NAME:HAMID KHAN

ROLL NUMBER:15730

SECTION:A

SEMESTER:2

SOFTWARE ENGINEERING

………………………………………………………………………………….

Q1.ANSWER) ACCESSS MODIFIERS: the access modifiers in java specifies the accessibility or scope of a field ,method,constructor,or class.we can change the access level of fields,constructors,methods,and class by applying the access modifier on it.a member has default accessibility when no accessibility modifier is specified.

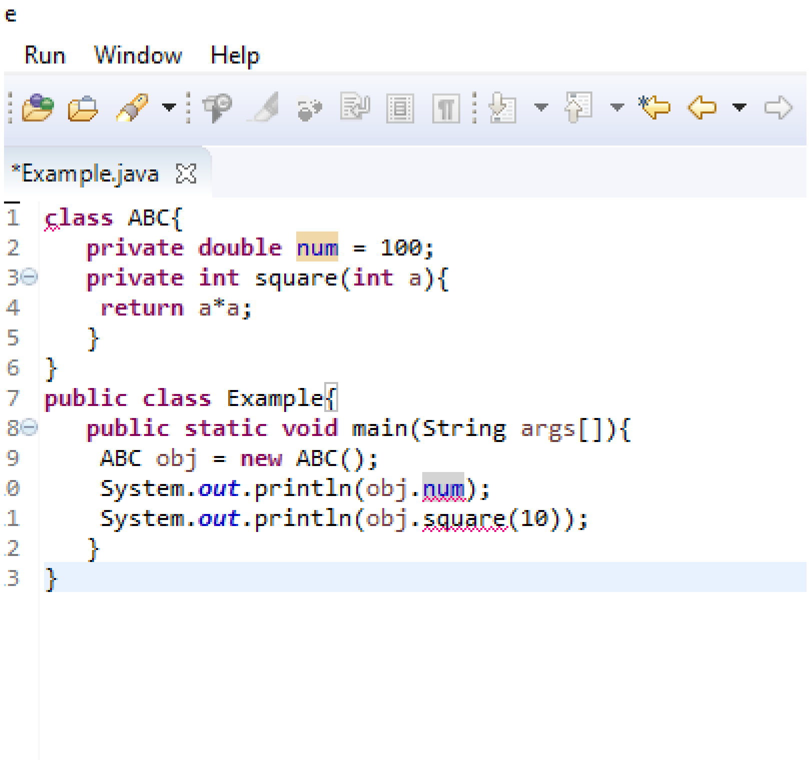
PRIVATE: Private: The private access modifier is specified using the keyword private. The methods or data members declared as private are accessible only within the class in which they are declared. Any other class of same package will not be able to access these members. Top level Classes or interface can not be declared as private because private means . only visible within the enclosing class. protected means only visible within the enclosing class and any subclasses Hence these modifiers in terms of application to classes, they apply only to nested classes and not on top level classes

Default: When no access modifier is specified for a class , method or data member It is said to be having the default access modifier by default.

- The data members, class or methods which are not declared using any access modifiers , having default access modifier are accessible only within the same package.

B ANSWER)private access modifier code example:

This example shows compile error because we are trying to access the private data member and method of class ABC in the class Example. The private data member and method are only accessible within the class.

