
                    GGT++;
            }
            else if(q.charAt(i+1)=='T')
            {
                    if(q.charAt(i+2)=='A')
                    GTA++;
                    else if(q.charAt(i+2)=='C')
                    GTC++;
                    else if(q.charAt(i+2)=='G')
                    GTG++;
                    else if(q.charAt(i+2)=='T')
                    GTT++;
            }
    }
    else if(q.charAt(i)=='T')
    {
        if(q.charAt(i+1)=='A')
        {
                    if(q.charAt(i+2)=='A')
                    TAA++;
                    else if(q.charAt(i+2)=='C')
                    TAC++;
                    else if(q.charAt(i+2)=='G')
                    TAG++;
```

```
		else if(q.charAt(i+2)=='T')
		TAT++;
	}
	else if(q.charAt(i+1)=='C')
	{
		if(q.charAt(i+2)=='A')
		TCA++;
		else if(q.charAt(i+2)=='C')
		TCC++;
		else if(q.charAt(i+2)=='G')
		TCG++;
		else if(q.charAt(i+2)=='T')
		TCT++;
	}
	else if(q.charAt(i+1)=='G')
	{
		if(q.charAt(i+2)=='A')
		TGA++;
		else if(q.charAt(i+2)=='C')
		TGC++;
		else if(q.charAt(i+2)=='G')
		TGG++;
		else if(q.charAt(i+2)=='T')
		TGT++;
	}
	else if(q.charAt(i+1)=='T')
	{
		if(q.charAt(i+2)=='A')
		TTA++;
		else if(q.charAt(i+2)=='C')
		TTC++;
		else if(q.charAt(i+2)=='G')
		TTG++;
		else if(q.charAt(i+2)=='T')
		TTT++;
```

```
            }
        }
    }

        AAA=AAA/length;
        AAC=AAC/length;
        AAG=AAG/length;
        AAT=AAT/length;
        ACA=ACA/length;
        ACC=ACC/length;
        ACG=ACG/length;
        ACT=ACT/length;
        AGA=AGA/length;
        AGC=AGC/length;
        AGG=AGG/length;
        AGT=AGT/length;
        ATA=ATA/length;
        ATC=ATC/length;
        ATG=ATG/length;
        ATT=ATT/length;

        CAA=CAA/length;
        CAC=CAC/length;
        CAG=CAG/length;
        CAT=CAT/length;

        CCA=CCA/length;
        CCC=CCC/length;
        CCG=CCG/length;
        CCT=CCT/length;

        CGA=CGA/length;
        CGC=CGC/length;
        CGG=CGG/length;
        CGT=CGT/length;
```

```
CTA=CTA/length;
CTC=CTC/length;
CTG=CTG/length;
CTT=CTT/length;

GAA=GAA/length;
GAC=GAC/length;
GAG=GAG/length;
GAT=GAT/length;

GCA=GCA/length;
GCC=GCC/length;
GCG=GCG/length;
GCT=GCT/length;

GTA=GTA/length;
GTC=GTC/length;
GTG=GTG/length;
GTT=GTT/length;

GGA=GGA/length;
GGC=GGC/length;
GGG=GGG/length;
GGT=GGT/length;

TAA=TAA/length;
TAC=TAC/length;
TAG=TAG/length;
TAT=TAT/length;

TCA=TCA/length;
TCC=TCC/length;
TCG=TCG/length;
TCT=TCT/length;
```

```
  TGA=TGA/length;

  TGC=TGC/length;

  TGG=TGG/length;

  TGT=TGT/length;


  TTA=TTA/length;

  TTC=TTC/length;

  TTG=TTG/length;

  TTT=TTT/length;




  double[]
e={AAA,AAC,AAG,AAT,ACA,ACC,ACG,ACT,AGA,AGC,AGG,AGT,ATA,ATC,ATG,ATT,CAA,CAC,CAG,CAT,CCA,CCC,CCG,CC
T,CGA,CGC,CGG,CGT,CTA,CTC,CTG,CTT,GAA,GAC,GAG,GAT,GCA,GCC,GCG,GCT,GGA,GGC,GGG,GGT,GTA,GTC,GTG,GTT,
TAA,TAC,TAG,TAT,TCA,TCC,TCG,TCT,TGA,TGC,TGG,TGT,TTA,TTC,TTG,TTT};

  return e;



}
  CategoryDataset GetData1(String s1,String s2) {


    // row keys...

