Assignment 2

Answer:1

Encapsulation

Encapsulation is accomplished when each object maintains a private state, inside a class. Other objects cannot access this state directly, instead, they can only invoke a list of public functions.

Abstraction

Abstraction is an extension of encapsulation. It is the process of selecting data from a larger pool to show only the relevant details to the object. example Suppose you want to create a dating application and you are asked to collect all the information about your users. You might receive the following data from your user: Full name, address, phone number, favorite food, favorite movie, hobbies, tax information, social security number, credit score, you only need to select the information that is pertinent to your dating application from that pool. Data like Full name, favorite food, favorite movie, and hobbies make sense for a dating application. The process of fetching/removing/selecting the user information from a larger pool is referred to as Abstraction.

Inheritance

Inheritance is the ability of one object to acquire some/all properties of another object. For example, Apple is a fruit so assume that we have a class Fruit and a subclass of it named Apple. Our Apple acquires the properties of the Fruit class. Other classifications could be grape, pear, and mango, etc. Fruit defines a class of foods that are mature ovaries of a plant, fleshy, contains a large seed within or numerous tiny seeds. Apple the sub-class acquires these properties from Fruit and has some unique properties, which are different from other sub-classes of Fruit such as red, round, depression at the top.

Polymorphism

Polymorphism gives us a way to use a class exactly like its parent so there is no confusion with mixing types. This being said, each child sub-class keeps its own functions/methods as they are. for example If we had a superclass called Mammal that has a method called mammalSound(). The sub-classes of Mammals could be Dogs, whales, elephants, and horses. Each of these would have their own iteration of a mammal sound (dog-barks, whale-clicks).

Answer:2

CONSTRUCTOR:

A constructor in Java 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:

The constructor’s name must match the class name, and it cannot have a return type (like void) and the constructor is called when the object is created.

All classes have constructors by default: if you do not create a class constructor yourself, Java creates one for you. However, then you are not able to set initial values for object attributes.

Following are the difference between constructor and method.

Abdul Moiz Chishti
BSE-20F-022

Assignment 2

Constructor is used to initialize an object whereas method is used to exhibits functionality of an object.

Constructors are invoked implicitly whereas methods are invoked explicitly.

Constructor does not return any value where the method may/may not return a value.

In case constructor is not present, a default constructor is provided by java compiler. In the case of a method, no default method is provided.

Constructor should be of the same name as that of class. Method name should not be of the same name as that of class.

Answer:3

Encapsulation

Encapsulation is the mechanism of hiding of data implementation by restricting access to public methods. Instance variables are kept private and accessor methods are made public to achieve this.

For example, we are hiding the name and dob attributes of person class in the below code snippet.

Encapsulation — private instance variable and public accessor methods.

```
public class Employee {
   private String name;
   private Date dob;
   public String getName() {
      return name;
   }
   public void setName(String name) {
      this.name = name;
   }
   public Date getDob() {
      return dob;
   }
   public void setDob(Date dob) {
      this.dob = dob;
   }
}
```

Abdul Moiz Chishti
BSE-20F-022

Assignment 2

Abstraction

Abstract means a concept or an Idea which is not associated with any particular instance. Using abstract class/Interface we express the intent of the class rather than the actual implementation. In a way, one class should not know the inner details of another in order to use it, just knowing the interfaces should be good enough.

Inheritance

Inheritances expresses “is-a” and/or “has-a” relationship between two objects. Using Inheritance, In derived classes we can reuse the code of existing super classes. In Java, concept of “is-a” is based on class inheritance (using extends) or interface implementation (using implements).

For example, FileInputStream "is-a" InputStream that reads from a file.

Polymorphism

It means one name many forms. It is further of two types — static and dynamic. Static polymorphism is achieved using method overloading and dynamic polymorphism using method overriding. It is closely related to inheritance. We can write a code that works on the superclass, and it will work with any subclass type as well.

Class is a blueprint or a set of instructions to build a specific type of object. It is a basic concept of Object-Oriented Programming which revolve around the real-life entities. Class in Java determines how an object will behave and what the object will contain.

Syntax

Object is an instance of a class. An object in OOPS is nothing but a self-contained component which consists of methods and properties to make a particular type of data useful. For example, color name, table, bag, barking. When you send a message to an object, you are asking the object to invoke or execute one of its methods as defined in the class.

From a programming point of view, an object in OOPS can include a data structure, a variable, or a function. It has a memory location allocated. Java Objects are designed as class hierarchies.

Answer:4

```
package assignment.pkg2;
public class Main_Class {
    public static void main(String[] args) {
        Shapes ob = new Shapes();
        Shapes ob1 = new Shapes(1,2);

        System.out.println("Volume of Sphere  " + ob.volume());
        System.out.println("Volume of Cylinder  " + ob.volume(1));
        System.out.println("Surface Area of Cylinder  " + ob.volume(1, 2));

        System.out.println("Volume of Sphere  " + ob1.volume());
        System.out.println("Volume of Cylinder  " + ob1.volume(1));
        System.out.println("Surface Area of Cylinder  " + ob1.volume(1 , 2));
    }
```

Abdul Moiz Chishti
BSE-20F-022

Assignment 2

```
}
package assignment.pkg2;
import java.util.Scanner;
import java.lang.Math;
public class Shapes {
    int radius;
    int height;
    Scanner input = new Scanner(System.in);
    Shapes (){
        radius = 10;
        height = 10;
    }
    Shapes (int a , int b){
        System.out.print("Enter radius : ");
        a = input.nextInt();
        System.out.print("Enter height : ");
        b = input.nextInt();
        radius = a;
        height = b;
    }

    double volume (){
        double v_of_s;
        v_of_s = ((4/3) * (Math.PI) * radius);
        return v_of_s;
    }

    double volume (double v_of_c){
        v_of_c = (Math.PI) * (Math.pow(radius , 2)) * (height);
        return v_of_c;
    }

    double volume(double c , double d){
        double sa_of_c;
        sa_of_c = (2 * Math.PI * Math.pow(radius , 2)) +
              (2 * Math.PI * radius * height);
        return sa_of_c;

    }

}
```

Abdul Moiz Chishti
BSE-20F-022