**Task#01**

* **Create a parent class “Account” with three variables num, title and bal with two functions withdraw and deposit.**
* **Create a child class “SavingAccount” with a function calculateProfit (that applies 7.5% profit on current balance).**
* **Finally create the class “TestApp” that creates the object of Saving account and call**

**withdraw, deposit functions.**

* **Override withdraw function in SavingAccount to deduct Rs. 250 if the balance after withdrawal of amount is less than 10000.**

|  |  |  |
| --- | --- | --- |
| **Account.java** | **SavingAccount.java** | **TestApp.java** |
| public class Account  {  public String acct\_num;  public String acct\_title;  public double acct\_bal;  public Account(){  }  public void withdraw (double amt)  {  if (acct\_bal > amt)  acct\_bal -= amt;  }  public void deposit (double amt)  {  acct\_bal += amt;  }  } | public class SavingAccount extends Account {  public SavingAccount(){  }  public void withdraw (double amt)  {  if (acct\_bal > amt)  acct\_bal -= amt;  if (acct\_bal < 10000)  acct\_bal -= 250;  }  public double calculateProfit ()  {  double profit = acct\_bal \* 0.075;  acct\_bal += profit;  return profit;  }  } | public class TestApp  {  public TestApp() {  }  public static void main(String args[])  {  SavingAccount s = new SavingAccount ();  s.acct\_num = “01-01-999888”;  s.acct\_title = “MyAcctTitle”  s.acct\_bal = 10000.0;  s.withdraw (2000.0);  System.out.println ("Balance="+s.acct\_bal);  }  } |

**Task#02**

**Create an interface for two functions for add and multiply. Create a class Calculator with two class level variables, a parameterized overloaded constructor and four functions for add, subtract, divide and multiply. Create a class TestApp with a main function and create object of Calculator interface to call functions.**

|  |  |  |
| --- | --- | --- |
| **CalcIntf.java** | **CalculatorImpl.java** | **TestApp.java** |
| public interface Calintfr {  public int sum(int a,int b);  public int mul(int a,int b);  //public int div(int a, int b);  //public int sub(int a, int b);  } | public class calimpl implements Calintfr {  int x, y;  public calimpl() {  }  @Override  public int sum(int a, int b)  {  return a+b;  }  @Override  public int mul(int a, int b)  {  return a\*b;  }  @Override  public int div(int a, int b)  {  return a/b;  }  @Override  public int sub(int a, int b)  {  return a-b;  }  } | public class test {  public test(){  }  public static void main(String args[])  {  /\*calimpl obj=new calimpl();  int n = obj.sum(10,20);  System.out.println("Sum=" + n);  int m=obj.mul(5,6);  System.out.println("mul="+m);  int result1 = obj.div(20,4);  System.out.println("Division="+ result1);  int result3 = obj.sub(9,3);  System.out.println("Difference="+ result3);\*/  Calintfr calc=new calimpl();  int result = calc.sum(10,20);  System.out.println("Sum=" + result);  result = calc.mul(5,6);  System.out.println("Product="+ result);  //int result1 = obj.div(20,4);  System.out.println("Division="+ result1);  //Not accessible to call  //result=calc.div(5,6);  result = calc.div(20,4);  System.out.println("Division="+ result);  result= calc.sub(9,3);  System.out.println("Difference="+ result);  }  } |

**Task#03**

**Catch the Arithmetic Exception for divide function in above calculator.**

Code which can throw exception called as **try** block and exception handling code in another block called **catch** block. Whenever an exception is caught corresponding catch block is executed.

public class test {

public test(){

}

public static void main(String args[])

{

calimpl obj=new calimpl();

try{

int result1 = obj.div(10,2);

System.out.println("Division="+ result1);

int result2 = obj.div(10,0);

System.out.println("Division="+ result2);

}

catch (ArithmeticException e) {

System.out.println("Division by zero.");

}

}

}

A simple way to capture any exception is to use an object of Exception class as other classes inherit Exception class.

**class** Exceptions {

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

String languages[] = { "C", "C++", "Java", "Perl", "Python" };

**try** {

**for** (**int** c = 1; c <= 5; c++) {

System.out.println(languages[c]);

}

}

**catch** (Exception e) {

System.out.println(e);

}

}

}

Output of program:

C++

Java

Perl

Python

java.lang.ArrayIndexOutOfBoundsException: 5

Here catch block capture an exception which occurs because we are trying to access an array element which does not exists (languages[5] in this case). Once an exception is thrown control comes out of try block and remaining instructions of try block will not be executed.

**Task#04**

**Create a class Calculator with two variables, a constructor with no parameter and two functions for add and multiply. Use netbeans to create window forms with two text fields and two buttons for plus and multiply. Write code behind button to take values from text fields and create object of Calculator class to call functions by passing parameters.**

* **Use of netbeans to create window forms, get values from text fields and action on button.**

|  |  |
| --- | --- |
| **Calculator.java** | **CalculatorForm.java** |
| public class Calculator  {  int x, y;  public Calculator() {  }  int sum(int a, int b)  {  return a+b;  }  int mul(int a, int b)  {  return a\*b;  }  } | public class CalculatorForm extends JFrame  {  public CalculatorForm() {  }  public void btnPlusActionPerformed(ActionEvent e)  {  int v1=Integer.parseInt(txtval1.getText());  int v2=Integer.parseInt(txtval2.getText());  Calculator obj=new Calculator();  int s = obj.sum(v1,v2);  lblOutput.setText(“Sum = ” +s);  }  } |

**Output:**