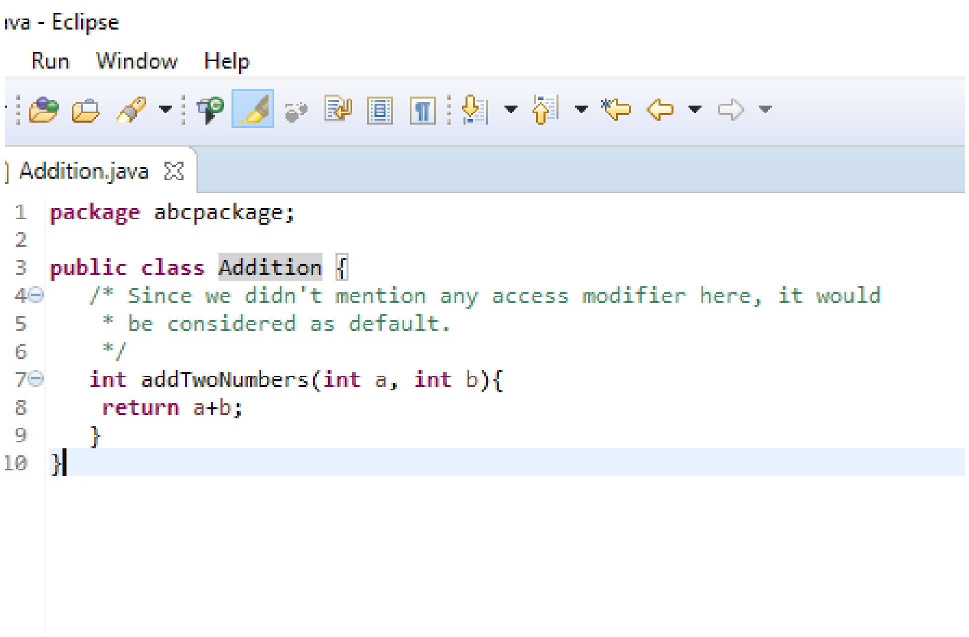


```java
class ABC{
   private double num = 100;
   private int square(int a){
    return a*a;
   }
}
public class Example{
   public static void main(String args[]){
    ABC obj = new ABC();
    System.out.println(obj.num);
    System.out.println(obj.square(10));
   }
}
```

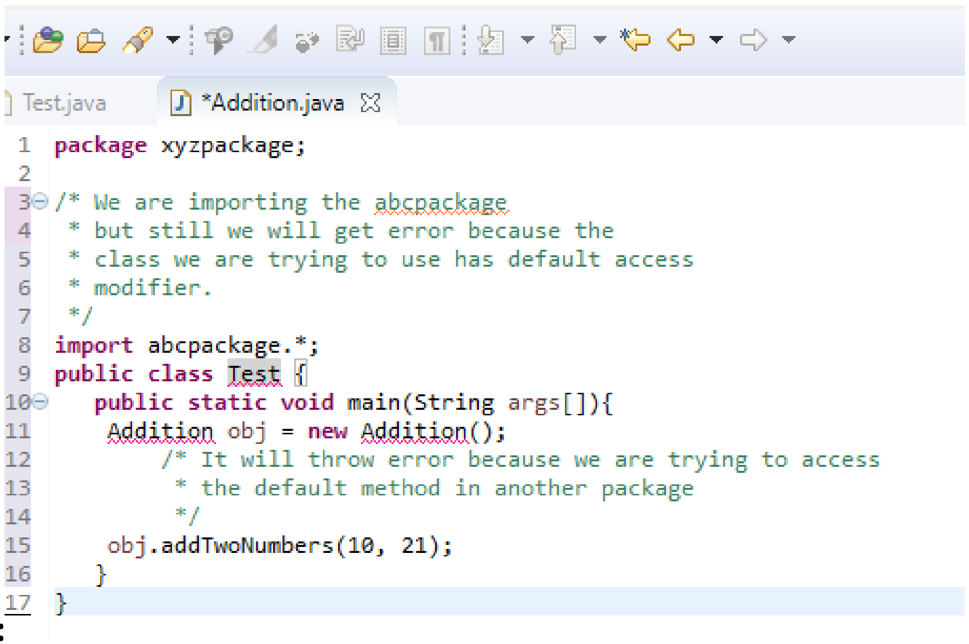
```
Compile - time error
```

DEFAULT ACCESS MODIFIER CODE EXAMPLE: In this example we have two classes, Test class is trying to access the default method of Addition class, since class Test belongs to a different package, this program would throw compilation error, because the scope of default modifier is limited to the same package in which it is declared.

