Course Name: A Level (2nd Sem) **Subject: JAVA**
Topic :try-catch Examples(contd) **Date: 13-04-20**

multiple catch blocks:

Example 2 of multiple catch blocks:

```
class Multicatch2{
  public static void main(String args[]){
    try{
        int a[]=new int[5];
        a[4]=20/0;
       System.out.println(a[7]);
        System.out.println("First  statement in try block");
    }
    catch(ArithmeticException e){
      System.out.println(" ArithmeticException occurs");
    }
    catch(ArrayIndexOutOfBoundsException e){
      System.out.println("ArrayIndexOutOfBoundsException occurs");
    }
    catch(Exception e){
      System.out.println(" Some Other exception occurs");
    }
  System.out.println("Out of try-catch block");
  }
}
```

Explanation:
In this example 2 exceptions occurred but at a time only on exception is handled and hence the corresponding catch block is resolved.

Example 3 of multiple catch blocks:

```
class Multicatch3{
  public static void main(String args[]){
    try{
        String s1=”Nielit”
       System.out.println(s1.charAt(9));

    }
    catch(ArithmeticException e){
      System.out.println(" ArithmeticException occurs");
```

```
    }
    catch(ArrayIndexOutOfBoundsException e){
      System.out.println("ArrayIndexOutOfBoundsException occurs");
    }
    catch(Exception e){
      System.out.println(" Some Other exception occurs");
    }
  System.out.println("Out of try-catch block");
 }
}
```

Explanation:

In this example the actual exception has not been handled in any of the try block, so in this case the generic exception will be caught.

Example 4 of multiple catch blocks:

```
class Multicatch4{
  public static void main(String args[]){
    try{
       String s1=”Nielit”
      System.out.println(s1.charAt(9));

    }
    catch(Exception e){
      System.out.println(" ArithmeticException occurs");
    }
    catch(ArrayIndexOutOfBoundsException e){
      System.out.println("ArrayIndexOutOfBoundsException occurs");
    }
    catch(ArithmeticException e){
      System.out.println(" Some Other exception occurs");
    }
  System.out.println("Out of try-catch block");
 }
}
```

Explanation:

In this example there will be compile time error because the sequence of exceptions have been changed.The exceptions must be ordered from most specific to most generic(i.e the exception);

Nested try block:

When a try catch block is present in another try block then it is called the nested try catch block. Each time a try block does not have a catch handler for a particular exception, then the catch blocks of parent try block are inspected for that exception, if match is found that that catch block executes.

```
class Nested{
 public static void main(String args[]){
  try{
    try{
  int a =20/0;
    }
catch(ArithmeticException e)
{
System.out.println(e);
}

    try{
    int a[]=new int[5];
    a[6]=7;
    }
catch(ArrayIndexOutOfBoundsException e)
{
System.out.println(e);
}

    System.out.println("normal statement);
  }
catch(Exception e)
{
System.out.println("exception handeled");
}

  System.out.println("out of try catch block....");
 }
}
```