MODULE-3

Q-7 Write a program “DivideByZero” that takes two numbers a and b as input, computes a/b, and invokes Arithmetic Exception to generate a message when the denominator is zero.

CODE :

```
import java.util.Scanner;
public class DivideByZero {
   public static void main(String[] args) {
      Scanner input = new Scanner(System.in);
      System.out.print("Enter a: ");
      double a=input.nextDouble();
      System.out.print("Enter b: ");
      double b=input.nextDouble();
      try{
        double result =a/b;
        System.out.println("a/b = " + a + "/" + b +" = " + result);
      }catch(ArithmeticException e){
        System.out.println("Division by zero");
      }
   }
}
```

OUTPUT :

```
Enter a: 5
Enter b: 0
a/b = 5.0/0.0 = Infinity
```

Q-8 Write a program to show the use of nested try statements that emphasizes the sequence of checking for catch handler statements.

CODE:

```
class MultipleTryCatch
{
```

```
public static void main(String arg[])
{
   int arr[]=new int[5];
   try
   {

      try
      {
         System.out.println("Divide 1");
         int b=23/0;
      }
      catch(ArithmeticException e)
      {
         System.out.println(e);
      }
      try
      {
         arr[7]=10;
         int c=22/0;
         System.out.println("Divide 2 : "+c);
      }
      catch(ArithmeticException e)
      {
         System.out.println("Err:Divide by 0");
      }
      catch(ArrayIndexOutOfBoundsException e)

      {
         System.out.println("Err:Array out of bound");
      }
   }
```

```
        catch(Exception e)
        {
            System.out.println("Handled");
        }
    }
}
```

OUTPUT:

```
Divide 1
java.lang.ArithmeticException: / by zero
Err:Array out of bound
```

Q-9 Write a program to create your own exception types to handle situation specific to your application (Hint: Define a subclass of Exception which itself is a subclass of Throwable).

CODE :

```
import java.util.Scanner;
class InvalidBoxException extends Exception{
    InvalidBoxException()
    {}

    public InvalidBoxException(String str)
    {
        super(str);
    }
}
public class java9 extends InvalidBoxException{
    public static void validate(int l, int b, int h)  throws InvalidBoxException{
        if (l < 0 || b < 0 || h < 0)
        {
            throw new InvalidBoxException("Invalid Box");
        }
        else{
```

```
            System.out.println("Valid Box");
        }
    }
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter length: ");
        int length = sc.nextInt();
        System.out.println("Enter breadth: ");
        int breadth = sc.nextInt();
        System.out.println("Enter height: ");
        int height = sc.nextInt();
        try
        {
            validate(length, breadth, height);
        }
        catch (InvalidBoxException e){
            System.out.println("Exception caught");
            System.out.println("Exception occured: " + e);
        }
        sc.close();
    }
}
```

OUTPUT :

```
Enter length:
1
Enter breadth:
2
Enter height:
0
Valid Box
```

Q-10

Write a small application in Java to develop Banking Application in which user deposits the amount Rs 1000.00 and then start withdrawing of Rs 400.00, Rs 300.00 and it throws exception “Not Sufficient Fund” when user withdraws Rs. 500 thereafter.

CODE :

```
import java.util.*;
class Bank
{
   float fund;
   void deposit(float amount)
   {
      fund=amount;
   }
   void withdraw(float money) throws Exception
   {
      float newFund=fund-money;
      if(newFund<500)
      {
         throw new Exception("Not Sufficient Fund");
      }
      else
      {
         fund=newFund;
         System.out.println("Balance After Withdraw : "+fund);
      }
   }
      public static void main(String arg[])
      {
         Bank b=new Bank();
         b.deposit(1000.00f);
         try
```

```
        {
            float money;
            Scanner sc=new Scanner(System.in);
            System.out.println("Enter Your Amount for withdraw : ");
            money=sc.nextInt();
            System.out.println("Withdrawing amount : "+money);
            b.withdraw(money);
            /* here test with static data so don't worry
            money=300;
            System.out.println("Withdrawing amount : "+money);
            b.withdraw(money); */
        }
        catch(Exception e)
        {
            System.out.println(e.getMessage());
        }

    }
}
```

OUTPUT :

```
Enter Your Amount for withdraw :
400
Withdrawing amount : 400.0
Balance After Withdraw : 600.0

Enter Your Amount for withdraw :
1500
Withdrawing amount : 1500.0
Not Sufficient Fund
```