ADDITION.JAVA:



```java
package abcpackage;

public class Addition {
   /* Since we didn't mention any access modifier here, it would
    * be considered as default.
    */
   int addTwoNumbers(int a, int b){
    return a+b;
   }
}
```



```java
package xyzpackage;

/* We are importing the abcpackage
 * but still we will get error because the
 * class we are trying to use has default access
 * modifier.
 */
import abcpackage.*;
public class Test {
   public static void main(String args[]){
    Addition obj = new Addition();
        /* It will throw error because we are trying to access
         * the default method in another package
         */
    obj.addTwoNumbers(10, 21);
   }
}
```

TEST.JAVA:

```
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
The method addTwoNumbers(int, int) from the type Addition is not visible
at xyzpackage.Test.main(Test.java:12)
```

OUTPUT:

……

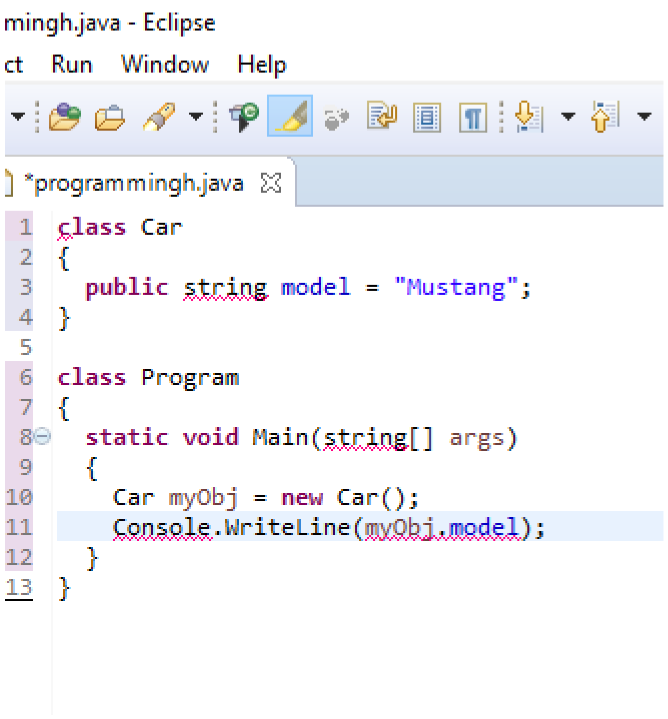
Q2.ANSWER)public access modifiers: The public access modifier is specified using the keyword public

- The public access modifier has the wideset scope among all other access modifiers.
- Classes, methods or data members which are declared as public are accessible from every where in the program. There is no restriction on the scope of a public data members.

PROTECTED ACCESS MODIFIERS: The protected access modifier is specified using the keyword protected.

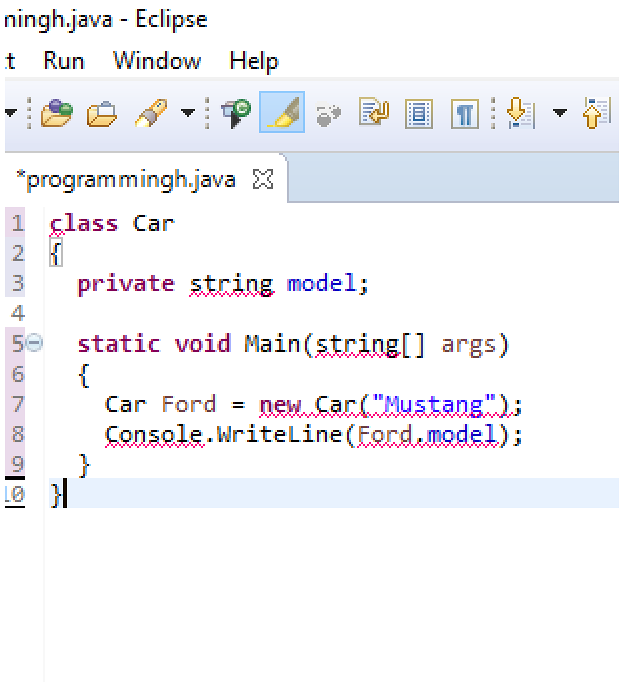
- The methods or data members declared as protected are accessible within same package or sub classes.

