NATIONAL UNIVERSITY

551 M.F. JHOCSON ST., SAMPALOC, MANILA

NATIONAL UNIVERSITY
COLLEGE OF ENGINEERING

COMPUTER ENGINEERING
CPOOPG2L: Object Oriented Programming

Name/Section	**Martin, Seth Marcus S.** **COE211**	**Rating**
Date performed	**January 16, 2023**	
Date submitted	**January 22, 2023**	
Professor	**Eng. Marlon Bagara**	

ACTIVITY NO. 6
CLASS

I. OBJECTIVES:

This activity aims to:

1. Discuss the concept of class by drawing figures out of a template.
2. Construct an object out of the class template made.

LEARNING OUTCOMES (LOs) *At the end of the activity, the students should be able to:*	**COURSE LEARNING OUTCOMES (CLOs)**				
	1	2	3	4	5
1. Construct a program that will implement class definition and object		●		●	

COURSE LEARNING OUTCOMES (CLOs)

1. Understand the fundamental concept of OOP through Java programming
2. Write programs using console and dialog box.
3. Apply the concept of iterative, control, and array structure programming..
4. Construct classes, objects, methods and constructor.
5. Write programs in GUI environment

II. SOFTWARE/HARDWARE/EQUIPMENT NEEDED:

1. Eclipse
2. Computer unit

III. SAFETY GUIDELINES:

1. Make sure you have both an adjustable table and chair so that ergonomic accommodations can be made for each person using the computer.
2. The computer screen should be front and centre so neck turning is unnecessary.
3. Keep your lab space clean and organized.
4. Clean your lab bench and equipment, and lock the door before you leave the laboratory.
5. Never eat, drink, or smoke while working in the laboratory.

6. DO NOT TOUCH ANYTHING WITH WHICH YOU ARE NOT COMPLETELY FAMILIAR!!! It is always better to ask questions to laboratory technicians or to your instructors than to risk harm to yourself or damage to the equipment.

IV. THEORY

A ***class*** is a template or blueprint from which objects are created. When you construct an object from a class, you are said to have created an ***instance*** of the class.

The bits of data in an object are called its ***instance fields***, and the procedures that operate on the data are called ***methods***.

Characteristics of an object

1. **The object's behavior** – what can you do on this object, or what method can you apply to it?
2. **The object's state** – how does the object reacts when you invoke those methods?
3. **The object's identity** – how is the object distinguished from others that may have the same behavior or state?

Defining a class

The general form of a class is as follows:

Modifiers$_{opt}$ ***class ClassName***
{
Class body
}

A typical class declaration look like this:

```
class MyClass {
        //attributes
        //constructors
        //methods
}
```

Example:

```
class Point {
 int x;
 int y;
}
```

To create an object, we use the following declaration:
Point p = new Point();

Use the **new** keyword to instantiate a class. The open and close parentheses indicate a method call. Once a class is instantiated, memory will be allocated to the object of the class to hold its data

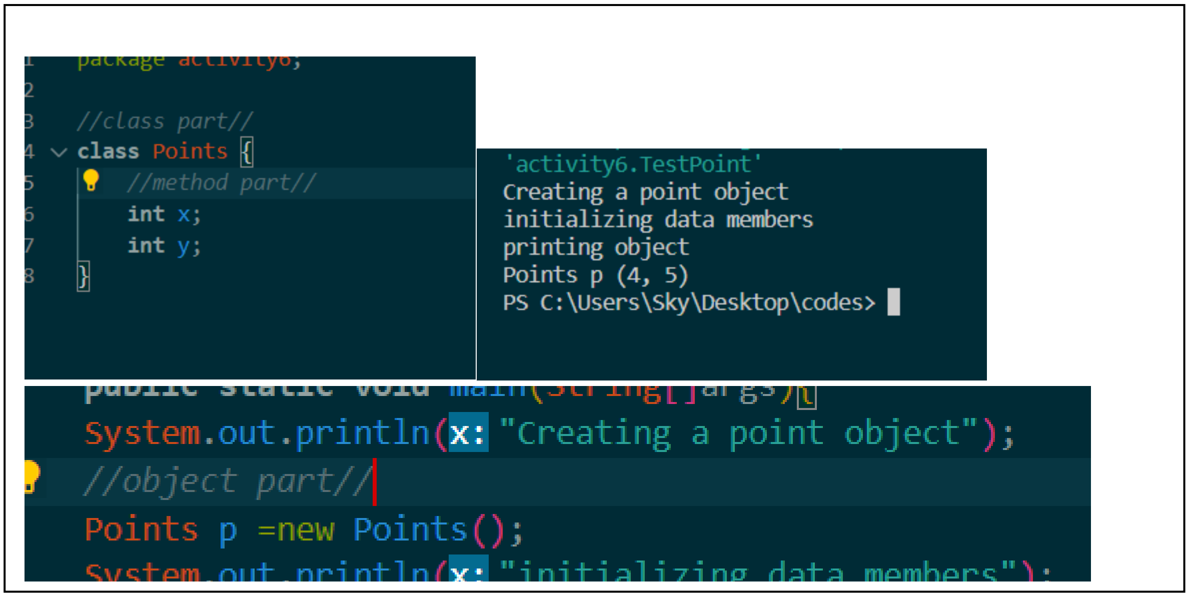
V. PROCEDURE:

1. Write the program below and identify the **class part**, the **object part**, and the **method part**.

Program1:

```
class Point{
I      int x;
       int y;
}
class TestPoint{
       public static void main(String[]args){
       System.out.println(“Creating a point object”);
       Point p =new Point();
       System.out.println(“initializing data members”);
             p.x=4;
             p.y=5;
System.out.println(“printing object”);
System.out.println(“Point p (“ + p.x +”, “ + p.y +”)”);
}
}
```

Output



```
package activity6;

//class part//
class Points {
    //method part//
    int x;
    int y;
}
```

```
'activity6.TestPoint'
Creating a point object
initializing data members
printing object
Points p (4, 5)
PS C:\Users\Sky\Desktop\codes>
```

```
public static void main(String[]args){
System.out.println(x: "Creating a point object");
//object part//
Points p =new Points();
System.out.println(x: "initializing data members");
```

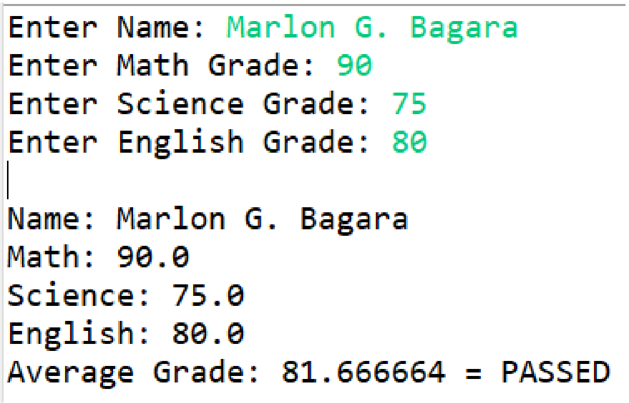
VI. PROBLEMS/QUESTIONS:

1. Construct a class Student with the following fields: Lastname. Firstname,and ID number. Instantiate the class three times. Illustrate in detail the class and object.

2. Construct a class Person with the following fields: Lastname. Firstname, Age, and address. Use the constructor to initialize the fields. Use a method to display the inputted Lastname, Firstname, Age, and address.

3. Use a Package with a name **StudentGrade**. Create an app that computes the student's Grade.

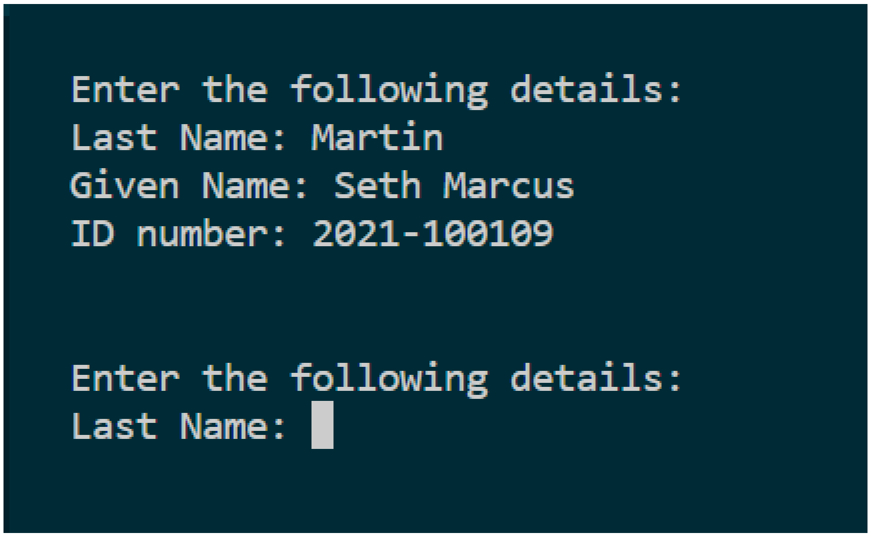
   a. The user will input the following: **Name, Math Grade, Science Grade, and English Grade**
   b. Create a Class called **StudentApp**.
   c. Use constructor to assign data in the fields: name, math, science, and English grades.
   d. Use a method to compute the average grade.
   e. Create a method to display the result, e.g.:



