Name of the course: Object Oriented Programming with Java	Course Code: 20CP204T
Program: B. Tech. Branch: CE	Semester: 3rd Academic Year: 2022-23

Tutorial- Find Output/Error

Q-1:

```
import java.util.Scanner;

class Point
{
   float x,y,z;
}

class test
{
   public static void main(String[] args)
   {
      Scanner sc = new Scanner(System.in);
      Point P[];
      int n;

      System.out.println("How many Points do you want: ");
      n = sc.nextInt();

      P=new Point[n];

      for ( int i = 0 ; i < n ; i++ )
      {
         System.out.println("Enter x,y and z for "+(i+1)+"th Point:");
         P[i].x= sc.nextFloat();
         P[i].y= sc.nextFloat();
         P[i].z= sc.nextFloat();
      }
   }
}
```

Q-2

```
class Automobile
{
   private String drive()
   {
        return "Driving vehicle";
    }
}
class Car extends Automobile
{
    protected String drive()
    {
        return "Driving car";
    }
}

public class test extends Car
{
    public final String drive()
    {
        return "Driving Electric car";
    }
    public static void main(String[] args)
    {
        final Car car = new test();
        System.out.println(car.drive());
    }
}
```

Q-3

```
class Super
{
    int i=15;
}
class Sub extends Super
{
    int i=10;
}
public class test
{
    public static void main(String[] args)
    {
        Super s1 = new Sub();

        System.out.println(s1.i);
    }
}
```

Q-4

```
abstract class Car
{
   static
   {
      System.out.print("1");
   }
   public Car(String name)
   {
      super();
      System.out.print("2");
   }
   {
      System.out.print("3");
   }
}
public class BlueCar extends Car
{
   {
      System.out.print("4");
   }
   public BlueCar()
   {
      super("blue");
      System.out.print("5");
   }
   public static void main(String[] args)
   {
      new BlueCar();
   }
}
```

Q-5

```
public class test
{
   public void print(Integer i)
   {
      System.out.println("Integer");
   }
   public void print(int i)
   {
      System.out.println("int");
   }
   public void print(long i)
```

```
    {
        System.out.println("long");
    }

    public static void main(String[] args)
    {
        test T1=new test();

        T1.print(10);
    }
}
```

Q-6:

```
class A
{
    public A(String s)
    {
        System.out.print("A");
    }
}

public class B extends A
{
    public B(String s)
    {
        System.out.print("B");
    }
    public static void main(String[] args)
    {
        new B("C");
        System.out.println(" ");
    }
}
```

Q-7:

```
class A
{

}

class B extends A
{

}

class C extends B
```

```
{

}

public class MainClass
{
   static void overloadedMethod(A a)
   {
      System.out.println("ONE");
   }

   static void overloadedMethod(B b)
   {
      System.out.println("TWO");
   }

   static void overloadedMethod(Object obj)
   {
      System.out.println("THREE");
   }

   public static void main(String[] args)
   {
      C c = new C();

      overloadedMethod(c);
   }
}
```

Q-8:

```
public class P
{
static void m1()
{
 System.out.println("Class P");
 }
}
public class Q extends P
{
static void m1()
{
 System.out.println("Class Q");
 }
}
```

Q-9:

```
public class Test{
 public static void main(String[] args){
  System.out.println("main method");
 }
 public static void main(String args){
  System.out.println("Overloaded main method");
 }
}
```

Q-10:

```
class X
{
   public X(int i)
   {
      System.out.println(1);
   }
}

class Y extends X
{
   public Y()
   {
      System.out.println(2);
   }
}
```

Q-11:

```
class Test
{
   public static void main (String[] args)
   {
      int arr1[] = {1, 2, 3};
      int arr2[] = {1, 2, 3};
      if (arr1 == arr2)
         System.out.println("Same");
      else
         System.out.println("Not same");
   }
}
```

Q-12:

```
package inheritancePractice;
class P {
   int a = 30;
}
class Q extends P {
    int a = 50;
}
public class Test extends Q {
    public static void main(String[] args) {
        Q q = new Q();
        System.out.println(" Value of a: " +q.a);
        P p = new Q();
        System.out.println("Value of a: " +p.a);
    }
}
```

Q-13:

```
final class Complex {
    private  double re,  im;
    public Complex(double re, double im) {
        this.re = re;
        this.im = im;
    }
    Complex(Complex c)
    {
      System.out.println("Copy constructor called");
      re = c.re;
      im = c.im;
    }
    public String toString() {
        return "(" + re + " + " + im + "i)";
    }
}
class Main {
    public static void main(String[] args) {
        Complex c1 = new Complex(10, 15);
        Complex c2 = new Complex(c1);
        Complex c3 = c1;
        System.out.println(c2);
    }
}
```