B ANSWER)PUBLIC CODE: If you declare a field with a `public` access modifier, it is accessible for all classes:



```
class Car
{
  public string model = "Mustang";
}

class Program
{
  static void Main(string[] args)
  {
    Car myObj = new Car();
    Console.WriteLine(myObj.model);
  }
}
```

PROTECTED ACCESS CODE: If you declare a field with a private access modifier, it can only be accessed within the same class.but if you try to access it outside the class and error will occur.

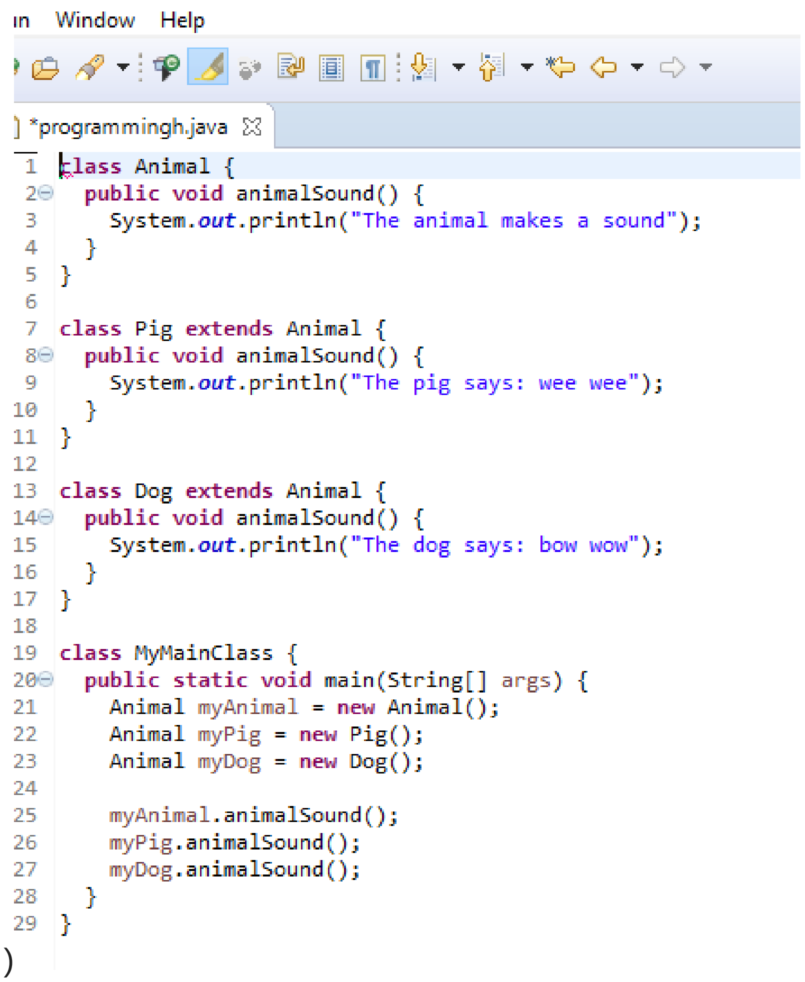


```
class Car
{
  private string model;

  static void Main(string[] args)
  {
    Car Ford = new Car("Mustang");
    Console.WriteLine(Ford.model);
  }
}
```

Q3.ANSWER) INHERITANCE: Inheritance is an important pillar of OOP(Object Oriented Programming). It is the mechanism in java by which one class is allow to **i**nherit the features(fields and methods) of another class. ... The subclass can add its own fields and methods in addition to the superclass fields and methods

