Name:- Amit Podder

ID:- 20-42273-1

Section:- [O]

Object-Oriented Programming 1 (JAVA)

Midterm Assignment

Ans to the ques No:- 1

a) Platform independence means if we write a program or implement things on an operating system or in a machine, we can also execute the same program on another platform or implement the things on another machine without changes.

b) When we compile a java program it creates a **.class** file which is a collection of byte code. This byte code is not machine code and the system only understands machine code. In this case, the JVM (Java Virtual Machine) can understand this byte code and converts this byte code into platform-specific machine code, and in this way, java achieves platform independence.

Ans to the ques No:- 2

a) Type Casting is the process of converting the value of a primitive data type to another primitive data type. There are two types of Type Casting:-

Implicit Type Casting:-

```
public class TypeCasting {
public static void main(String[] args) {
short s = 210;
int i;

i = s;

System.out.println("short to int");
System.out.println("short: " + s);
System.out.println("int: " + i);

    }
 }
```

Here, the java first converts the short data type to int data type. And then it will assign an int variable. It is converting the value of a smaller primitive data type to a larger primitive data type. Hence, there is no loss in data.

Explicit Type Casting:-

```
public class TypeCasting {

public static void main(String[] args) {
```

```
int i = 210;

short s;


s = (short) i ;


System.out.println("int to short");

System.out.println("int: " + i);

System.out.println("short: " + s);


      }

 }
```

Here, the java first converts the int data type to short data type. And then it will assign a short variable. It is converting the value of a larger primitive data type to a smaller primitive data type. Hence, there is a loss of data.

b) We need to provide the alphabet " f " after every float value. **For example:-**

3.5 f

With 3.5 we must use the alphabet " f " to count it as a float value. If we don't use the alphabet " f ", it will count the value as double and it will show an error in the system. That’s why we need to provide the alphabet “ f ” after every float value.

Ans to the ques No:- 3

“ this ” keyword refers to the instance variable (i.e. a member variable). If a local variable name and an instance variable name are the same then **“ this ”** keyword separates them.

Example:-

this.instancevariablename = instancevariablename

Here, because of **“ this ”** keyword it will keep the two variables separate. And in this way, **“ this ”** keyword prevents member variable and local variable name conflict issue.

Ans to the ques No:- 4

a) In class-based object-oriented programming, a constructor is a special method that is used to initialize objects. The constructor is called when an object of a class is created. It can be used to set initial values for object attributes.

b) ***Local Variable:-*** A variable declared inside the body of the method is called a local variable.

Instance Variable:- A variable declared inside the class but outside the body of the method is called an instance variable.

Class Variable:- A variable declared as static but outside of the method is called a class variable.

Example:-

```
 class Account{

        // Instance Variable
        int accountNo;
        double balance;

        // Class Variable
        static double perDayTransactionLimit;

        // Local Variable
        void method addInterest(double rate){
        balance  = balance + (balance * rate / 100);
        }

        void show( ){
               System.out.println("AccountNo: " + accountNo);
               System.out.println("Balance: "+ balance);
         }
     }
```

<u>Ans to the ques No:- 5</u>

<u>a)</u> There are two ways to create a string in java-

1. <u>By string literal:-</u> Java string literal can be created by using double-quotes.
   <u>Example:-</u>
   String s1 = “Hello Java”;
2. <u>By **new** keyword:-</u> Java string can be created by using the **“ new ”** keyword.
   <u>Example:-</u>
   String s2 = new String (“Hello Java”);

<u>b)</u> The String pool is a special storage area in the Java heap. When a string is created and if the string already exists in the pool, the reference of the existing string will be returned. If the string does not exist in the pool, a new string object initializes and is placed in the pool.

