OBJECT ORIENTED PROGRAMMING
CCS20704

Received Date	**: 2 October 2024**
Submission Due Date	**: 10 October 2024**
Lecturer	**: Dr. Jamal Abdullahi**
Weightage	**: 10%**
Semester	**: SEPTEMBER 2024**

Instruction to students:

- This is a GROUP assignment.
- Complete this cover sheet and attach it to your assignment (first page).

Student declaration:

I declare that:

- ***This assignment is my/our own work.***
- ***I/we understand what is meant by plagiarism.***
- ***My lecturer has the right to deduct my marks in case of:***
  - ***Late submission***
  - ***Any plagiarism found in my assignment.***

NAME: NUR NAJIHA NABILA BINTI MOHD ADIRA
MATRICS NO: 2082024070012
PROGRAMME: BACHELOR COMPUTER SCIENCE

The class code is Student Info. This class includes fields for the student's name, age, id , cgrpa student and mentor name. Along with a constructor, getter methods, and a toString method for easy output. This class displays student info that has been entered such as student name Najiha, age 22, student id, student cgpa 3.4 and student mentor name. In addition, this class uses constructor parameters that display information related to students.

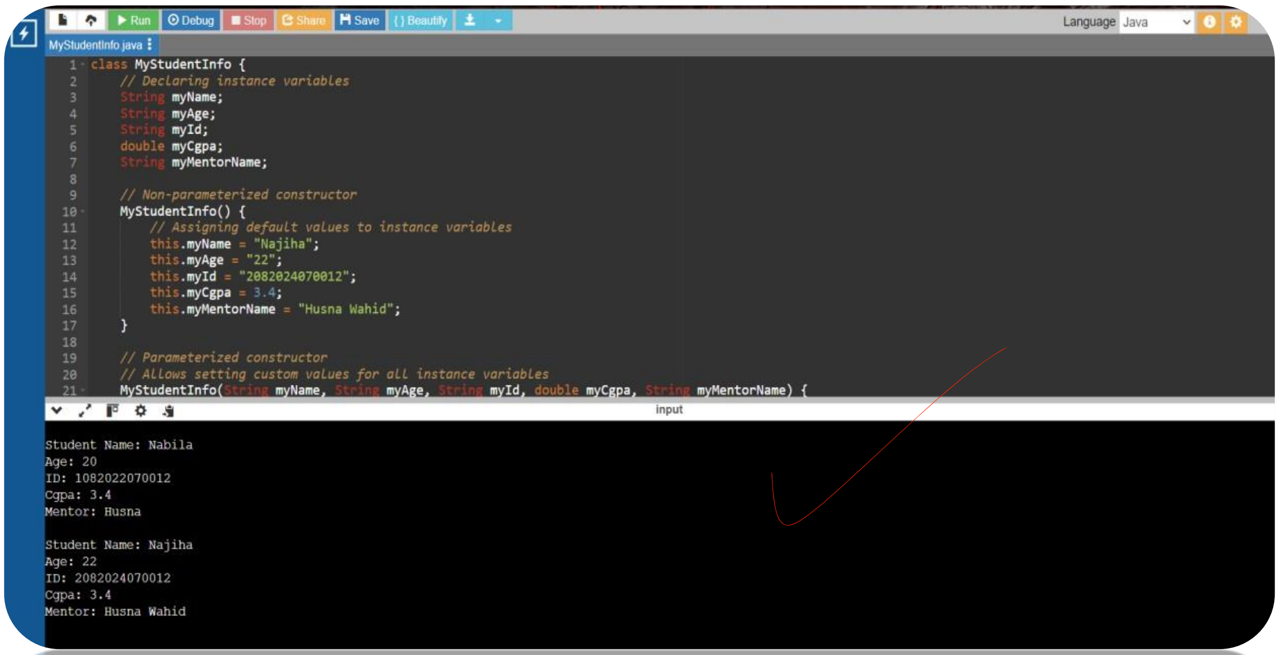
Coding

```
class MyStudentInfo {
    // Declaring instance variables
    String myName;
    String myAge;
    String myId;
    double myCgpa;
    String myMentorName;
```