    final String series1 = s1+"Dinucleotide Frequency";

    final String series2 = s2+"Dinucleotide Frequency";


    // column keys...

    final String type1 = "AA";

    final String type2 = "AC";

    final String type3 = "AG";

    final String type4 = "AT";

    final String type5 = "CA";

    final String type6 = "CC";

    final String type7 = "CG";

    final String type8 = "CT";
final String type9 = "GA";

final String type10 = "GC";

final String type11= "GG";

final String type12 = "GT";

final String type13 = "TA";

final String type14 = "TC";

final String type15 = "TG";

final String type16 = "TT";




// create the dataset...

final DefaultCategoryDataset dataset = new DefaultCategoryDataset();

double[] d=calcDifreq(s1);


dataset.addValue(d[0], series1,type1);

dataset.addValue(d[1], series1,type2);

dataset.addValue(d[2], series1,type3);

dataset.addValue(d[3], series1,type4);

dataset.addValue(d[4], series1,type5);

dataset.addValue(d[5], series1,type6);

dataset.addValue(d[6], series1,type7);

dataset.addValue(d[7], series1,type8);

dataset.addValue(d[8], series1,type9);

dataset.addValue(d[9], series1,type10);

dataset.addValue(d[10], series1,type11);

dataset.addValue(d[11], series1,type12);

dataset.addValue(d[12], series1,type13);

dataset.addValue(d[13], series1,type14);

dataset.addValue(d[14], series1,type15);

dataset.addValue(d[15], series1,type16);


d=calcDifreq(s2);


dataset.addValue(d[0], series2,type1);

dataset.addValue(d[1], series2,type2);
```

```
    dataset.addValue(d[2], series2,type3);

    dataset.addValue(d[3], series2,type4);

    dataset.addValue(d[4], series2,type5);

    dataset.addValue(d[5], series2,type6);

    dataset.addValue(d[6], series2,type7);

    dataset.addValue(d[7], series2,type8);

    dataset.addValue(d[8], series2,type9);

    dataset.addValue(d[9], series2,type10);

    dataset.addValue(d[10], series2,type11);

    dataset.addValue(d[11], series2,type12);

    dataset.addValue(d[12], series2,type13);

    dataset.addValue(d[13], series2,type14);

    dataset.addValue(d[14], series2,type15);

    dataset.addValue(d[15], series2,type16);




    return dataset;


}

CategoryDataset GetData2(String s1,String s2) {


    // row keys...

    final String series1 = s1+"Trinucleotide Frequency";

    final String series2 = s2+"Trinucleotide Frequency";






    // column keys...

    final String type1a = "AAA";

    final String type1b = "AAC";

    final String type1c = "AAG";

    final String type1d = "AAT";
final String type2a = "ACA";

final String type2b = "ACC";

final String type2c = "ACG";

final String type2d = "ACT";


final String type3a = "AGA";

final String type3b = "AGC";

final String type3c = "AGG";

final String type3d = "AGT";


final String type4a = "ATA";

final String type4b = "ATC";

final String type4c = "ATG";

final String type4d = "ATT";


final String type5a = "CAA";

final String type5b = "CAC";

final String type5c = "CAG";

final String type5d = "CAT";


final String type6a = "CCA";

final String type6b = "CCC";

final String type6c = "CCG";

final String type6d = "CCT";


final String type7a = "CGA";

final String type7b = "CGC";

final String type7c = "CGG";

final String type7d = "CGT";


final String type8a = "CTA";

final String type8b= "CTC";

final String type8c = "CTG";

final String type8d = "CTT";
final String type9a = "GAA";

final String type9b = "GAC";

final String type9c = "GAG";

final String type9d = "GAT";


final String type10a = "GCA";

final String type10b = "GCC";

final String type10c = "GCG";

final String type10d = "GCT";


final String type11a = "GGA";

final String type11b = "GGC";

final String type11c = "GGG";

final String type11d = "GGT";


final String type12a = "GTA";

final String type12b = "GTC";

final String type12c = "GTG";

final String type12d = "GTT";


final String type13a = "TAA";

final String type13b = "TAC";

final String type13c = "TAG";

final String type13d = "TAT";


final String type14a = "TCA";

final String type14b = "TCC";

final String type14c = "TCG";

final String type14d = "TCT";


final String type15a = "TGA";

final String type15b = "TGC";

final String type15c = "TGG";

final String type15d = "TGT";
final String type16a = "TTA";

final String type16b = "TTC";

final String type16c = "TTG";

final String type16d = "TTT";