<u>Example:-</u>

```
public class StringPool{
public static void main(String[] args) {
       String s1 = "Hello Java";
       String s2 = "Hello Java";
       String s3 = new String("Hello Java");

       System.out.println(s1 == s2);  // true
       System.out.println(s1 == s3);  // false

    }
 }
```

Here, at first, JVM checks if the same value exists in the string constant pool or not. If yes, it occupies the already existing value like string s1 == s2. So, its value is true. But if no, it creates a new string by itself and adds to the string pool using the “ new ” operator like string s1 == s3. So, its value is false.

c) The string is immutable because java uses the string literal concept. Since string literals with the same contents share storage from a common pool, java string is designed to be immutable. Once a string object is created, its contents cannot be modified or changed.

Example:-

```
public class Stringimmutable {

public static void main(String[] args) {


            String s1 = "Hello";
            String s2 = new String("Java");


System.out.println("Using concat: " + s1.concat(s2));


      }
 }
```

Here, Hello has not changed but a new object is created HelloJava. Because of this, the string is immutable for java.

Ans to the ques No:- 6

In Java, a Jagged array is a multidimensional array where member arrays are different in size. These are special types of arrays that can be used to store rows of data of various lengths.

Example:-

```
  public class Array {
  public static void main(String[] args){
   int arr[][] = new int[2][];

  arr[0] = new int[4];
  arr[1] = new int[6];

   int count = 0;
   for (int r=0; r<arr.length; r++)
   for (int c=0; c<arr[r].length; c++)
   arr[r][c] = count++;

  System.out.println("2D Jagged Array");
  for (int r=0; r<arr.length; r++)
  {
    for(int c=0; c<arr[r].length; c++)
    System.out.print(arr[r][c] + " ");
    System.out.println();
   }
 }
}
```

Here, we have created a 2D Jagged Array. The output is-

```
2D Jagged Array
0 1 2 3
4 5 6 7 8 9
```

Here, in the output, the first row of the Jagged array has 4 columns and the second row of the Jagged array has 6 columns. In this way, we can store data of various lengths by using the jagged array.

Ans to the ques No:- 7

a) Object-Oriented Programming (OOP) has four principles. Inheritance is one of the four principles. Inheritance is the process where one class posses the properties of another class. It will help us in re-using the code and it can easily be maintained.

b) Inheritance is the process where one class posses the properties of another class. There are 3 types of inheritance-

1. Single Inheritance:- When a class inherits properties from a single class, then it is called Single Inheritance.

   For Example:-

An apple can inherit the properties of the fruit. So, the fruit is the parent class and the apple is the child class.

2. Multilevel Inheritance:- When a class inherits properties from a class which again has to inherit properties from another class, then it is called Multilevel Inheritance.

   For Example:-

   A Car can inherit the properties of the motor vehicle and the motor vehicle can inherit the properties of the vehicle. So, the car is the subclass or child class of the motor vehicle and the motor vehicle is the child class of vehicle.

3. Hierarchical Inheritance:- When more than one classes inherit the same class, then it is called Hierarchical Inheritance.

   For Example:-

   A student, teacher, and officer can inherit the properties of a person. So, the person is the parent class of the student, teacher, and officer.

Ans to the ques No:- 8

There are many differences between **Encapsulation** and **Inheritance**:-

Encapsulation is the mechanism that binds together code and the data it manipulates. On the other hand,

Inheritance is the process where one class posses the properties of another class.

Encapsulation is used to hide the internal details of a class. But, **Inheritance** is used to create a parent-child relationship.

Encapsulation is a primary attribute of object-oriented programming. Classes use it to manage access to attributes contained within itself. On the other hand, **Inheritance** is used to create new classes that share some of the attributes of existing classes.

Encapsulation keeps the internal details safe from outside interference and misuse which **Inheritance** cannot provide.

Inheritance is known for code reusability. It helps us to re-use the code. On the other hand, **Encapsulation** doesn't provide the re-use of code. It is not one of the ways to re-use the code.

These are some of the differences between **Encapsulation** and **Inheritance**.

