|  |  |
| --- | --- |
|  | **CS 313: Intermediate Computer Programming**  **Project Work 1 (Thursday, 21st February 2019)** |

**Instructions**

You may work in teams of three (3) and four (4) members for this exercise. As you work in teams, remember that each member should be fully engaged and fully understand all work done, so that you can do the work on your own. **You are supposed to submit your work (i.e. Java project, Javadoc, and reflective report document) individually on CAMU by 11:55pm on (Saturday 16th March 2019).**

**[130 POINTS: (2 & 3) 100 points + (4 A) 10 points + (4 B) 10 points + (4 C) 10 points]**

**1. Objectives**

The objectives of this project work are for you to demonstrate that you can:

- Define classes, fields, constructors, and methods in Java

- Use appropriate types, including collections

- Implement basic algorithms using collections

- Write a main method including console I/O

- Implement GUI for Java projects with Swing.

- Implement JDBC for SQL queries and database storage.

- Document your solution with Javadoc

- Writing reflective reports

**2. Scenario: Illegal mining (i.e. “galamsey”) monitoring**

Illegal mining can have devastating effects on natural resources such as vegetation, water bodies and land, and although it is known that they are more likely to occur in some places than in others they are almost impossible to detect it during its operation. In order to help African security agencies and mining authorities understand and detect illegal mining better, an international monitoring system (using drone images) is in place to record where “galamsey” occur and how devastating they are in Africa. “Galamsey” impact is measured by the ‘vegetation colour’: geographical locations with

* green vegetation is/are identified to have a healthy vegetation with no “galamsey” presence (number value **1** is assigned),
* yellow vegetation are identified to have a fair vegetation with a potential “galamsey” presence (number value **2** is assigned), and
* brown vegetation are identified to have a poor vegetation with a certain “galamsey” presence (number value **3** is assigned).

National “galamsey” observatories record “galamsey” that occur, although different observatories were set up at different times, so the period over which historical data is available varies from place to place.

**3. Problem specification [100 POINTS]**

VERSION 1 [40 Points]

[8 POINTS]

Define a Java class Galamsey with appropriate fields, methods and constructor to store and retrieve information about the vegetation colour, colour value, position (latitude and longitude) and year of the event.

[2 points: 0.5 each] Fields: vegetation colour, colour value, position (latitude and longitude) and year of the event.

[6 points] Methods: Accessor (2), Mutator (2), Inherited/Auxiliary (1), Constructor (1)

[16 POINTS]

Define a Java class Observatory with appropriate fields, methods and constructor to store and retrieve the name of the observatory, the name of the country in which it is located, the year in which “galamsey” observations started, the area covered by the observatory (in square kilometers) and a list of “galamsey” events that it has recorded. Include methods to return:

[2.5 points: 0.5 each] Fields: the name of the observatory, the name of the country in which it is located, the year in which “galamsey” observations started, the area covered by the observatory (in square kilometers) and a list of “galamsey” events that it has recorded.

[7 points] Methods: Accessor (2.5), Mutator (2.5), Inherited (1), Constructor (1)

- [2 points] The largest “galamsey” colour value recorded by the observatory.

- [2 points] The average “galamsey” colour value recorded at the observatory.

- [2.5 points] A list of all “galamsey” recorded at the observatory with a colour value greater than a given/arbitrary number.

[9 POINTS]

Define a Java class Monitoring, which holds information about all observatories. Include methods to return:

- [2 points] The observatory with the largest average “galamsey” colour value.

- [2 points] The largest “galamsey” colour value ever recorded.

- [5 points] A list of all “galamsey” recorded with colour value greater than a given/arbitrary number.

[7 POINTS]

Define a class MonitoringIO, with a main method which does the following:

- Presents the user with a menu (printed to the console) of features:

o [2 points] enter observatory data;

o [2 points] enter “galamsey” data;

o [2 points] provide monitoring statistics on largest average “galamsey”, largest “galamsey” ever and all “galamsey” with colour value greater than a given number; or

o [0.5 point] exit

- Takes user input from the console to choose one of the menu features

- Allows the user to input, via the console, the details of observatories and “galamsey”

- [0.5 point] After executing one of the features, returns the user to the menu to choose another option

VERSION 2 [60 Points]

Modify your solution above to include graphical user interface (GUI) to allow the user to provide the details of observatories and “galamsey”s. Define a class MonitoringGUI for the described requirement, that also saves the user inputs into a database using a JDBC for storage. This feature is more dynamic and secure than saving data to a file.

[20 points] GUI Functionality: details of observatories (10: GUI[6], details[2], Objects[2]) and earthquakes (10: GUI[6], details[2], Object[2])

[25 points] JDBC

* Register [3]
* Connection [4]
* Statements [16: 5 points for database (Galamsey/Observatory), 1 pt Initialize connection, 5 pts Query }]: Demo by students
* ResultSet [2]

[5 points] Design + Creativity

All classes should be properly documented with Javadoc and include appropriate defensive programming approaches**. You may need to research relevant Java Class libraries in order to get help with certain features (such as JDBC for database storage).**

[10 points] JAVADOC

**4. What to submit [30 POINTS]**

You must submit 3 deliverables to CAMU, as follows:

o A zip file named ‘CSC313\_project\_FirstnameLastname.zip’, where ‘Firstname’ is replaced with your first name, and ‘Lastname’ is replaced with your last name. The zip file must contain:

- The source code of your Java classes (i.e. the two versions).

- The generated Javadoc for your classes.

- A PDF document named ‘CSC313\_project1Doc\_FirstnameLastname.pdf’ following the same conventions as above, containing a **reflective report** on your implementation (between 500 to 1,000 words), saying what went well and what you would do to improve or extend your program (refer to PART A below). You may use appropriate diagrams for your descriptions.

**PART A – Reflective Report (10 Points)**

From your experiences (i.e. documentation and teamwork) on the Software Engineering project, you are to submit an **individual** report indicating:

- what went well,

- what you would do to improve or extend your program,

- what new skill you have acquired from the process or through a teammate,

- justification of major decisions taken, etc.

You may use appropriate diagrams for your descriptions.

**PART B – Presentation or Demonstration of Project (10 Points)**

**NB:** [10 points] Teams will present/demo their work in class on Monday 18th March 2019.

* [4 points] Run code with reasonably appropriate sample data
* [2 points] Show Javadoc
* [4 points] Talk about reflective report

**PART C – Peer Review (10 Points)**

There will also be a peer-review within teams with regards to how well team members contributed to this project. You are supposed to assess your teammates in this part. As you all know, team dynamics and team work are critical in Computer Science projects.

|  |  |  |  |
| --- | --- | --- | --- |
| **Names** | **Group Number** | **Peer Grade**  **(Choose from 0 - 5)** | **Comments** |
| 1. (Reviewer) |  | (No grade for self) |  |
| 2. |  |  |  |
| 3. |  |  |  |
| 4. |  |  |  |

**NB:** Submit this form on CAMU after project submission and presentation/demo on Tuesday 19th March 2019.

All the best!