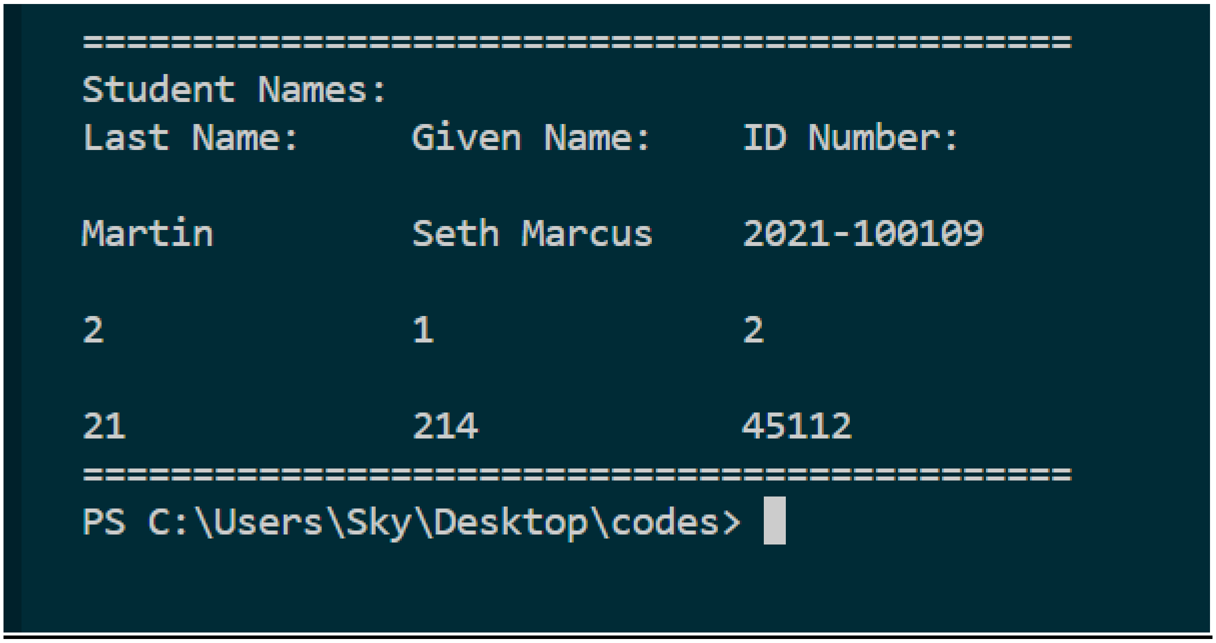
```
Enter Name: Marlon G. Bagara
Enter Math Grade: 90
Enter Science Grade: 75
Enter English Grade: 80

Name: Marlon G. Bagara
Math: 90.0
Science: 75.0
English: 80.0
Average Grade: 81.666664 = PASSED
```

VII. INTERPRETATION/ANALYSIS OF DATA:

The first given question is quite easy as we already discussed how to create attributes and use method and objects to run the code. As there are three attributes in the first question, I was tasked to run it three time using objects. The data that I used instead was inputs as it requires the person to input data instead of already setting data beforehand, making it different every time it was used.

