**Assisted Practice: 3.5 Keywords and Custom Exceptions**

This section will guide you to:

* Write a program in Java to demonstrate try and catch
* Use Eclipse (the popular text editor for Java programs)
* Push code to Git

This lab has six sub-sections, namely:

* + 1. Creating a new project in Eclipse
    2. Writing a program in Java to demonstrate the **throw** keyword
    3. Writing a program in Java to demonstrate the **throws** keyword
    4. Writing a program in Java to demonstrate the **finally** keyword
    5. Writing a program in Java to demonstrate custom exceptions
    6. Pushing the code to your GitHub repositories

**Step 3.5.1:** Creating a new project in Eclipse

* Open Eclipse
* Go to File -> New -> Project -> Java Project -> Next.
* Type in any project name and click on “Finish.”
* Select your project and go to File -> New -> Class.
* Enter **ThrowDemo** in class name, check the checkbox “public static void main(String[] args)”, and click on “Finish.”

**Step 3.5.2:** Writing a program in Java to demonstrate the **throw** keyword

public class ThrowDemo

{

public static void main(String[] args)

{

int a=45,b=0,rs;

try

{

if(b==0)

throw(new ArithmeticException("Can't divide by zero."));

else

{

rs = a / b;

System.out.print("\n\tThe result is : " + rs);

}

}

catch(ArithmeticException Ex)

{

System.out.print("\n\tError : " + Ex.getMessage());

}

System.out.print("\n\tEnd of program.");

}

}

**Output:**



**Step 3.5.3:** Writing a program in Java to demonstrate the **throws** keyword

* Select your project and go to File -> New -> Class.
* Enter **ThrowsDemo** in class name, check the checkbox “public static void main(String[] args)”, and click on “Finish.”

public class ThrowsDemo

{

void Division() throws ArithmeticException

{

int a=45,b=0,rs;

rs = a / b;

System.out.print("\n\tThe result is : " + rs);

}

public static void main(String[] args)

{

ThrowsDemo T = new ThrowsDemo();

try

{

T.Division();

}

catch(ArithmeticException Ex)

{

System.out.print("\n\tError : " + Ex.getMessage());

}

System.out.print("\n\tEnd of program.");

}

}

**Output:**



**Step 3.5.4:** Writing a program in Java to demonstrate the **finally** keyword

* Select your project and go to File -> New -> Class.
* Enter **FinallyBlockDemo** in class name, check the checkbox “public static void main(String[] args)”, and click on “Finish.”

public class FinallyBlockDemo

{

public static void main(String[] args)

{

int a=45,b=0,rs=0;

try

{

rs = a / b;

}

catch(ArithmeticException Ex)

{

System.out.print("\n\tError : " + Ex.getMessage());

}

finally

{

System.out.print("\n\tThe result is : " + rs);

}

}

}

**Output:**



**Step 3.5.5:** Writing a program in Java to demonstrate custom exceptions

* Select your project and go to File -> New -> Class.
* Enter **Main** in any class name, check the checkbox “public static void main(String[] args)”, and click on “Finish.”

class MyException extends Exception

{

public MyException(String s)

{

super(s);

}

}

public class Main

{

public static void main(String args[])

{

try

{

throw new MyException("temp");

}

catch (MyException ex)

{

System.out.println("Caught");

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

}

}

}

**Output:**



**Step 3.5.6:** Pushing the code to your GitHub repositories

* Open your command prompt and navigate to the folder where you have created your files.

**cd <folder path>**

* Initialize your repository using the following command:

**git init**

* Add all the files to your git repository using the following command:

**git add .**

* Commit the changes using the following command:

**git commit . -m “Changes have been committed.”**

* Push the files to the folder you initially created using the following command:

**git push -u origin master**