Shreyas Sawant
D7A-55

AIM: To demonstrate the concept of exception handling in Java.

THEORY:

- Exception Handling

An exception is an abnormal condition that arises in a code sequence at run time. In other words, exception is a run-time error.
Java's exception handling avoids manually handling exceptions /errors and thus brings run-time error management into the object oriented world.
When an exception condition arises, an object representing that exception is created and thrown in the method that caused the error.

Java's exception handling is managed by five keywords - try, catch, throw, throws and finally.

(i) try statement - Program statements that you want to monitor for exceptions and contained in a try block. If an exception occurs, it is thrown.

Syntax :-
```
try {
        // block to monitor errors
}
```

(ii) catch statement - The exception thrown in a try statement is caught in catch statement. It can handle the exception in a rational manner.

Syntax :-
```
catch ( ExceptionType obj ) {
        //exception handler
}
```

There can be multiple catch statements for multiple ExceptionTypes

(iii) throw statement - System generated exceptions are automatically thrown by Java run-time system. To manually throw an exception, use throw keyword.

Syntax :- try {
throw new NullPointerException ("Demo");
} //similar for other exceptions

(iv) throws statement - If a method is capable of causing an exception that it does not handle, it must specify this behaviour. A throws clause lists the types of exceptions that the method might throw.

Syntax : type method-name (parameters) throws list {
// body of method
}

Here list is a comma-seperated list of exceptions that a method can throw

(v) finally statement : When exceptions are thrown, execution is a method takes a rather abrupt, non-linear path that alters the normal flow through the method. The finally block will execute whether or not an exception is thrown.

If a method opens a file, you will not want the code that closes the file to be bypassed by exception-handling mechanism.

Syntax : finally {
// code
}

- Multiple-catch statements :

In some cases, more than one exception could be raised by a single piece of code. To handle this, you can specify two or more catch clauses, each with a different type of exception. When an exception is thrown, each catch statement is inspected in order and the first one whose type matches that of the exception is executed.

Example -

```
class Multicatch {
    public static void main (String args[]) {
     try {
        int a = args.length ;
        int b = 42/a ;
        int c[] = {1} ;
        c[42] = 100 ;
      } catch (ArithmeticException e) {
          System.out.println (e) ;
      } catch (ArrayIndexOutOfBoundsException e) {
          System.out.println (e) ;
      }
    }
  }
}
```

The program will cause a division-by-zero exception if it is started with no command-line arguments (a = 0 and 42/a = 42/0)

It will causes ArrayIndexOutOfBoundsException if you provide a command-line argument, since int array c has length 1, yet the programs attempts to assign value to c[42].

It is important to remember that exception subclass must come before any of its superclass. This is because a catch statement that uses a superclass will catch exceptions of that type. Thus, subclass would never reach.

Although, Java's in-built exceptions handle most common errors, you can create you own exceptions to handle specific situations. Define a subclass of Exception.

Specifying a destroying is useful. This is done by overriding toString() message.

- Importance of Exception Handling

(i) Exception Handling provides a powerful mechanism for controlling complex programs that have many dynamic run-time characteristics.

(ii) When a method fails, it throws an exception. This is a cleaner way to handle failure modes.

CONCLUSION

- Errors encountered

1. error : ';' expected
   import java.util.*;

Solution Correct syntax : import java.util.*;

2. error : illegal start of expression
   if (a<0 ||| a>100)

Solution Correct syntax : if (a<0 || a>100)

LAB 9: EXCEPTION HANDLING IN JAVA

Name: Shreyas Sawant Div: D7A Roll No.: 55

Q. 1. Write a program to demonstrate the concept of multiple catch statements.

CODE:

```
public class MultipleCatches
{
    public static void main(String args[])
{
            try{
                 int a[]=new int[5];
                 a[Integer.parseInt(args[0])]=10;
                 a[5]=30/0;

                }
                 catch(ArrayIndexOutOfBoundsException e)
                   {
                    System.out.println("\nGoing out of range");
                   }
                catch(ArithmeticException e)
                   {
                    System.out.println("\nWhy are you dividing by zero?");
                   }
                catch(Exception e)
                   {
                    System.out.println("Parent Exception occurs");
                   }
               System.out.println("Out of try catch blocks");
    }
}
```

OUTPUT:

```
C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 09>javac MultipleCatches.java

C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 09>java MultipleCatches 10

Going out of range
Out of try catch blocks

C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 09>java MultipleCatches 4

Why are you dividing by zero?
Out of try catch blocks

C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 09>
```

Q.2. Write a program to demonstrate the working/importance of finally block.