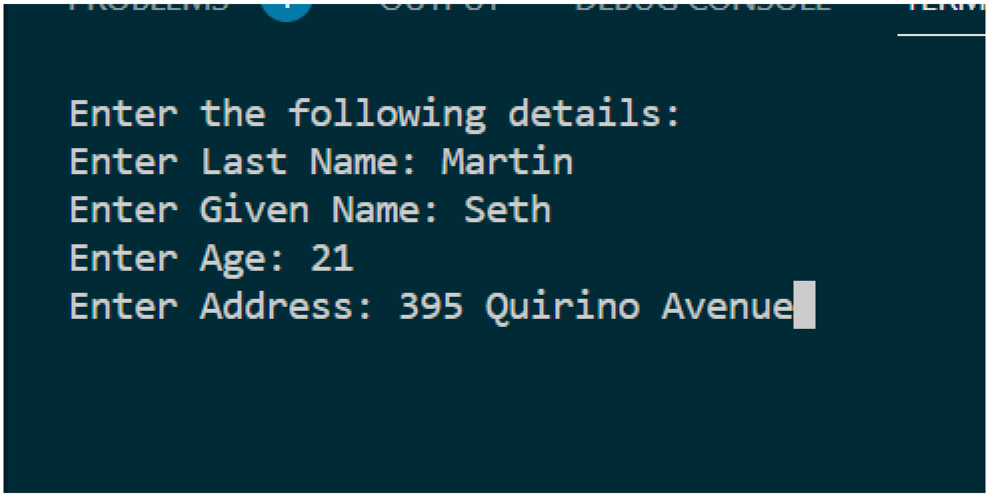


```
Enter the following details:
Last Name: Martin
Given Name: Seth Marcus
ID number: 2021-100109


Enter the following details:
Last Name: 
```

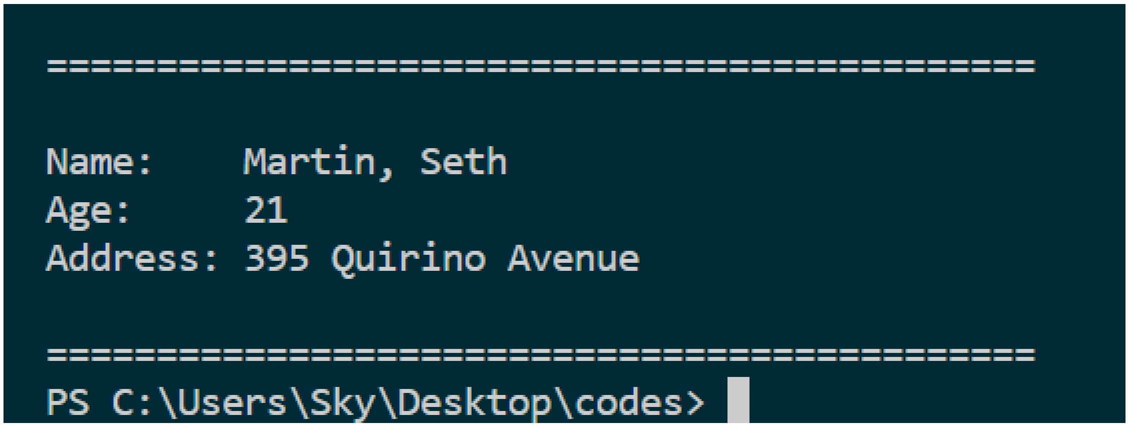


```
=============================================
Student Names:
Last Name:     Given Name:    ID Number:

Martin         Seth Marcus    2021-100109

2              1              2

21             214            45112
=============================================
PS C:\Users\Sky\Desktop\codes> 
```

The second question is like the first one, but instead of ID number, we are asked to provide our address, as it was quite tricky since it asked for integer and strings alternatively, as I try to debug it I found out that using nextLine command it overwrites the nextInt, by adding an extra nextLine command it was fixed. Compared to the first question that only use strings, mixing in an integer made a huge difference, by reading and understanding my own code, I learned more.