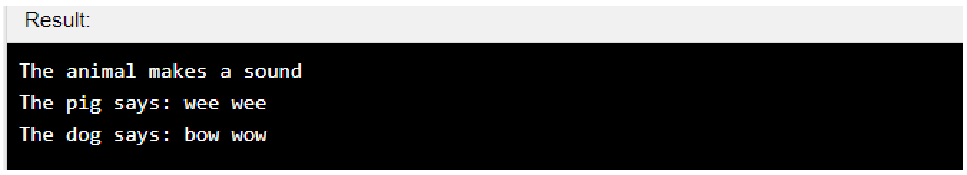
PURPOSE: The primary purpose of inheritance is to reuse code from an existing class. Inheritance allows you to create a new class that starts off by including all data and implementation details of the base class. ... You can also use inheritance to modify the behavior of an existing class

B ANSWER)



```java
class Animal {
  public void animalSound() {
    System.out.println("The animal makes a sound");
  }
}

class Pig extends Animal {
  public void animalSound() {
    System.out.println("The pig says: wee wee");
  }
}

class Dog extends Animal {
  public void animalSound() {
    System.out.println("The dog says: bow wow");
  }
}

class MyMainClass {
  public static void main(String[] args) {
    Animal myAnimal = new Animal();
    Animal myPig = new Pig();
    Animal myDog = new Dog();

    myAnimal.animalSound();
    myPig.animalSound();
    myDog.animalSound();
  }
}
```

)


Result:

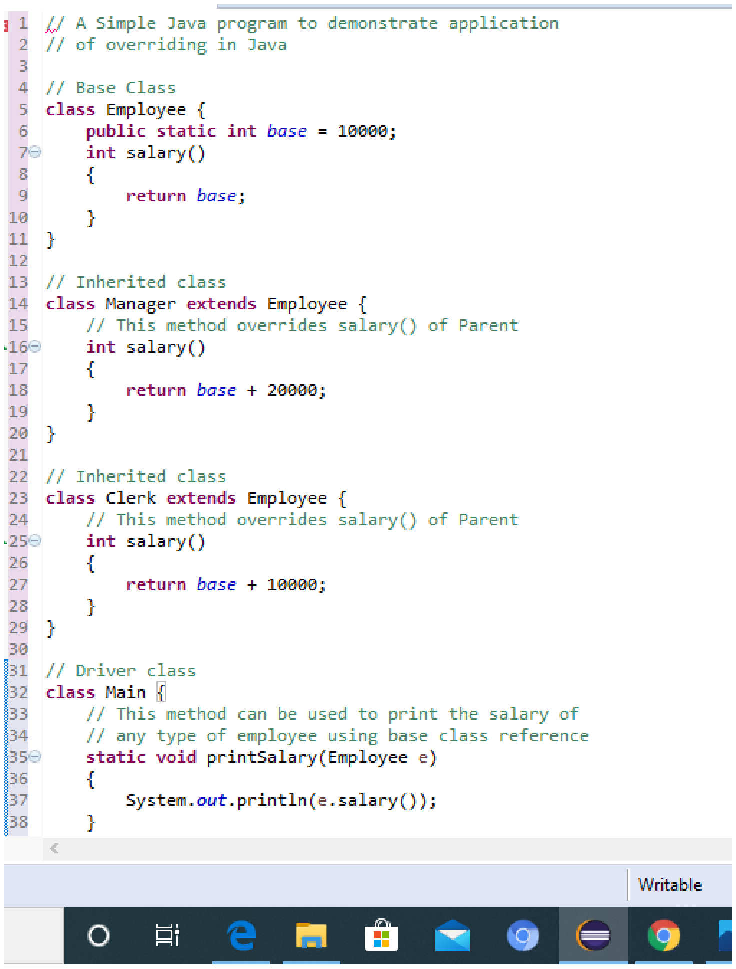
```
The animal makes a sound
The pig says: wee wee
The dog says: bow wow
```

Q4ANSWER)

POLYMORPHISM: Polymorphism is the ability of an object to take on many forms. The most common use of polymorphism in OOP occurs when a parent class reference is used to refer to a child class object. Any Java object that can pass more than one IS-A test is considered to be polymorphic. ... A reference variable can be of only one type

