**TASK 11**

**JAVA EXCEPTIONS AND ERROR HANDLING**

1. **What are the four access modifiers available in java and what is their significance in terms of class, methods and variable accessibility?**

There are four types of access modifiers available in Java:

1. Default – No keyword required
2. Private
3. Protected
4. Public

**Default:**

When no access modifier is specified for a class, method, or data member – It is said to be having the default access modifier by default. The data members, classes, or methods that are not declared using any access modifiers i.e. having default access modifiers are accessible only within the same package.

**Private:**

The private access modifier is specified using the keyword private. The methods or data members declared as private are accessible only within the class in which they are declared.

* Any other class of the same package will not be able to access these members.
* Top-level classes or interfaces cannot be declared as private because

Hence these modifiers in terms of application to classes, apply only to nested classes and not on top-level classes

**Protected:**

The protected access modifier is specified using the keyword protected. The methods or data members declared as protected are accessible within the same package or subclasses in different packages. And protected means “only visible within the enclosing class and any subclasses”

**Public:**

The public access modifier is specified using the keyword public. The public access modifier has the widest scope among all other access modifiers. Classes, methods, or data members that are declared as public are accessible from everywhere in the program. There is no restriction on the scope of public data members.

1. **What is the difference between exception and error?**

**Errors:**

Errors are usually caused by serious problems that are outside the control of the program, such as running out of memory or a system crash

**Exceptions:**

Exceptions, on the other hand, are used to handle errors that can be recovered from within the program

1. **What is the difference between checked exception and unchecked exception?**

**Checked exception:**

Theseare the exceptions that are checked at compile time. If some code within a method throws a checked exception, then the method must either handle the exception or it must specify the exception using the [throws keyword](https://www.geeksforgeeks.org/throw-throws-java/). In checked exceptions, there are two types: fully checked and partially checked exceptions. A fully checked exception is a checked exception where all its child classes are also checked, like IOException, and InterruptedException. Everything under throwable is checked Exception.

**Unchecked exception:**

These are the exceptions that are not checked at compile time. Exceptions under Error and RuntimeException classes are unchecked exceptions. In short unchecked exceptions are runtime exceptions that are not required to be caught or declared in a throws clause. These exceptions are usually caused by programming errors, such as attempting to access an index out of bounds in an array or attempting to divide by zero. Unchecked exceptions include all subclasses of the RuntimeException class, as well as the Error class and its subclasses.

1. **Write a Java program that reads user input for two integers and performs division. Handle the exception that is thrown when the second number is zero, and display an error message to the user.**

**package** task11;

**import** java.util.Scanner;

**public** **class** DivisibleByZero {

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

**int** a,b;

Scanner scan =**new** Scanner(System.***in***);

System.***out***.println("Enter the no 1");

a=scan.nextInt();

System.***out***.println("Enter the no 2");

b=scan.nextInt();

**try** {

System.***out***.println("Divided value of 2 numbers: "+(a/b));

//while dividing if it throws error it will send to catch statement

}**catch**(ArithmeticException ex) {

System.***out***.println("Divided by zero operation cannot possible so try again "+ex);

}

scan.close();

}

}

**OUTPUT:**

Enter the no 1

2

Enter the no 2

0

Divided by zero operation cannot possible so try again java.lang.ArithmeticException: / by zero

1. **Write the code of ArrayIndexOutOfBoundsException & StringIndexOutOfBoundsException?**

**package** task11;

**import** java.util.Scanner;

**public** **class** ArrayAndStringIndexOutBound {

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

**int** n,k,p;

Scanner scan =**new** Scanner(System.***in***);

System.***out***.println("Enter the no:");

n=scan.nextInt();

scan.nextLine();

System.***out***.println("Enter the String:");

String arr[]=**new** String[n];

**for**(**int** i=0;i<n;i++)

arr[i]=scan.nextLine();

**try** {

System.***out***.println("Enter the index in which you need to access the array:");

k=scan.nextInt();//if the input exceed the array limit or less then 0 it will pass to ArrayIndexOutOfBoundsException

System.***out***.println(arr[k]);

System.***out***.println("Enter the index of String which you need access of the selected String:");

p=scan.nextInt();//if the input exceed the string limit or less then 0 it will pass to StringIndexOutOfBoundsException

System.***out***.println(arr[k].charAt(p));

System.***out***.println("Selected string is: "+arr[k]+ " Selected char is : "+arr[k].charAt(p));

}**catch**(ArrayIndexOutOfBoundsException ex) {

System.***out***.println(ex);

}**catch**(StringIndexOutOfBoundsException ex1) {

System.***out***.println(ex1);

}**catch**(Exception ex2) {

System.***out***.println(ex2);

}

}

}

**OUTPUT:**

Enter the no:

9

Enter the String:

89

Enter the index in which you need to access the array:

2

Enter the index of String which you need access of the selected String:

3

java.lang.StringIndexOutOfBoundsException: Index 3 out of bounds for length 0

2.)