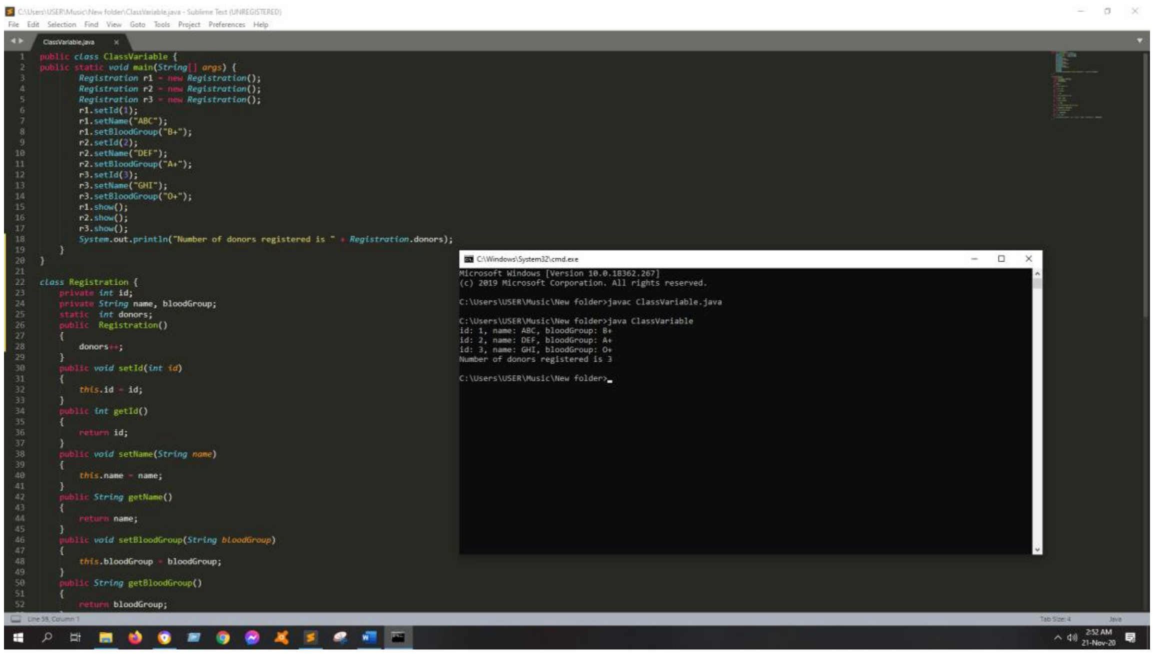
Ans to the ques No:- 9

```
public class ClassVariable {
public static void main(String[] args) {
		Registration r1 = new Registration();
		Registration r2 = new Registration();
		Registration r3 = new Registration();
		r1.setId(1);
		r1.setName("ABC");
		r1.setBloodGroup("B+");
		r2.setId(2);
		r2.setName("DEF");
		r2.setBloodGroup("A+");
		r3.setId(3);
		r3.setName("GHI");
		r3.setBloodGroup("O+");
		r1.show();
		r2.show();
		r3.show();
		System.out.println("Number of donors
registered is " + Registration.donors);
	}
}

class Registration {
	private int id;
	private String name, bloodGroup;
```

```
static  int donors;
public  Registration()
{
	donors++;
}
public void setId(int id)
{
	this.id = id;
}
public int getId()
{
	return id;
}
public void setName(String name)
{
	this.name = name;
}
public String getName()
{
	return name;
}
public void setBloodGroup(String bloodGroup)
{
	this.bloodGroup = bloodGroup;
}
public String getBloodGroup()
{
```

```
                return bloodGroup;
        }
        public void show()
        {
                System.out.println("id: " + id + ", name: " +
name + ", bloodGroup: " + bloodGroup);
        }
}
```