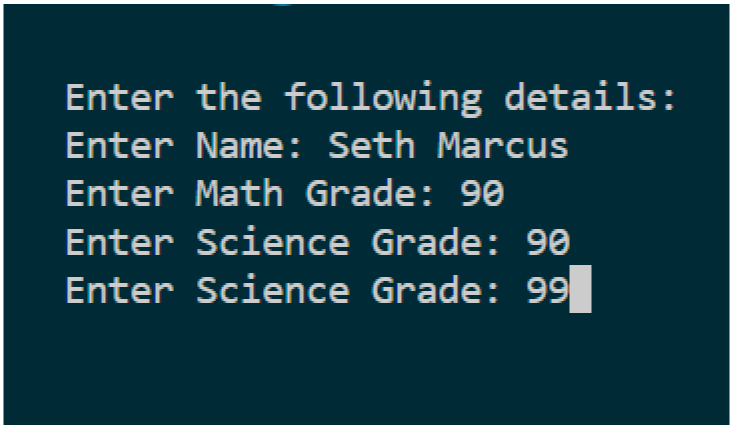


```
Enter the following details:
Enter Last Name: Martin
Enter Given Name: Seth
Enter Age: 21
Enter Address: 395 Quirino Avenue
```

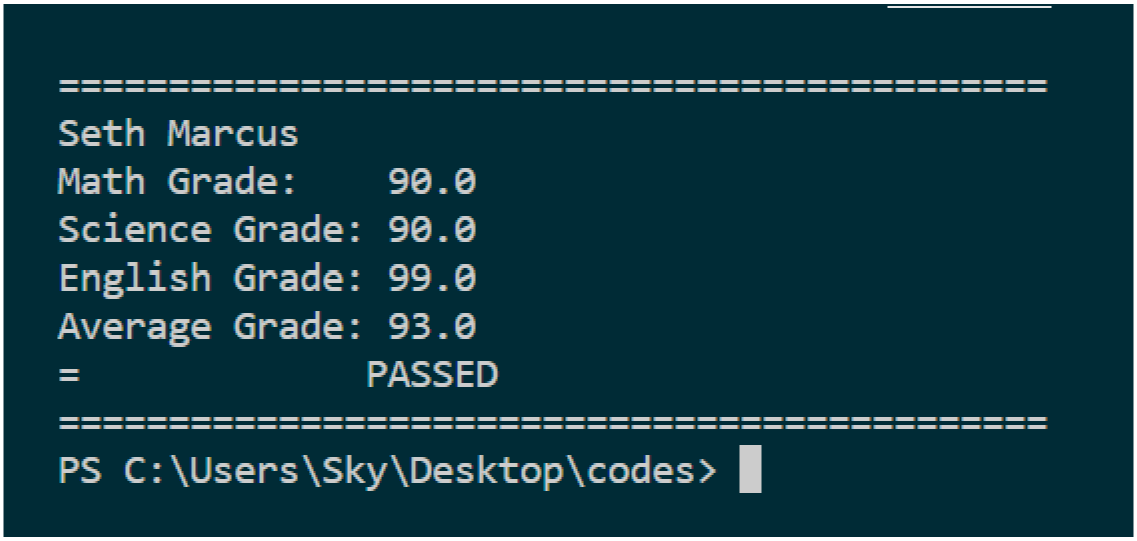


```
=============================================

Name:    Martin, Seth
Age:     21
Address: 395 Quirino Avenue

=============================================
PS C:\Users\Sky\Desktop\codes>
```

As we already discussed how formula works, this is quite easy and with the previous question, I did what I did beforehand and using the knowledge I learned. Throughout the activity, I did it with my own style and using the knowledge where I'm more comfortable with. I also try some few tricks along the way and finding out a quicker and better way of doing things.



```
Enter the following details:
Enter Name: Seth Marcus
Enter Math Grade: 90
Enter Science Grade: 90
Enter Science Grade: 99
```



```
=============================================
Seth Marcus
Math Grade:     90.0
Science Grade: 90.0
English Grade: 99.0
Average Grade: 93.0
=             PASSED
=============================================
PS C:\Users\Sky\Desktop\codes> 
```

VIII. CONCLUSION/RECOMMENDATION:

The activity at hand was more of problem analyzation as it requires you to thoroughly look for improvements and understanding each and every line. The activity also shows how we navigate through two codes that are inside one package, making it more interactive than before. Having two code intersecting to each other making it more easy to read and write codes. Using the idea of constructors as a medium for our objects is more easy than to manually input variables for each objects.

IX. REFERENCES:

Dale, Nell. (2018). Object-oriented data structures using Java.
Litvin, Maria. (2015). Java methods: : Object-oriented programming and data structures. 3rd Ed.
Baesens, Bart. (2015). Beginning java programming: : The object-oriented approach.
Gaddis, Tony. (2013).Starting out with Java: from control structures through objects. 5TH ed. Boston: Pearson