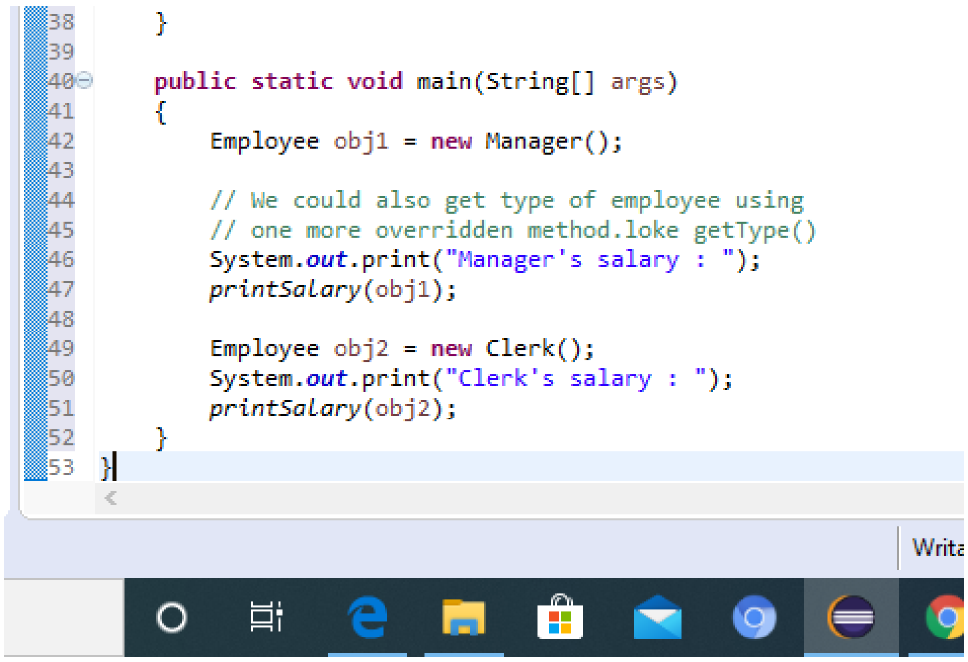
PURPOSE: Polymorphism is considered as one of the important features of Object Oriented Programming. Polymorphism allows us to perform a single action in different ways. In other words, polymorphism allows you to define one interface and have multiple implementations.

BANSWER)

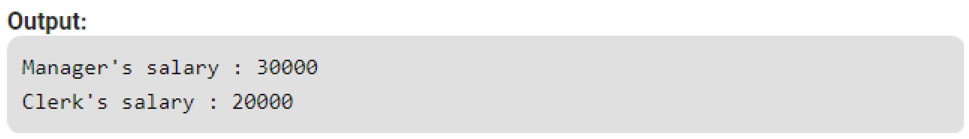


```java
// A Simple Java program to demonstrate application
// of overriding in Java

// Base Class
class Employee {
    public static int base = 10000;
    int salary()
    {
        return base;
    }
}

// Inherited class
class Manager extends Employee {
    // This method overrides salary() of Parent
    int salary()
    {
        return base + 20000;
    }
}

// Inherited class
class Clerk extends Employee {
    // This method overrides salary() of Parent
    int salary()
    {
        return base + 10000;
    }
}

// Driver class
class Main {
    // This method can be used to print the salary of
    // any type of employee using base class reference
    static void printSalary(Employee e)
    {
        System.out.println(e.salary());
    }
```

Writable



```java
    }

    public static void main(String[] args)
    {
        Employee obj1 = new Manager();

        // We could also get type of employee using
        // one more overridden method.loke getType()
        System.out.print("Manager's salary : ");
        printSalary(obj1);

        Employee obj2 = new Clerk();
        System.out.print("Clerk's salary : ");
        printSalary(obj2);
    }
}
```