C:\Users\USER\Music\New folder\ClassVariable.java - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
ClassVariable.java
Line 39, Column 1

```
public class ClassVariable {
public static void main(String[] args) {
        Registration r1 = new Registration();
        Registration r2 = new Registration();
        Registration r3 = new Registration();
        r1.setId(1);
        r1.setName("ABC");
        r1.setBloodGroup("B+");
        r2.setId(2);
        r2.setName("DEF");
        r2.setBloodGroup("A+");
        r3.setId(3);
        r3.setName("GHI");
        r3.setBloodGroup("O+");
        r1.show();
        r2.show();
        r3.show();
        System.out.println("Number of donors registered is " + Registration.donors);
    }
}

class Registration {
    private int id;
    private String name, bloodGroup;
    static  int donors;
    public  Registration()
    {
        donors++;
    }
    public void setId(int id)
    {
        this.id = id;
    }
    public int getId()
    {
        return id;
    }
    public void setName(String name)
    {
        this.name = name;
    }
    public String getName()
    {
        return name;
    }
    public void setBloodGroup(String bloodGroup)
    {
        this.bloodGroup = bloodGroup;
    }
    public String getBloodGroup()
    {
        return bloodGroup;
```

```
Microsoft Windows [Version 10.0.18362.267]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\USER\Music\New folder>javac ClassVariable.java

C:\Users\USER\Music\New folder>java ClassVariable
id: 1, name: ABC, bloodGroup: B+
id: 2, name: DEF, bloodGroup: A+
id: 3, name: GHI, bloodGroup: O+
Number of donors registered is 3

C:\Users\USER\Music\New folder>
```

Ans to the ques No:- 10

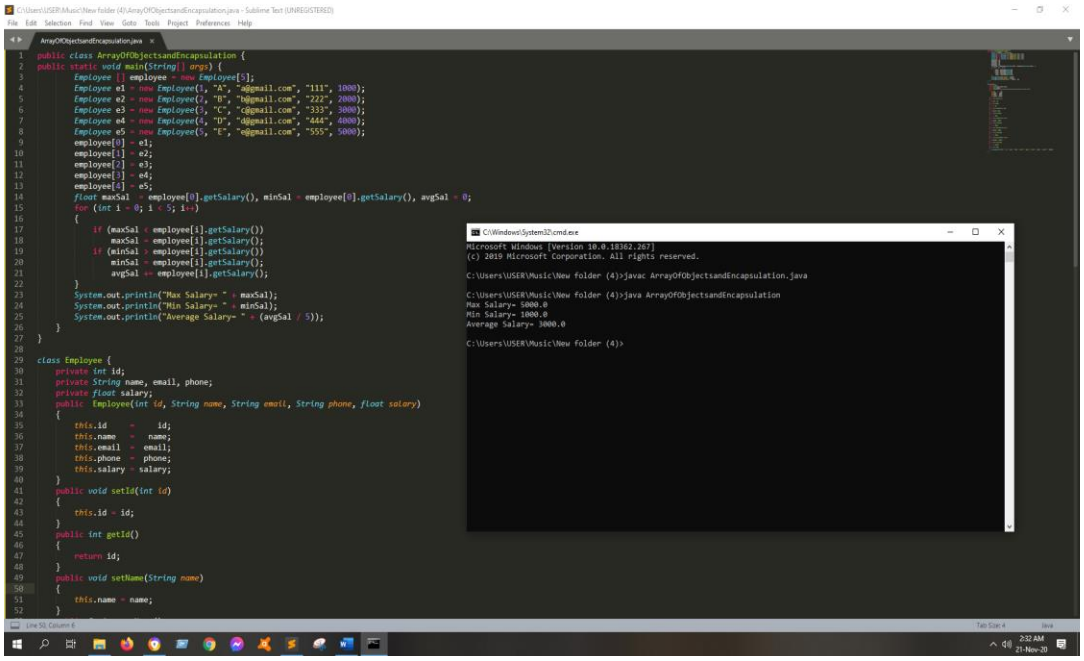
```
public class ArrayOfObjectsandEncapsulation {
public static void main(String[] args) {
		Employee [] employee = new Employee[5];
		Employee e1 = new Employee(1, "A",
"a@gmail.com", "111", 1000);
		Employee e2 = new Employee(2, "B",
"b@gmail.com", "222", 2000);
		Employee e3 = new Employee(3, "C",
"c@gmail.com", "333", 3000);
		Employee e4 = new Employee(4, "D",
"d@gmail.com", "444", 4000);
		Employee e5 = new Employee(5, "E",
"e@gmail.com", "555", 5000);
		employee[0] = e1;
		employee[1] = e2;
		employee[2] = e3;
		employee[3] = e4;
		employee[4] = e5;
		float maxSal  = employee[0].getSalary(),
minSal = employee[0].getSalary(), avgSal = 0;
		for (int i = 0; i < 5; i++)
		{
			if (maxSal < employee[i].getSalary())
				maxSal = employee[i].getSalary();
			if (minSal > employee[i].getSalary())
```

```
                        minSal = employee[i].getSalary();
                    avgSal += employee[i].getSalary();
            }
            System.out.println("Max Salary= " +
maxSal);
            System.out.println("Min Salary= " + minSal);
            System.out.println("Average Salary= " +
(avgSal / 5));
    }
}

class Employee {
    private int id;
    private String name, email, phone;
    private float salary;
    public  Employee(int id, String name, String
email, String phone, float salary)
    {
            this.id     =     id;
            this.name   =   name;
            this.email  =  email;
            this.phone  =  phone;
            this.salary = salary;
    }
    public void setId(int id)
    {
            this.id = id;
```

```
}
public int getId()
{
        return id;
}
public void setName(String name)
{
        this.name = name;
}
public String getName()
{
        return name;
}
public void setEmail(String email)
{
        this.email = email;
}
public String getEmail()
{
        return email;
}
public void setPhone(String phone)
{
        this.phone = phone;
}
public String getPhone()
{
```

```java
			return phone;
		}
		public void setSalary(float salary)
		{
			this.salary = salary;
		}
		public float getSalary()
		{
			return salary;
		}
		public void show()
		{
			System.out.println("id: " + id + ", name: " + name + ", email: " + email + ", phone: " + phone + ", salary: " + salary);
		}
}
```