Q-14:

```
public class A {
            public static void main(String[] args)
            {
                        System.out.println('j' + 'a' + 'v' + 'a');
            }
            }
```

Q-15:

```
class demo
```

```
{
   int a, b;
   demo()
   {
      a = 10;
      b = 20;
   }

   public void print()
   {
      System.out.println ("a = " + a + " b = " + b + "n");
   }
}
class Test
{

   public static void main(String[] args)
   {
      demo obj1 = new demo();
      demo obj2 = obj1
      obj1.a += 1;
      obj1.b += 1;
       System.out.println ("Values of obj1 : ");
      obj1.print();
      System.out.println ("Values of obj2 : ");
      obj2.print();

   }
}
```

<u>Q-16:</u>

```
// Find num1 & num2
public class Main
{
   static int findNum1(int a, int b, int c){
      int num1 = a;
      boolean b1 = (num1<b) && ((num1=b)>0);
      b1 = (num1<c) && ((num1=c)>0);
      return num1;
   }
   static int findNum2(int a, int b, int c){
      int num2 = a;
      boolean b1 = (num2>b) && ((num2=b)>0);
      b1 = (num2>c) && ((num2=c)>0);
      return num2;
   }
```

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

                System.out.println("num1: "+findNum1(11,-16,12));
                System.out.println("num2: "+findNum2(11,-16,12));
        }
}
```

Q-17:

```
public class Code
{
   public static void main(String args[])
   {
      int y = 08;
      y = y + 2;
      System.out.println(y);
   }
}
```

Q-18:

```
class Exercise1b {
public static void main(String [] args) {
        int x = 1;
        while ( x < 10 ) {
                if ( x > 3) {
                        System.out.println("big x");
                }
        }
}
}
```

Q-19:

```
public static void main(String [] args) {
        int x = 5;
        while ( x > 1 ) {
                 x = x - 1;
                if ( x < 3) {
                 System.out.println("small x");
                }
        }
}
```

Q-20.

```
class TapeDeck {


   boolean canRecord = false;
   void playTape() {
     System.out.println("tape playing");
   }


   void recordTape() {
     System.out.println("tape recording");
   }
 }


 class TapeDeckTestDrive {
   public static void main(String [] args) {


     t.canRecord = true;
     t.playTape();


     if (t.canRecord == true) {
       t.recordTape();

     }
   }
 }
```

Q-21.

```
class DVDPlayer {


   boolean canRecord = false;


   void recordDVD() {
     System.out.println("DVD recording");
   }
 }


 class DVDPlayerTestDrive {
   public static void main(String [] args) {


     DVDPlayer d = new DVDPlayer();
     d.canRecord = true;
```

```
      d.playDVD();


      if (d.canRecord == true) {
        d.recordDVD();

      }
    }
  }
```

Q-22.

```
class Books {
    String title;
    String author;
  }


  class BooksTestDrive {
    public static void main(String [] args) {


      Books [] myBooks = new Books[3];
      int x = 0;
      myBooks[0].title = "The Grapes of Java";
      myBooks[1].title = "The Java Gatsby";
      myBooks[2].title = "The Java Cookbook";
      myBooks[0].author = "bob";
      myBooks[1].author = "sue";
      myBooks[2].author = "ian";


      while (x < 3) {
        System.out.print(myBooks[x].title);
        System.out.print(" by ");
        System.out.println(myBooks[x].author);
        x = x + 1;
      }
    }
  }
```

Q-23.

What is the output of the following code snippet?

```
            int five = 5;
            int two = 2;
```

```
int total = five + (five > 6 ? ++two : --two);
```

Q-24:

```
public static void main(String... args) {
  String car, bus = "petrol";
  car = car + bus;
  System.out.println(car);
}
```

Options:

a. petrol
b. petrolpetrol
c. compilation error
d. runtime error

Q-25.

```
class A
{
        public A(String s)
        {
                System.out.print("A");
        }
}

public class B extends A
{
        public B(String s)
        {
                System.out.print("B");
        }
        public static void main(String[] args)
        {
                new B("C");
                System.out.println(" ");
        }
}
```

Q-26.

```
class Clidder
{
    private final void flipper()
    {
            System.out.println("Clidder");
    }
}

public class Clidlet extends Clidder
```

```
{
     public final void flipper()
     {
              System.out.println("Clidlet");
     }
     public static void main(String[] args)
     {
              new Clidlet().flipper();
     }
}
```

Q-27:

Will this code compile successfully? If yes, what is output? If no, identify the errors.