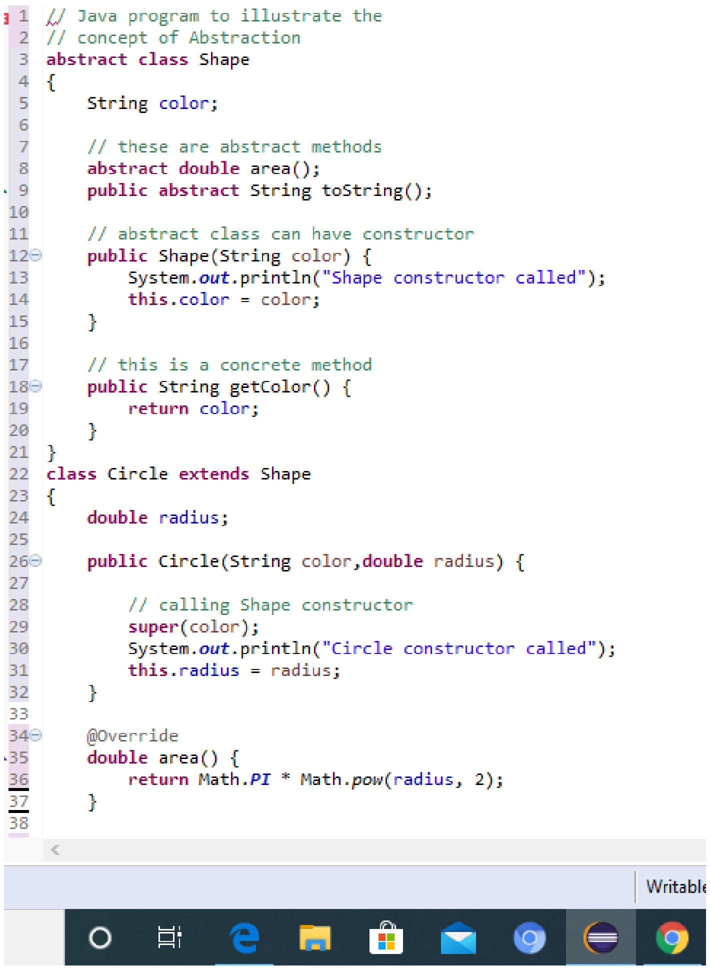
Output:

```
Manager's salary : 30000
Clerk's salary : 20000
```

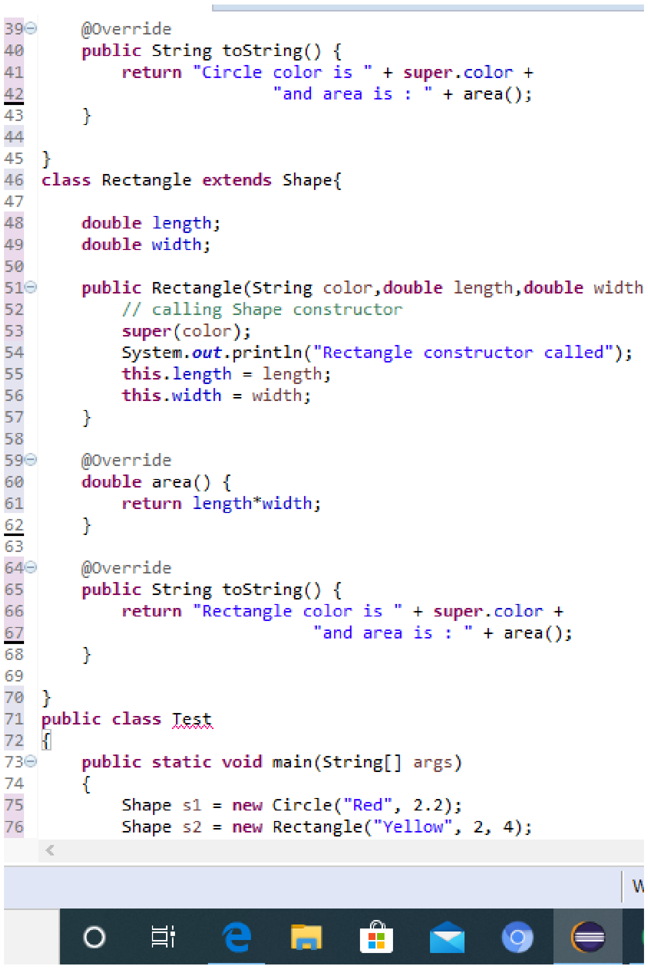
Q5.ANSWER)ABSTRACTION: Abstraction is selecting data from a larger pool to show only the relevant details of the object to the user. Abstraction "shows" only the essential attributes and "hides" unnecessary information. It helps to reduce programming complexity and effort. It is one of the most important concepts of OOPs.