// create the dataset...

final DefaultCategoryDataset dataset = new DefaultCategoryDataset();

double[] d=calcTrifreq(s2);


dataset.addValue(d[0], series2,type1a);

dataset.addValue(d[1], series2,type1b);

dataset.addValue(d[2], series2,type1c);

dataset.addValue(d[3], series2,type1d);


dataset.addValue(d[4], series2,type2a);

dataset.addValue(d[5], series2,type2b);

dataset.addValue(d[6], series2,type2c);

dataset.addValue(d[7], series2,type2d);


dataset.addValue(d[8], series2,type3a);

dataset.addValue(d[9], series2,type3b);

dataset.addValue(d[10], series2,type3c);

dataset.addValue(d[11], series2,type3d);


dataset.addValue(d[12], series2,type4a);

dataset.addValue(d[13], series2,type4b);

dataset.addValue(d[14], series2,type4c);

dataset.addValue(d[15], series2,type4d);


dataset.addValue(d[16], series2,type5a);

dataset.addValue(d[17], series2,type5b);

dataset.addValue(d[18], series2,type5c);

dataset.addValue(d[19], series2,type5d);
dataset.addValue(d[20], series2,type6a);
dataset.addValue(d[21], series2,type6b);
dataset.addValue(d[22], series2,type6c);
dataset.addValue(d[23], series2,type6d);

dataset.addValue(d[24], series2,type7a);
dataset.addValue(d[25], series2,type7b);
dataset.addValue(d[26], series2,type7c);
dataset.addValue(d[27], series2,type7d);

dataset.addValue(d[28], series2,type8a);
dataset.addValue(d[29], series2,type8b);
dataset.addValue(d[30], series2,type8c);
dataset.addValue(d[31], series2,type8d);

dataset.addValue(d[32], series2,type9a);
dataset.addValue(d[33], series2,type9b);
dataset.addValue(d[34], series2,type9c);
dataset.addValue(d[35], series2,type9d);

dataset.addValue(d[36], series2,type10a);
dataset.addValue(d[37], series2,type10b);
dataset.addValue(d[38], series2,type10c);
dataset.addValue(d[39], series2,type10d);

dataset.addValue(d[40], series2,type11a);
dataset.addValue(d[41], series2,type11b);
dataset.addValue(d[42], series2,type11c);
dataset.addValue(d[43], series2,type11d);

dataset.addValue(d[44], series2,type12a);
dataset.addValue(d[45], series2,type12b);
dataset.addValue(d[46], series2,type12c);
dataset.addValue(d[47], series2,type12d);
dataset.addValue(d[48], series2,type13a);

dataset.addValue(d[49], series2,type13b);

dataset.addValue(d[50], series2,type13c);

dataset.addValue(d[51], series2,type13d);


dataset.addValue(d[52], series2,type14a);

dataset.addValue(d[53], series2,type14b);

dataset.addValue(d[54], series2,type14c);

dataset.addValue(d[55], series2,type14d);


dataset.addValue(d[56], series2,type15a);

dataset.addValue(d[57], series2,type15b);

dataset.addValue(d[58], series2,type15c);

dataset.addValue(d[59], series2,type15d);


dataset.addValue(d[60], series2,type16a);

dataset.addValue(d[61], series2,type16b);

dataset.addValue(d[62], series2,type16c);

dataset.addValue(d[63], series2,type16d);


d=calcTrifreq(s1);

dataset.addValue(d[0], series1,type1a);

dataset.addValue(d[1], series1,type1b);

dataset.addValue(d[2], series1,type1c);

dataset.addValue(d[3], series1,type1d);


dataset.addValue(d[4], series1,type2a);

dataset.addValue(d[5], series1,type2b);

dataset.addValue(d[6], series1,type2c);

dataset.addValue(d[7], series1,type2d);


dataset.addValue(d[8], series1,type3a);

dataset.addValue(d[9], series1,type3b);

dataset.addValue(d[10], series1,type3c);

dataset.addValue(d[11], series1,type3d);
dataset.addValue(d[12], series1,type4a);
dataset.addValue(d[13], series1,type4b);
dataset.addValue(d[14], series1,type4c);
dataset.addValue(d[15], series1,type4d);

dataset.addValue(d[16], series1,type5a);
dataset.addValue(d[17], series1,type5b);
dataset.addValue(d[18], series1,type5c);
dataset.addValue(d[19], series1,type5d);