CODE:

```
public class FinallyImp
{
      public static void main(String args[])
{
      try {
                System.out.println("Inside the try block");
                int data=25/0;
                System.out.println(data);
          }
        catch(NullPointerException e)
        {
                 System.out.println(e);
        }

        finally
        {
                 System.out.println("\nFinally block is always executed");
                 System.out.println("Could not find suitable catch\n");
                 System.out.println("Error thrown by compiler if it is there: \n");
        }

      System.out.println("Code is out of try catch and finally");
    }
}
```

OUTPUT:

```
C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 09>javac FinallyImp.java

C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 09>java FinallyImp
Inside the try block

Finally block is always executed
Could not find suitable catch

Error thrown by compiler if it is there:

Exception in thread "main" java.lang.ArithmeticException: / by zero
        at FinallyImp.main(FinallyImp.java:7)

C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 09>
```

Q.3. Write a program to create a user defined exception to calculate the result. The result should consist of name, seat no, date, centre no and marks of 3 semesters. Create a user defined exception class MarksOutOfBoundsException. If entered marks of any subject is greater than 100 or less than 0 then program should create a user defined exception of the type MarksOutOfBoundsException and must have a provision to handle it.

CODE:

```
import java.util.*;
class MarksOutOfBoundsException extends Exception
{
        int a;
        MarksOutOfBoundsException(int m)
        {
                a=m;
        }
        public String toString(){
                return "java.MarksOutOfBoundsException: "+a+" must be in range of 0 to 100 ";
        }
}


class Result
{
        void check(int a)throws MarksOutOfBoundsException
        {
                if(a<0||a>100)
                 throw new MarksOutOfBoundsException(a);
        }

        void input()
        {
                Scanner s=new Scanner(System.in);
                int seat_no,date,centre_no,marks1=0,marks2=0,marks3=0;
                String name;
```

```
                String name;
                System.out.print("\nStudent's name: ");
                name=s.nextLine();
                System.out.print("\nSeat number: ");
                seat_no=s.nextInt();
                System.out.print("\nDate: ");
                date=s.nextInt();
                System.out.print("\nCentre number: ");
                centre_no=s.nextInt();
                System.out.print("\nMarks of 3 semesters: ");
                int k=2;
                do
                {       k=2;
                        marks1=s.nextInt();
                        marks2=s.nextInt();
                        marks3=s.nextInt();
                        try
                        {
                                check(marks1);
                                check(marks2);
                                check(marks3);
                        }
                        catch(MarksOutOfBoundsException e)
                        {
                                System.out.println("You made an error: "+e);
                                System.out.print("\nEnter marks again: ");
                                k=1;
                        }
```

```
                        check(marks3);
                }
                catch(MarksOutOfBoundsException e)
                {
                        System.out.println("You made an error: "+e);
                        System.out.print("\nEnter marks again: ");
                        k=1;
                }

        }while(k==1);

        System.out.println("\nResult of "+name);
        System.out.println("Student's Name: "+name);
        System.out.println("Seat No. "+seat_no);
        System.out.println("Exam Date "+date);
        System.out.println("Centre No. "+centre_no);
        System.out.println("Marks: "+marks1+" "+marks2+" "+marks3);

    }

    public static void main(String args[])
    {
        Result a=new Result();
        a.input();
    }
}
```

OUTPUT:

```
C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 09>javac Result.java

C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 09>java Result

Student's name: Liton Dhu

Seat number: 4568

Date: 050221

Centre number: 456987

Marks of 3 semesters: 45 65 21

Result of Liton Dhu
Student's Name: Liton Dhu
Seat No. 4568
Exam Date 50221
Centre No. 456987
Marks: 45 65 21

C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 09>java Result

Student's name: Poli Kio

Seat number: 4865

Date: 090620

Centre number: 4578213
```

```
Centre number: 4578213

Marks of 3 semesters: 102 112 12
You made an error: java.MarksOutOfBoundsException: 102 must be in range of 0 to 100

Enter marks again: 12 45 96

Result of Poll Kio
Student's Name: Poll Kio
Seat No. 4865
Exam Date 90620
Centre No. 4578213
Marks: 12 45 96

C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 09>
```