```
package pack1;
public class A
{
  private int x = 50;
  protected int y = 100;
   int z = 200;
}
package pack2;
import pack1.A;
public class B extends A {

}
import pack2.B;
public class Test {
public static void main(String[] args)
{
  B b = new B();
  System.out.println(b.x);

  System.out.println(b.y);
  System.out.println(b.z);
 }
}
```

Q-28:

```
class Base {
   public void show() {
      System.out.println("Base::show() called");
   }
}

class Derived extends Base {
   public void show() {
      System.out.println("Derived::show() called");
   }
}

public class Main {
   public static void main(String[] args) {
       Base b = new Derived();
       b.show();
   }
}
```

Q-29:

```
package overridingPrograms;
public class X
{
void draw(int a, float b) throws Throwable
{
 System.out.println("Circle");
 }
}
public class Y extends X
{
@Override
void draw(int a, float b)
{
 System.out.println("Rectangle");
 }
}
public class Z extends Y
{
@Override
void draw(int a, float b) throws ArithmeticException
{
 System.out.println("Square");
 }
}
public class Test
{
public static void main(String[] args) throws Throwable
{
 X x = new Y();
 x.draw(20, 30.5f);
 Y y = (Y)x;
 y.draw(10,2.9f);
 Z z = (Z)y;
 z.draw(20, 30f);
 }
}
```

Q-30:

```
class Automobile {
   private String drive() {
      return "Driving vehicle";
   }
}

class Car extends Automobile {
   protected String drive() {
      return "Driving car";
   }
}

public class ElectricCar extends Car {

   @Override
   public final String drive() {
      return "Driving electric car";
   }

   public static void main(String[] wheels) {
      final Car car = new ElectricCar();
      System.out.print(car.drive());
   }
}
```

A. Driving vehicle
B. Driving electric car
C. Driving car
D. The code does not compile

Q-31:

```
class Building {
   Building() {
      System.out.println("pdeu's-Building");
   }

   Building(String name) {
      this();
      System.out.println("pdeu's-building: String Constructor" + name);
   }
}

public class House extends Building {
   House() {
      System.out.println("pdeu's-House ");
```

```
    }

    House(String name) {
        this();
        System.out.println("pdeu's-house: String Constructor" + name);
    }

    public static void main(String[] args) {
        new House(" pdeu");
    }
}
```

Q-32:

```
        class Test
        {
            final int MAXIMUM = m1();

        private int m1()
            {
                System.out.println(MAXIMUM);
                return 1500;
            }

        public static void main(String[] args)
            {
                Test t = new Test();

                System.out.println(t.MAXIMUM);
            }
        }
```

a) Compilation error
b) Runtime error
c) 0
1500
d) 1500
1500

Q-33:

```
class Test {
public static void main(String[] args)
    {
        int arr[] = { 1, 2, 3 };

        // final with for-each statement
        for (final int i : arr)
```

```
            System.out.print(i + " ");
    }
}
```
a) Compilation error
b) Runtime error
c) 1 2 3

Q-34:

```
class Test {
public
    static void main(String[] args)
    {
        int x = 20;
        System.out.println(x);
    }
    static
    {
        int x = 10;
        System.out.print(x + " ");
    }
}
```
Option
A) 10 20
B) 20 10
C) 10 10
D) 20 20

Q-35:

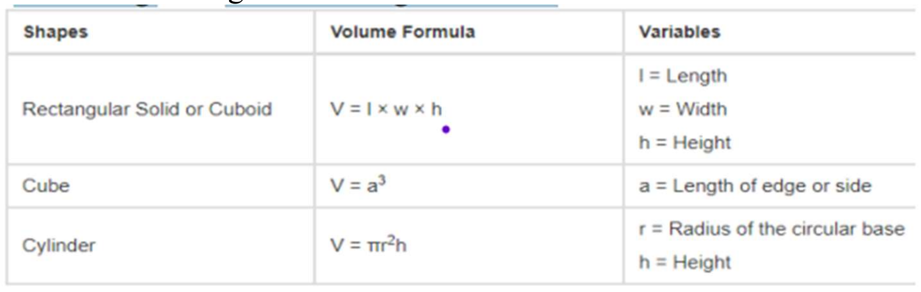
```
public class Test {
    public static void main(String[] args) {
        method(null);
    }

    public static void method(Object o) {
        System.out.println("Object method");
    }

    public static void method(String s) {
        System.out.println("String method");
    }
}
```

PROGRAMS:

1) Create a class Person using constructors that has a single variable age. Such that when the object person1 is created, it gets initialized to default age 20 and when person2 is created the user input his choice of age.
2) Write a program to print the area of two rectangles having sides (4,5) and (5,8) respectively by creating a class named 'Rectangle' with a method named 'Area' which returns the area and length and breadth passed as parameters to its constructor.Construct a class to find the volume of a cuboid, cube and cylinder using the concept of overloading. The formulas are given below:



Shapes	Volume Formula	Variables
Rectangular Solid or Cuboid	$V = l \times w \times h$	l = Length w = Width h = Height
Cube	$V = a^3$	a = Length of edge or side
Cylinder	$V = \pi r^2 h$	r = Radius of the circular base h = Height