example of a man driving a car. The man only knows that pressing the accelerators will increase the speed of car or applying brakes will stop the car but he does not know about how on pressing the accelerator the speed is actually increasing, he does not know about the inner mechanism of the car or the implementation of accelerator, brakes in the car.this is abstraction

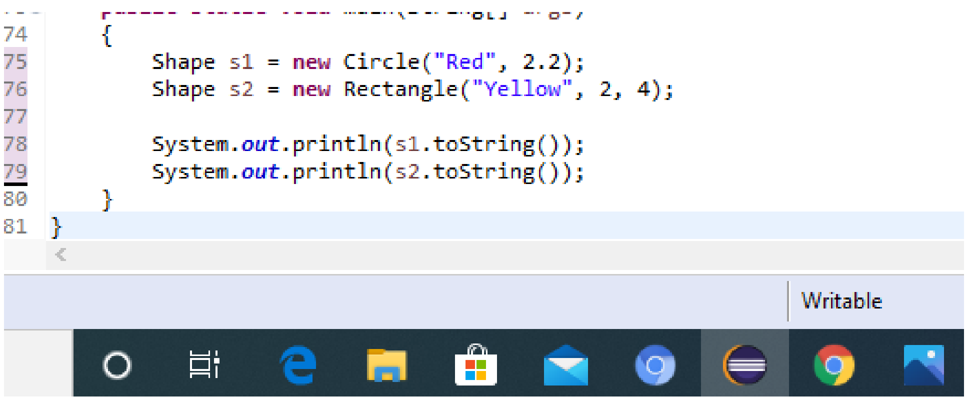
B ANSWER)



```
// Java program to illustrate the
// concept of Abstraction
abstract class Shape
{
    String color;

    // these are abstract methods
    abstract double area();
    public abstract String toString();

    // abstract class can have constructor
    public Shape(String color) {
        System.out.println("Shape constructor called");
        this.color = color;
    }

    // this is a concrete method
    public String getColor() {
        return color;
    }
}
class Circle extends Shape
{
    double radius;

    public Circle(String color,double radius) {

        // calling Shape constructor
        super(color);
        System.out.println("Circle constructor called");
        this.radius = radius;
    }

    @Override
    double area() {
        return Math.PI * Math.pow(radius, 2);
    }

```



```java
    @Override
    public String toString() {
        return "Circle color is " + super.color +
                       "and area is : " + area();
    }

}
class Rectangle extends Shape{

    double length;
    double width;

    public Rectangle(String color,double length,double width
        // calling Shape constructor
        super(color);
        System.out.println("Rectangle constructor called");
        this.length = length;
        this.width = width;
    }

    @Override
    double area() {
        return length*width;
    }

    @Override
    public String toString() {
        return "Rectangle color is " + super.color +
                       "and area is : " + area();
    }

}
public class Test
{
    public static void main(String[] args)
    {
        Shape s1 = new Circle("Red", 2.2);
        Shape s2 = new Rectangle("Yellow", 2, 4);
```



```java
    {
        Shape s1 = new Circle("Red", 2.2);
        Shape s2 = new Rectangle("Yellow", 2, 4);

        System.out.println(s1.toString());
        System.out.println(s2.toString());
    }
}
```

Writable

```
Shape constructor called
Circle constructor called
Shape constructor called
Rectangle constructor called
Circle color is Redand area is : 15.205308443374602
Rectangle color is Yellowand area is : 8.0
```