C:\Users\USER\Music\New folder (4)\ArrayOfObjectsandEncapsulation.java - Sublime Text (UNREGISTERED)

```java
public class ArrayOfObjectsandEncapsulation {
public static void main(String[] args) {
        Employee [] employee = new Employee[5];
        Employee e1 = new Employee(1, "A", "a@gmail.com", "111", 1000);
        Employee e2 = new Employee(2, "B", "b@gmail.com", "222", 2000);
        Employee e3 = new Employee(3, "C", "c@gmail.com", "333", 3000);
        Employee e4 = new Employee(4, "D", "d@gmail.com", "444", 4000);
        Employee e5 = new Employee(5, "E", "e@gmail.com", "555", 5000);
        employee[0] = e1;
        employee[1] = e2;
        employee[2] = e3;
        employee[3] = e4;
        employee[4] = e5;
        float maxSal  = employee[0].getSalary(), minSal = employee[0].getSalary(), avgSal = 0;
        for (int i = 0; i < 5; i++)
        {
            if (maxSal < employee[i].getSalary())
                maxSal = employee[i].getSalary();
            if (minSal > employee[i].getSalary())
                minSal = employee[i].getSalary();
                avgSal += employee[i].getSalary();
        }
        System.out.println("Max Salary= " + maxSal);
        System.out.println("Min Salary= " + minSal);
        System.out.println("Average Salary= " + (avgSal / 5));
    }
}

class Employee {
    private int id;
    private String name, email, phone;
    private float salary;
    public  Employee(int id, String name, String email, String phone, float salary)
    {
        this.id     =     id;
        this.name   =   name;
        this.email  =  email;
        this.phone  =  phone;
        this.salary = salary;
    }
    public void setId(int id)
    {
        this.id = id;
    }
    public int getId()
    {
        return id;
    }
    public void setName(String name)
    {
        this.name = name;
    }
```

C:\Windows\System32\cmd.exe

```
Microsoft Windows [Version 10.0.18362.267]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\USER\Music\New folder (4)>javac ArrayOfObjectsandEncapsulation.java

C:\Users\USER\Music\New folder (4)>java ArrayOfObjectsandEncapsulation
Max Salary= 5000.0
Min Salary= 1000.0
Average Salary= 3000.0

C:\Users\USER\Music\New folder (4)>
```