LAB EXERCISE ON EXCEPTION HANDLING

NAME : SHERAL SIMON WASKAR

REG. NO : 20BCE1182

COURSE CODE : CSE 1007 LAB

COURSE : JAVA PROGRAMMING LAB

SLOT : L13-L14

FACULTY : J V THOMAS ABRAHAM

DATE : 17/03/2022

1)Write an application that throws and catches an ArithmeticException when you attempt to take the square root of a negative value. Prompt the user for an input value and try the Math.sqrt() method on it. The application either displays the square root or catches the thrown Exception and displays an appropriate message. Save the file as SqrtException.java.

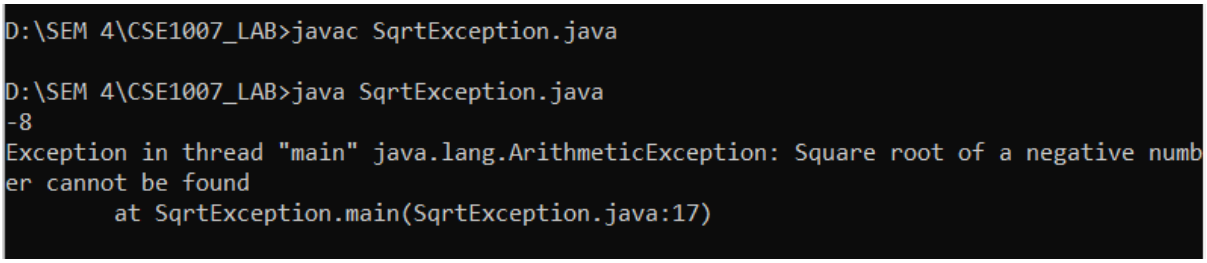
Code :

```
import java.util.Scanner;
import java.io.*;
import java.lang.Math;

class SqrtException
{
```

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


    Scanner sc = new Scanner(System.in);
    double n = sc.nextDouble();
    if (n<0)
    {
    throw new ArithmeticException("Square root of a
negative number cannot be found ");
    }
    else
{
   double res = Math.sqrt(n);
  System.out.println("Square root of number is  : " +
res);


}


}
}
```

Output :

```
D:\SEM 4\CSE1007_LAB>javac SqrtException.java

D:\SEM 4\CSE1007_LAB>java SqrtException.java
16
Square root of number is  : 4.0
```



2)The Double.parseDouble() method requires a String argument, but it fails if the String cannot be converted to a floating-point number. Write an application in which you try accepting a double input from a user and catch a NumberFormatException if one is thrown. The catch block forces the number to 0 and displays an appropriate error message. Following the catch block, display the number. Save the file as TryToParseDouble.java.

Code :

```
import java.util.Scanner;
import java.io.*;

class TryToParseDouble
{
```

```
public static void main(String[] args)

{
try
{

double r = Double.parseDouble("56.45");
System.out.println("The number is : " + r);

double s = Double.parseDouble("78.556");
System.out.println("The number is : " +s);

double t = Double.parseDouble("apple");
System.out.println(t);




}

catch(NumberFormatException nf)
```

{

System.out.println("The given input string cannot be converted to double ");

}

}

}

Output :

```
D:\SEM 4\CSE1007_LAB>javac TryToParseDouble.java

D:\SEM 4\CSE1007_LAB>java TryToParseDouble.java
The number is : 56.45
The number is : 78.556
The given input string cannot be converted to double
```

3) Define Employee class with Employee code, name, date of birth and date of appointment. The Employee code must have the format of year-designation-number. The year is a two digit integer such as 87. the designation is a single letter code M for manager, A for Administrative staff, H for HR dept staff, E for Executive staff, and T for Technical staff. The number is a three digit number. The following are some sample employee codes. 82-M-183 76-A-242 71-H-107 .Write a

Java program to read the employee code, name, date of birth, and date of appointment and validate the employee code. If the employee code is incorrect a suitable user defined exception must be thrown. Then verify if date of birth is before date of appointment. If it is not so, then throw another user defined Exception. If it is correct, then create the Employee object, display the count of employee and display the details of employees.

Code :

```
import java.io.*;
class InvalidEmpNumberException extends Exception
{
	String a;
	InvalidEmpNumberException(String detail)
	{
		a=detail;
	}
	public String toString()
	{

return("InvalidEmpNumberException:"+a);
```

```
        }
}
class InvalidDateOfJoinException extends Exception
{
        String h;
        InvalidDateOfJoinException(String vals)
        {
                h=vals;
        }
        public String toString()
        {
                return("InvalidDateOfJoinException:"+h);
        }
}
class Main
{
static void compute(String x)throws
InvalidEmpNumberException
{
        String y=x;
```

```
        int count=0,count1=0;
        char z[]=new char[10];
        for(int i=0;i<y.length();i++)
        {
                z[i]=y.charAt(i);
        }
        for(int i=0;i<6;i++){
        if(i!=2){
                if(z[i]=='0' ||z[i]=='1' ||z[i]=='2' ||z[i]=='3'
||z[i]=='4' ||z[i]=='5' ||z[i]=='6' ||z[i]=='7' ||z[i]=='8'
||z[i]=='9')
                count=count+1;
        }}
        if(z[2]=='m' || z[2]=='a' || z[2]=='h' || z[2]=='e'
|| z[2]=='t')
        count=count+1;

        if(count==y.length())
        {
        System.out.println("the Employee code is
valid");
```

```
        count1 = count1 + 1;
        System.out.println("The count of valid
employees is : " + count1);
        }
        else
        throw new InvalidEmpNumberException(y);
}
static void calc(String n)throws
InvalidDateOfJoinException
{
        int x;
        x=Integer.parseInt(n);
        System.out.println("date of joining: 5600");
        if(x<5600)
        System.out.println("valid date");
        else
        throw new InvalidDateOfJoinException(n);
}
public static void main(String args[]) throws
InvalidEmpNumberException
{
```

```
        try
        {
                System.out.println("Employee code :
94e320");
                compute("94e320");
                System.out.println("Birth year:1983");
                calc("1983");
                System.out.println("Birth year:2012");
                calc("2012");
                System.out.println("Employee code
:97tm564");
                compute("97tm564");

        }
        catch(InvalidDateOfJoinException e)
        {
                System.out.println(e);
        }
        catch(InvalidEmpNumberException e1)
        {
                System.out.println(e1);
```

```
        }
}
}
```

Output :

```
D:\SEM 4\CSE1007_LAB>javac Employee.java

D:\SEM 4\CSE1007_LAB>java Employee
```

```
Employee code : 94e320
the Employee code is valid
The count of valid employees is : 1
Birth year:1983
date of joining: 5600
valid date
Birth year:2012
date of joining: 5600
valid date
Employee code :97tm564
InvalidEmpNumberException:97tm564
```