OOJ

George Abraham
1BM19CS197

Part-A

1. Exception : Exception indicates conditions that a reasonable application might try to catch.

Exception Hierarchy:
All exception and errors types are sub classes of class Throwable, which is base class of hierarchy. One branch is headed by Exception. This class is used for exceptional conditions that user programs should catch. Null Pointer Exception is an example of such an exception. Another branch, Error are used by the Java run-time system (JVM) to indicates errors having to do with the run time environment itself (JRE)
Stack OverFlow Error is an example of such an error.

Part B

2a
```
public class Multi_Catch
{
  public static void main (String args[])
  {
     int array[] = {20, 10, 30};
     int num1 = 15, num2 = 0;
     int res = 0;
  try
  { res = num1/num2;
    System.out.println("The result is" +res);
      for (int ct = 2; ct >= 0 ; ct--)
      {
        System.out.println("The value of array are"
                                    + array[ct]);
```

```
    }
  }
  catch (ArrayIndexOutOfBoundsException e)
  {
    System.out.println("Error ? . Array is out of Bounds");
  }
  catch (ArithmeticException e)
  {
    System.out.println("Can't be divided by Zero");
  }
  }
}
```

2b

2c

```
class Calculator
int add(int a, int b)
{
    return a+b;
}
int sub(int a, int b)
{
    return a-b;
}
}
public class AdvancedCalculator
extends Calculator {
    int mult (int a, int b)
    {
        return a*b;
    }
    int div (int a, int b)
    {
        return a/b
    }
    public static void main (String args[])
    {
    ~~AdvancedCalculator cal = new AdvancedCalculator~~
AdvancedCalculator cal = new AdvancedCalculator();
        System.out.println(cal.add(1,2));
        System.out.println(cal.sub(1,2));
        System.out.println(cal.mult(1,2));
        System.out.println(cal.div(1,2));
    }
}
```

Part C

3a.

```
import Java.util.Scanner
abstract class Shape
{
  int a;
  int b;
  abstract void printArea();

  public Shape (int a, int b)
  {
    this.a = a;
    this.b = b;
  }
  }
class Rectangle extends Shape
{
  public Rectangle (int a, int b) {
    super (a,b);
  }
  Void printArea ()
  {
  System.out.println ("Area of rectangle:" + (ab));
    }
    }
class Triangle extends Shape
 {
  public Triangle (int a ,int b ) {
    super (a,b);
  }
void print Area () {
  System.out.println ("Area of triangle: "+
      ( 1/2*a*b)); }
  }
```

```
class Prog5 {

public static void main (String[] args) {
  Scanner sc = new Scanner (System.in);
  System.out.print ("Enter lenght of ~~strngt~~ rectangle:");
int l_rec = sc.nextInt();
  System.out.print ("Enter breadth of rectangle :");
     int b_rec = sc.nextInt();
  System.out.print ("Enter height of triangle :");
    int l_tri = sc.nextInt();
    System.out.print ("Enter base of triangle :");
  int b_tri = sc.nextInt();
~~System.out.print ("Enter radius of~~
    Rectangle r = new
    Rectangle (l_rec, b_rec);
     Triangle t = new Triangle (l_tri, b_tri);
     r.printArea();
     t.printArea();
     ~~c.printArea();~~
  }
}
```

4a

```
interface stackOp
{
   void push(int item);
   int pop();
}
class fixedStack implements stackop
{
   private int stk[];
   private int tos;
   fixed Stack (int size)
  {   stk = new int[size];
      tos = -1;
  }
```

```
}
public void push (int item)
{
  if (tos == stk.lenght - 1)
{
System.out.println ("Stack Overflows");
        int t [] = new
int [stk.lenght * 2];
              for (int i=0; i<stk.lenght; i++)
      t[i] = stk[i];
    stk = t;
    stk [++tos] = item;
}
  else
    stk [++tos] = item;
}
  public int pop ()
{
    if (tos < 0)
    {
System.out.println ("Stack UnderFlow.");
      return 0;
      }
       else
       return stk [tos--];
}
}
class StackTest
{
  Public static void main (String args [])
{
  Fixed Stack fs = new Fixed Stack (3);
Dyn Stack ds = new Dyn Stack (5);
StackOp mystk;
```

```
for (int i = 0; i < 3; i++)
    fs.push(i);
System.out.println("Fixed lenght Stack contents");
for( int i = 0; i < 3; i++)
    System.out.println(fs.pop());
for (int i = 0; i < 7; i++)
    ds.push(i);
System.out.println("Dynamic lenght Stack
    contents are");
for (int i = 0; i < 7; i++)
System.out.println(ds.pop());
    MyStk = fs;
    for (int i = 0; i < 3; i++)
    MyStk.push(i);
System.out.println("Fixed length");
    for int i = 0; i < 3; i++)
```