Only 5

```
    // Non-parameterized constructor
    MyStudentInfo() {
        // Assigning default values to instance variables
        this.myName = "Najiha";
        this.myAge = "22";
        this.myId = "2082024070012";
        this.myCgpa = 3.4;
        this.myMentorName = "Husna Wahid";
    }

    // Parameterized constructor
    // Allows setting custom values for all instance variables
    MyStudentInfo(String myName, String myAge, String myId, double
myCgpa, String myMentorName) {
        this.myName = myName;
        this.myAge = myAge;
        this.myId = myId;
        this.myCgpa = myCgpa;
        this.myMentorName = myMentorName;
    }

    // Method to display student's information
    public void myInfo() {
        // Prints out the student's name, Age, ID, CGPA, and mentor's
name
        System.out.println("\nStudent Name: " + myName
                            + "\nAge: " + myAge
                            + "\nID: " + myId
                            + "\nCgpa: " + myCgpa
                            + "\nMentor: " + myMentorName);
    }

    // Main method - program entry point
    public static void main(String[] args) {
        // Creating an object using the non-parameterized constructor
        MyStudentInfo myObj = new MyStudentInfo();

        // Creating an object using the parameterized constructor with
custom values
        MyStudentInfo myObj1 = new MyStudentInfo("Nabila", "20",
"1082022070012", 3.4, "Husna");

        // Calling myInfo() method to display information of the first
object (custom values)
        myObj1.myInfo();

        // Calling myInfo() method to display information of the
second object (default values)
        myObj.myInfo();
    }
}
```

Output



Language Java

```java
class MyStudentInfo {
    // Declaring instance variables
    String myName;
    String myAge;
    String myId;
    double myCgpa;
    String myMentorName;

    // Non-parameterized constructor
    MyStudentInfo() {
        // Assigning default values to instance variables
        this.myName = "Najiha";
        this.myAge = "22";
        this.myId = "2082024070012";
        this.myCgpa = 3.4;
        this.myMentorName = "Husna Wahid";
    }

    // Parameterized constructor
    // Allows setting custom values for all instance variables
    MyStudentInfo(String myName, String myAge, String myId, double myCgpa, String myMentorName) {
```

```
Student Name: Nabila
Age: 20
ID: 1082022070012
Cgpa: 3.4
Mentor: Husna

Student Name: Najiha
Age: 22
ID: 2082024070012
Cgpa: 3.4
Mentor: Husna Wahid
```

RUBRIC FOR ASSIGNMNET 1

Criteria	Excellent (4)	Good (3)	Satisfactory (2)	Needs Improvement (1)	Marks
Object Design (20%)	Object has at least 6 attributes, well-named, and appropriately typed. Object has at least 4 well-defined and well-documented methods. Proper use of constructor, setters, and getters with clear functionality and documentation.	Object has at least 5 attributes, mostly well-named and appropriately typed. Object has at least 4 well-defined and documented methods. Adequate use of constructor, setters, and getters.	Object has less than 5 attributes, some not well-named or typed. Object has less than 4 methods or lacking documentation. Limited or unclear use of constructor, setters, and getters.	Object design is incomplete or missing significant attributes or methods. Poor or missing use of constructor, setters, and getters.	15
Encapsulation (20%)	Excellent use of encapsulation with clear visibility modifiers and access control. All data is properly encapsulated with no direct access to attributes. Proper use of accessors and mutators.	Good use of encapsulation with mostly clear visibility modifiers and access control. Most data is encapsulated, with minimal direct access to attributes. Adequate use of accessors and mutators.	Some issues with encapsulation and access control, but it generally prevents unauthorized access. Limited use of accessors and mutators.	Poor or no use of encapsulation, allowing direct access to attributes. Lack of accessors and mutators.	15
Main Class Implementation (20%)	Effective and creative use of control structures to manipulate the object's attributes and methods Code is well-organized, easy to read, and free from errors. Exceptional demonstration of programming skills.	Good use of control structures to manipulate the object, with some creativity. Code is mostly organized, readable, and has minor errors. Demonstrates solid programming skills.	Adequate use of control structures, but the manipulation lacks creativity or completeness. Code organization and readability need improvement. Basic demonstration of programming skills.	Incomplete or minimal use of control structures, leading to limited object manipulation. Poor code organization, readability, and numerous errors. Lack of programming skills demonstrated.	20

Creativity and Object Choice (20%)	Highly creative and unique choice of object, showing deep understanding of OOP principles. Innovative use of attributes and methods that go beyond the basic requirements. Demonstrates exceptional creativity.	Creative choice of object, demonstrating an understanding of OOP principles. Some innovative use of attributes and methods. Shows creativity.	Object choice is somewhat conventional, with limited creativity. Attributes and methods are basic and meet minimum requirements. Limited creativity.	Lack of creativity in object choice or generic selection. Attributes and methods are basic and lack creativity. No demonstration of creative thinking.	10
Documentation and Comments (10%)	Code is extensively documented with clear explanations for each attribute, method, and control structure. Comments are used effectively to provide insights into the code's functionality. Excellent documentation.	Code is adequately documented with explanations for most attributes, methods, and control structures. Some comments are used to clarify code functionality. Good documentation.	Limited documentation with explanations for only a few attributes, methods, or control structures. Minimal use of comments. Basic documentation.	Code is poorly or not documented, with little or no explanation of attributes, methods, or control structures. Lack of comments. Inadequate documentation.	10
REMARKS Add minimum six attributes, Use the getter and setters in the main and in the display method. use if or loop structures					80
TOTAL MARKS					

10

END OF QUESTION