Enter the no:

9

Enter the String:

0

Enter the index in which you need to access the array:

0

0

Enter the index of String which you need access of the selected String:

1

java.lang.StringIndexOutOfBoundsException: Index 1 out of bounds for length 1

1. **You are building a login system for a website using Java. If the user enters an incorrect password, you want to display a message informing them of the error. How would you use exception handling to handle this situation?**

**package** task11;

**import** java.util.MissingFormatArgumentException;

**import** java.util.Scanner;

**import** javax.naming.NameNotFoundException;

**public** **class** Login {

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

// String Uname[]= {"Admin","User1","User2"};

// for (int i=0;i<3;i++)

// System.out.println(Uname[i].equals("Admin")+Uname[i]);

String name,password;

String Uname[]= {"Admin","User1","User2"};

String Pass[]= {"Admin123","pass","word"};

Scanner scan =**new** Scanner(System.***in***);

System.***out***.println("Enter the user name:");

name=scan.nextLine();

System.***out***.println("Enter the password: ");

password=scan.nextLine();

System.***out***.println("Entered Username:"+name+" Password :"+password);

**try** {

**int** poc;

**for**(**int** i=0;i<3;i++) {

**if**(Uname[i].equals(name))

poc=i;

**else**

**throw** **new** NameNotFoundException("User Name not found please try again !...");

**if**(password.equals(Pass[poc])) {

System.***out***.println("Welcome to the webpage...!");

**break**;

}

**else**

**throw** **new** MissingFormatArgumentException("Password is not matching please try again !...");

}

}**catch**(NameNotFoundException ex) {

System.***out***.println(ex.getMessage());

}

**catch**(MissingFormatArgumentException ex1) {

System.***out***.println(ex1.getMessage());

}

}

}

**OUTPUT:**

Enter the user name:

Admin

Enter the password:

Admin321

Entered Username:Admin Password :Admin321

Format specifier 'Password is not matching please try again'

1. **Create a custom exception in Java called "InvalidAgeException" that is thrown when the user enters an age less than 18. Implement exception handling in a Java program to catch the "InvalidAgeException" and display an error message.**

**package** task11;

**import** java.util.Scanner;

**class** InvalidAgeException **extends** Exception{

**public** InvalidAgeException(String msg) {

**super**(msg);

}

}

**public** **class** AgeException {

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

**int** age;

Scanner scan=**new** Scanner(System.***in***);

**try** {

System.***out***.println("Enter the Age to check if its ok to enter the venue:");

age=scan.nextInt();

**if**(age>17)

System.***out***.println("you can enter the venue...!");

**else**

**throw** **new** InvalidAgeException("Age should be greater then 18 and you have not permitted to enter the venue");

}**catch**(InvalidAgeException ex) {

System.***out***.println(ex.getMessage());

}

}

}

**OUTPUT:**

1.)Enter the Age to check if its ok to enter the venue:

16

Age should be greater then 18 and you have not permitted to enter the venue

2.) Enter the Age to check if its ok to enter the venue:

25

you can enter the venue...!

1. **Implement exception handling in a Java program that reads data from a file. If the file does not exist, throw a "FileNotFoundException" and display an error message to the user.**

package task11;

import java.io.BufferedReader;

import java.io.FileNotFoundException;

import java.io.FileReader;

import java.io.IOException;

import java.util.Scanner;

public class FileNotFoundexp {

public static void main(String[] args) {

String filePath;

Scanner sc=new Scanner(System.in);

System.out.println("Enter the File Path:");

filePath=sc.nextLine();

try {

FileReader fr=new FileReader(filePath);

BufferedReader br = new BufferedReader(fr);

String data =null;

while ((data = br.readLine()) != null)

{

System.out.println(data);

}

sc.close();

} catch (FileNotFoundException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

**OUTPUT:**

Enter the File Path:

C:\Users\Lenovo\OneDrive\Desktop\TASK

java.io.FileNotFoundException: C:\Users\Lenovo\OneDrive\Desktop\TASK (Access is denied)

at java.base/java.io.FileInputStream.open0(Native Method)

at java.base/java.io.FileInputStream.open(FileInputStream.java:213)

at java.base/java.io.FileInputStream.<init>(FileInputStream.java:152)

at java.base/java.io.FileInputStream.<init>(FileInputStream.java:106)

at java.base/java.io.FileReader.<init>(FileReader.java:60)

at task11.FileNotFoundexp.main(FileNotFoundexp.java:17)