

CPT111 2223

Coursework 3 Task Sheet

Overview

Coursework 3 (CW3) is the final coursework component of the course this semester. It contributes to **30% of your final marks**.

You will use your object-oriented techniques, file processing, and data structures you have learned throughout the semester to solve a problem that consists of two main tasks. In addition, you will create a video presentation to showcase your problem solving knowledge and algorithm analysis skill, which mainly involves string processing.

Submit the Java source file, the video mp4 file, and the ppt used in the video to Learning Mall for grading. Announcement regarding code submission will be given. Complete the Ethics and Privacy Online Quiz in Learning Mall as well.

Timeline

Week 11, Tuesday , Nov 22, 2022, 18:00 CST	CW3 is released (This task sheet, skeleton codes, partial test cases)
Week 14 , TBA	CW3 Java source file, video files (mp4, ppt) and the ethics and privacy online quiz are open
Week 14, Sunday , Dec 18, 2022, 23:59 CST	CW3 Java source file, video files (mp4, ppt), and the online quiz are due
Late Submission Period	5% lateness penalty per-day Max 5 days (Monday-Friday)
Reading Week, Friday , Dec 23, 2022, 23:59 CST	End of Late Submission Period No submissions are accepted thereafter

Outline

The rest of the task sheet will describe the background of the problem, detailed specification of the two main tasks, and the deliverables you have to submit.

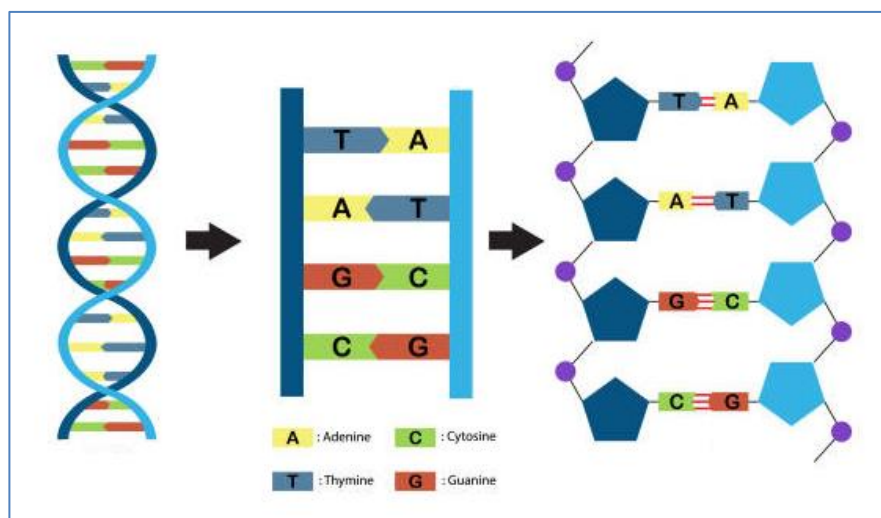
Coursework 3 – DNA for Profiling and Disease Detection

Background

DNA carries the genetic information in living beings. Interestingly, it has been used in criminal justice system for profiling work, as well as disease diagnosis in medicine. In this coursework, your task is to develop algorithms for those two purposes.

DNA

Deoxyribonucleic acid (DNA) is a sequence of molecules called nucleotides, arranged into a double helix shape. Each nucleotide of DNA contains one of four different bases: Adenine (A), Cytosine (C), Guanine (G), or Thymine (T).




Every human cell has billions of these nucleotides arranged in sequence. Some portions of this sequence are the same or very similar, across almost all humans. However, there are some portions of the sequence have a higher genetic diversity and thus vary more across the population.

Short Tandem Repeats (STRs)

One place where DNA tends to have high genetic diversity is in Short Tandem Repeats (STRs). An STR is a short sequence of DNA bases that is repeated continuously numerous times at specific locations in DNA. The number of times any particular STR repeats varies a lot among different people.

In the DNA samples below, for example, Alice has the STR **AAGT** repeated back-to-back three times in her DNA, while Bob has the same STR repeated back-to-back four times.



Alice: CT**AAGT****AAGT****AAGT**AAGATA
Bob: CT**AAGT****AAGT****AAGT****AAGT**TA

The diagram shows two DNA sequences within a blue rectangular border. Alice's sequence is 'CTAAGTAAGTAAGTAAGATA' and Bob's is 'CTAAGTAAGTAAGTAAGTTA'. In both sequences, the 'AAGT' STR is repeated consecutively. In Alice's sequence, the first three 'AAGT' repeats are each enclosed in a red rounded rectangle. In Bob's sequence, all four 'AAGT' repeats are each enclosed in a red rounded rectangle.

