SYBSC IT Semester IV Core Java Practical

Practical 1

a) **Write a program to create a class and implement a default, overloaded and copy constructor.**

```
class MyClass
{
private int a;
public MyClass()
{
        System.out.println("Default Constructor");
}
public MyClass(int value)
{
        a=value;
        System.out.println("Parameterized Constructor and value is :"+a);
}
public MyClass(MyClass other)
{
        a=other.a;
        System.out.println("Copy Constructor and value is :"+a);
}
}
Public class Pr
{
Public static void main(String[] args)
{
MyClass obj1=new MyClass();
MyClass obj2=new MyClass(7);
MyClass obj3=new MyClass(obj2);

}
}
```

b) **Write a program to create a class and implement the concepts of Method Overloading**

```
class OperOver
{
public int add(int a, int b)
{
        return a+b;
}
public int add(int a, int b,int c)
{
        return a+b+c;
}
```

```
}
public class Prb
{
public static void main(String[] args)
{
OperOver obj=new OperOver ();
int sum1=obj.add(5,10);
int sum2=obj.add(5,10,15);

System.out.println("Sum of two integers is :"+sum1);

System.out.println("Sum of three integers is :"+sum2);

}
}
```

c) Write a program to create a class and implement the concepts of Static Methods

```
class DemoStaticMethods
{
public static int add(int a, int b)
{
        return a+b;
}
Public static  int subtract(int a, int b)
{
        return a-b;
}
}
public class Prc
{
public static void main(String[] args)
{
int sum= DemoStaticMethods.add(8,4);
int diff= DemoStaticMethods.subtract(7,6);

System.out.println("Sum is :"+sum);

System.out.println("Difference is :"+diff);

}
}
```

Practical 2

a) **Write a program to create a class and implement the concepts of Inheritance and Method overriding.**

```
class A
{
void show()
{
        System.out.println("Base Class");
}
}
class B extends A
{
void show()
{
        System.out.println("Derived Class");
}
}
public class Pr2a
{
public static void main(String[] args)
{
B s=new B();
s.show();
}
}
```

b) **Write a program to create a class and implement the concepts of Abstract classes and methods.**

```
abstract class shape
{
public abstract double area();
}
}
class circle extends shape
{
private double radius;
public circle(double radius)
{
        This.radius=radius;
}
Public double area()
```

```
{
Return Math.PI*radius*radius;
}
}
public class Pr2ab
{
public static void main(String[] args)
{
circle c=new circle(10.0);
System.out.println("Circle Area:"+c.area());
}
}
```

c) **Write a program to create a class and implement the concepts of Intefaces.**

```
interface shape
{
double area();
double perimeter();
}
}
class circle implements shape
{
private double radius;
public circle(double radius)
{
        this.radius=radius;
}
public double area()
{
return Math.PI*radius*radius;
}
public double perimeter()
{
return 2*Math.PI*radius;
}

}
public class Pr2c
{
public static void main(String[] args)
{
circle c=new circle(10.0);
System.out.println("Circle Area:"+c.area());
```

```
        System.out.println("Circle Perimeter:"+c.perimeter());
        }
}
```

Practical 3

a) Write a program to raise built-in exception and raise them As per the requirements.

```
public class Pr3a
{
public static void main(String[] args)
{
Try
{
int result=divide(10,0);
System.out.println("Result:"+result);
}
catch(ArithmeticException e)
{
System.out.println("Error:Division by zero");
}
public static int divide(int a,int b)
{
return a/b;
}
}
```

b) Write a program to defined exceptions and raise them as per the requirements.

```
class cuException extends Exception
{
public cuException(Strin message)
{
Super(message);
}
}

public class Pr3b
{
public static void main(String[] args)
{
```

```
        try
        {
        int age=-20;
        if(age<0)
        {
        throw new cuException("Age cannot be negative");
        }
        System.out.println("Age:"+age);
        }
        catch(ArithmeticException e)
        {
        System.out.println("Error:"+e.getMessage());
        }
        }
}
```