dataset.addValue(d[20], series1,type6a);
dataset.addValue(d[21], series1,type6b);
dataset.addValue(d[22], series1,type6c);
dataset.addValue(d[23], series1,type6d);

dataset.addValue(d[24], series1,type7a);
dataset.addValue(d[25], series1,type7b);
dataset.addValue(d[26], series1,type7c);
dataset.addValue(d[27], series1,type7d);

dataset.addValue(d[28], series1,type8a);
dataset.addValue(d[29], series1,type8b);
dataset.addValue(d[30], series1,type8c);
dataset.addValue(d[31], series1,type8d);

dataset.addValue(d[32], series1,type9a);
dataset.addValue(d[33], series1,type9b);
dataset.addValue(d[34], series1,type9c);
dataset.addValue(d[35], series1,type9d);

dataset.addValue(d[36], series1,type10a);
dataset.addValue(d[37], series1,type10b);
dataset.addValue(d[38], series1,type10c);
dataset.addValue(d[39], series1,type10d);
dataset.addValue(d[40], series1,type11a);

dataset.addValue(d[41], series1,type11b);

dataset.addValue(d[42], series1,type11c);

dataset.addValue(d[43], series1,type11d);


dataset.addValue(d[44], series1,type12a);

dataset.addValue(d[45], series1,type12b);

dataset.addValue(d[46], series1,type12c);

dataset.addValue(d[47], series1,type12d);


dataset.addValue(d[48], series1,type13a);

dataset.addValue(d[49], series1,type13b);

dataset.addValue(d[50], series1,type13c);

dataset.addValue(d[51], series1,type13d);


dataset.addValue(d[52], series1,type14a);

dataset.addValue(d[53], series1,type14b);

dataset.addValue(d[54], series1,type14c);

dataset.addValue(d[55], series1,type14d);


dataset.addValue(d[56], series1,type15a);

dataset.addValue(d[57], series1,type15b);

dataset.addValue(d[58], series1,type15c);

dataset.addValue(d[59], series1,type15d);


dataset.addValue(d[60], series1,type16a);

dataset.addValue(d[61], series1,type16b);

dataset.addValue(d[62], series1,type16c);

dataset.addValue(d[63], series1,type16d);




return dataset;
```

```
    }
    void drawchart1(CategoryDataset dataset)
    {
       JFrame frame1 = new JFrame();
    JFreeChart chart;
       chart = ChartFactory.createLineChart(
              "Line Chart For "+a+" and "+b,      // chart title
              "Frequency",                 // domain axis label
              "Type",                // range axis label
              dataset,                // data
              PlotOrientation.VERTICAL,  // orientation
              true,                  // include legend
              true,                  // tooltips
              false                  // urls
       );
       ChartPanel chartPanel = new ChartPanel(chart);
       chartPanel.setPreferredSize(new Dimension(700, 350));
       frame1.setContentPane(chartPanel);
       frame1.setSize(1300,500);
       frame1.setVisible(true);

       final CategoryPlot plot = (CategoryPlot) chart.getPlot();
       plot.setBackgroundPaint(Color.lightGray);
       plot.setRangeGridlinePaint(Color.white);
       plot.setDomainGridlinePaint(Color.white);
       plot.setDomainGridlinesVisible(true);
       plot.setRangeGridlinesVisible(true);

       chart.setBackgroundPaint(Color.white);

       final NumberAxis rangeAxis = (NumberAxis) plot.getRangeAxis();
       rangeAxis.setStandardTickUnits(NumberAxis.createStandardTickUnits());
       rangeAxis.setAutoRangeIncludesZero(true);
       final LineAndShapeRenderer renderer = (LineAndShapeRenderer) plot.getRenderer();
//        renderer.setDrawShapes(true);
```

```
        renderer.setSeriesStroke(
            0, new BasicStroke(
                2.0f, BasicStroke.CAP_ROUND, BasicStroke.JOIN_ROUND,
                1.0f, new float[] {10.0f, 6.0f}, 0.0f
            )
        );
        renderer.setSeriesStroke(
            1, new BasicStroke(
                2.0f, BasicStroke.CAP_ROUND, BasicStroke.JOIN_ROUND,
                1.0f, new float[] {6.0f, 6.0f}, 0.0f
            )
        );
        renderer.setSeriesStroke(
            2, new BasicStroke(
                2.0f, BasicStroke.CAP_ROUND, BasicStroke.JOIN_ROUND,
                1.0f, new float[] {2.0f, 6.0f}, 0.0f
            )
        );
    }
}
```