DNA Profiling and Database

DNA profiling is a procedure used to identify individuals on the basis of their unique genetic makeup. Recording the number of STR of the population in a DNA database, and then firstly using it for searching can help speeding up the identification process.

Using multiple STRs, we can improve the accuracy of DNA profiling. If the probability that two people have the same number of a single STR is 5% and we look at 10 different STRs, then the probability that two DNA samples match solely by chance (assuming independence of all STRs) is about 1 in 1 quadrillion. So, if two DNA samples match in the number of continuous repeats for each of the STRs, we can have enough confidence that they came from the same person.

Let us have a very simple DNA database in the form of a CSV file. Each row corresponds to an individual, and each column corresponds to a particular STR.

For example, `database.csv` contains:

```
name,AAGT,ACTC,TATG
```

```
Alice,3,10,8
```

```
Bob,4,2,8
```

The data in the above CSV file would suggest that Alice has the sequence AAGT repeated 3 times consecutively somewhere in her DNA, the sequence ACTC repeated 10 times, and TATG repeated 8 times. Bob, meanwhile, has those same three STRs repeated 4 times, 2 times, and 8 times, respectively.

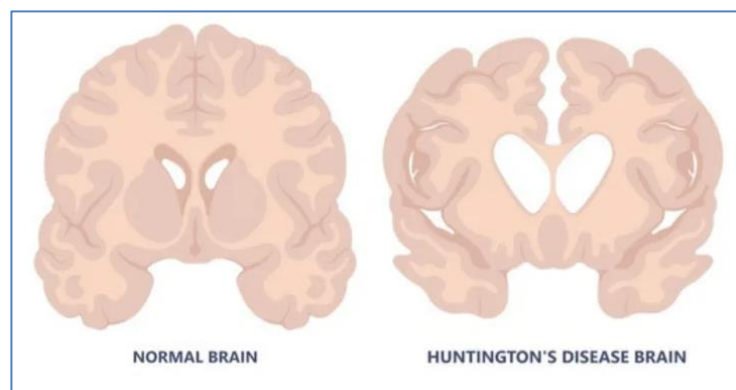
Next, a sequence of DNA is queried to the database. Given that sequence of DNA, how can one identify to whom it belongs? Well, for example, one may first search for the longest length of consecutive repeats of AAGT in the sequence, followed similarly by ACTC and TATG. If one then found that the longest sequence of AAGTs is 3 repeats long, ACTCs is 10 repeats long, and TATGs is 8; one may as a result

conclude that the DNA was Alice's. Finally, it's also possible that after one takes the counts for each of the STRs, it doesn't match anyone in the DNA database, in which case one reports no match.

One of your task is to write a program that will first take a CSV file containing STR counts for a list of individuals, build a DNA database of your own, take another TXT file that contains a DNA sequence, and then output to whom the DNA belongs or reports no match.

Huntington's Disease Diagnosis

Huntington's disease (HD) is an inherited and terminal neurological disorder. It is a condition that stops parts of the brain working properly over time, and is usually fatal after a period of up to 20 years.



At this time, there is no cure for HD. However, in 1993, a group of scientists discovered a very accurate genetic test for diagnosing HD. The gene that causes HD is actually located on Chromosome 4, and has a consecutive repeats of **CAG**. The normal range of CAG repeats is between 10 and 35. Individuals with HD have between 36 and 180 repeats.

Doctors use a certain DNA test to count the number of CAG repeats; and consult the following table to produce a diagnosis:

Number of Repeats	Diagnosis
0 - 9	Faulty Test
10 - 35	Normal
36 - 39	High Risk
40 - 180	Huntington's
≥ 181	Faulty Test

The other one of your task is to write a method that based on the DNA sequence read before, will analyze that sequence for Huntington's disease and produce a diagnosis following the table above.

Specification and Deliverables

In this section, you will find details about your implementation and the files that you have to submit.

Specification

Your implementation must satisfy the following specification:

1. You will implement your program in `DnaProfileDiagnosis.java`.
2. A new object of `DnaProfileDiagnosis` is created by calling `DnaProfileDiagnosis` constructor. The file name of the CSV file containing the DNA database would be passed to the constructor.
3. Your program should open the CSV file and read its contents into the instance variables. You may assume that the first row of the CSV file will be the column names. The first column will be the word name and the remaining columns would be the STR sequences. The following columns would be the actual name and the corresponding STR counts.
4. The file name of the TXT file containing the DNA sequence would be passed to the `readDna` instance method. Your program should open the TXT file and read its contents into the instance variables.
5. The DNA sequence in the TXT file may contain some whitespace (spaces, tabs, newlines). Your program must remove any whitespace before storing and computing on it.
6. The method `checkProfile` could then be called, after setting the query sequence. Your algorithm will try to match the STRs counts of the database and the DNA sequence. If a match is found, the name of the individual will be returned as a `String`, such as "Alice". Otherwise, the `String` "No match" will be returned.
You may assume the STR counts will *not* match more than one individual.
7. Calling the `checkProfile` method before setting the DNA sequence would cause an `IllegalArgumentException` to be thrown.
8. The method `diagnoseHd` could also then be called after setting the DNA sequence.
Your algorithm will perform a diagnosis based on the CAG repeats and the table at the previous section. The output of the method would be one of the following `Strings`: "Faulty Test", "Normal", "High Risk", or "Huntington's".
9. Calling the `diagnoseHd` method before setting the DNA sequence would cause an `IllegalArgumentException` to be thrown.
10. Another `readDna` calls may be made to change the DNA sequence.

Instance Variable and Complexity Requirements

In this Coursework 3, to store, to query and compute on the DNA database and the DNA sequence, you must use LinkedList and/or HashMap, and their methods.

```
public class DnaProfileDiagnosis {
    // you may modify/add more instance variables
    // but your algorithms must primarily use the following
    // list and/or map
    private LinkedList list;
    private HashMap map;
    private String dna;
    ...
}
```

Failing in satisfying this requirement would result in **0 marks**.

There is **no requirements** on the running time of your program.

Public API

```
public class DnaProfileDiagnosis {

    // build a database from database.csv
    public DnaProfileDiagnosis(String database)

    // store a dna sequence with no whitespace from dna.txt
    public void readDna(String dna)

    // based on the STR counts, return either a name in
    // database, or "No Match"
    // throws IllegalArgumentException if dna has not been set
    public String checkProfile()

    // based on the CAG repeats, return either "Faulty Test",
    // "Normal", "High Risk", or "Huntington's"
    // throws IllegalArgumentException if dna has not been set
    public String diagnoseHd()

}
```

Sample Client

Your program should behave as the example below:

```
public class TestCoursework {

    public static void main(String[] args) {

        String db1 = "data/db1.csv";
        DnaProfileDiagnosis test = new
                                DnaProfileDiagnosis(db1);
        String dna1 = "data/dna1.txt";
        test.readDna(dna1);
        System.out.println(test.checkProfile()); // Alice
        System.out.println(test.diagnoseHd());   // Normal

        String dna2 = "data/dna2.txt";
        test.readDna(dna2);
        System.out.println(test.checkProfile()); // Bob
        System.out.println(test.diagnoseHd());   // Huntington's

        String db2 = "data/db2.csv";
        DnaProfileDiagnosis test2 = new
                                DnaProfileDiagnosis(db2);
        System.out.println(test2.checkProfile()); // Illegal
                                                // ArgumentException thrown

    }
}
```

Code, PowerPoint Slides, Video Requirements

Submit code online (TBA), and create a ppt, video and make a submission to Learning Mall with the following requirements:

1. Cite in code/ppt whenever you use materials that are **not** your own.
2. The video must contain **discussions of the algorithms** you use to complete both the profiling and the diagnosis tasks, followed by their **running time analysis**.
3. The length of the video must be **less than or equal to 4 minutes**.
Violating the length requirements will result in **0 marks** of your video grade.
4. Your video **must show your face** for the purpose of authenticity verification.
Violating the showing face requirement will result in **0 marks** in your video grade.

5. You may want to make your video look nicer, however, the grade will not be based on the looks. Only the quality and clarity of the algorithms' discussion and analysis will count.

A simple recording of a ppt explanation while showing the presenter face in a box using shared-screen by, for example, Camtasia, BBB, Zhumu, or Tencent Meeting would be sufficient.

6. Submit to Learning Mall the following:
 - a. The code (details provided later in Announcement)
 - b. The video file in **.mp4**
 - c. The **PPT** file you used to create a video
 - d. **(Optional)** A link to **YouTube** or **BiliBili** of your uploaded video

Grades

The marks of your submission:

1. Correctness of all the methods:	70 marks
2. Algorithm discussion, analysis and clarity of the video:	25 marks
3. Ethics and Privacy Online Quiz	5 marks
Total	100 marks

Academic Integrity

This Coursework is an individual work. Plagiarism, e.g. copying materials from other sources without proper acknowledgement, copying and collusion are serious academic offences. Plagiarism, copying and collusion will **not** be tolerated and will be dealt with in accordance with the University Code of Practice on Academic Integrity.

In some cases, individual students may be invited to explain parts of their code in person, and if they fail to demonstrate an understanding of the code, no credit will be given for that part.

In more severe cases, the violation will be reported to the exam officer for further investigation and will be permanently recorded in the student's official academic transcript.