Practical 4

Write a java application to demonstrate multiple bouncing balls of different colors using threads.

```
import java.util.*;
import java.awt.*;
import java.util.ArrayList;
import java.util.List;
import java.util.Random;
public class BouncingBall extends JPanel implements Runnable
{
public static final int WIDTH=800;
public static final int HEIGHT=600;
public static final int NUM_BALLS=5;

private List<Ball>balls;
public BouncingBall()
{
```

```
Balls=new ArrayList<>();
Random r=new Random();
For(int i=0;i<NUM_BALLS;i++)
{
int x=random.nextInt(WIDTH);
int x=random.nextInt(HEIGHT);
int xspeed=random.nextInt(5)+1;
int yspeed=random.nextInt(5)+1;
Color color=new Color(random.nextInt(256), random.nextInt(256), random.nextInt(256));
balls.add(new Ball(x,y,xSpeed,ySpeed.color));
}
}
public void run()
{
while(true)
{
for(Ball ball:balls)
{
ball.move();
}
repaint();
try
{
Thread.sleep(10);
}
catch(InterruptedException e)
{
e.printStackTrace();
}
}
```

```
}
protected void paintComponent(Graphics g)
{
super.paintComponent(g)
for(Ball ball:balls)
{
ball.draw(g);
}
}
public static void main(String args[])
{
JFrame frame=new JFrame("5 Colours Bouncing Balls");
BouncingBall bb=new BouncingBall();
frame.add(bb);
frame.setSize(WIDTH,HEIGHT);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
frame.setVisible(true);

Thread thread=new Thread(bb);
Thread.start();
}
}
```

Practical 5

a) **Create Swing application that randomly changes color on button click.**

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.Random;
public class ChangeColor extends JFrame
{
Private JPanel cp;
Private JButton ccb;
```

```
public ChangeColor()
{
setTitle("Random Color Changer");
setSize(300,200);
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setLayout(new BorderLayout());

cp=new JPanel();
ccb=new JButton("Change Color");
add(cp.BorderLayout.CENTER);
add(ccb.BorderLayout.SOUTH);

ccb.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e)
changeColor();
}
});
}
private void changeColor()
Random r=new Random();
Color rc=new Color(random.nextInt(256), random.nextInt(256), random.nextInt(256));
Cp.setBackground(rc);
}

public static void main(String args[])
{
SwingUtilities.invokeLater(()->{
ChangeColor app=new ChangeColor()
app.setVisible(true);
});
}
}
```

Practical 6

Write a program to Demonstrate the following Events

a) ActionEvent

```
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class BtnClkDemo
{
```

```
public static void main(String args[])
{
JFrame f=new JFrame(“Button Click Demo”);
f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

JButton b=new JButton(“Click me”);
b.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e)
{
JOptionPane.showMessageDialog(f,”Button Clicked”)
}
});
f.setContentPane().add(button);
f.pack();
f.setVisible(true);
}
}
```

b) ActionEvent with Menu Item

```
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class MenuClkDemo
{
public static void main(String args[])
{
JFrame f=new JFrame(“Button Click Demo”);
f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

JMenuBar mb=new JMenuBar();
JMenu fm=new JMenu(“File”);
JMenuItem=ot=new JMenuItem(“open”);

ot.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e)
{
JOptionPane.showMessageDialog(f,”File->open clicked”)
}
});
fm.add(ot);
Mb.add(fm);
g.setJMenuBar(mb);
f.setVisible(true);
f.setSize(400,300);
```

```
}
}
```

c) KeyEvent: Program to demonstrate how to handle KeyEvents

```
import javax.swing.*;
import java.awt.event.*;

public class Allketevents
{
public static void main(String args[])
{
JFrame f=new JFrame("Button Click Demo");
f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

JTextField tf=new JTextField(20);
f.add(tf);

tf.addKeyListener(new KeyAdapter() {
public void keyTyped(KeyEvent e)
{
System.out.println("Key Typed:"+e.getKeyChar());
}

public void keyPressed(KeyEvent e)
{
System.out.println("Key Pressed:"+KeyEvent.getKeyText(e.getKeyCode()));
}

public void keyReleased(KeyEvent e)
{
System.out.println("Key Released:"+KeyEvent.getKeyText(e.getKeyCode()));
}
});
f.pack();
f.setVisible(true);
f.setSize(400,300);
}
}
```