**Section 1: Define / Answer**

Describe the difference between an **Error** in Java and an **Exception.**

**Errors should not be caught or handled (except in the rarest of cases). Exceptions are the bread and butter of exception handling.**

Describe the difference between **Checked Exceptions** and **Unchecked Exceptions**

Checked: are 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 throws keyword.

Unchecked are the exceptions that are not checked at compiled time.

Give a few examples of **Checked Exceptions.**

|  |
| --- |
| class Main {      public static void main(String[] args) {          FileReader file = new FileReader("C:\\test\\a.txt");          BufferedReader fileInput = new BufferedReader(file);            // Print first 3 lines of file "C:\test\a.txt"          for (int counter = 0; counter < 3; counter++)              System.out.println(fileInput.readLine());            fileInput.close();      }  } |

Output:

Exception in thread "main" java.lang.RuntimeException: Uncompilable source code -

unreported exception java.io.FileNotFoundException; must be caught or declared to be

thrown

at Main.main(Main.java:5)

Give a few examples of **Unchecked Exceptions.**

|  |
| --- |
| class Main {     public static void main(String args[]) {        int x = 0;        int y = 10;        int z = y/x;    }  } |

Output:

Exception in thread "main" java.lang.ArithmeticException: / by zero

at Main.main(Main.java:5)

Java Result: 1

Describe basic structure of

**try**{

}

**Catch**

{

}

**finally**

A try statement is used to catch exceptions that might be thrown as your program executes. You should use a try statement whenever you use a statement that might throw an exception That way, your program won’t crash if the exception occurs.

The finally block *always* executes when the try block exits. This ensures that thefinally block is executed even if an unexpected exception occurs. But finally is useful for more than just exception handling — it allows the programmer to avoid having cleanup code accidentally bypassed by a return, continue, or break. Putting cleanup code in afinally block is always a good practice, even when no exceptions are anticipated.

When to use **throws** vs **try/catch**?

Before you can catch an exception, some code somewhere must throw one. Any code can throw an exception: your code, code from a package written by someone else such as the packages that come with the Java platform, or the Java runtime environment. Regardless of what throws the exception, it's always thrown with the throw statement.

You associate exception handlers with a try block by providing one or more catch blocks directly after the try block. No code can be between the end of the try block and the beginning of the first catch block.

Task 1-

**USE OBJECT ORIENTATED PROGRAM DESIGN TO SOLVE PROBLEM**

Change assignment #5

Complete the program with nested menus.

First provide a menu giving the user the opportunity to select the vehicle type first.

Then provide a second menu to allow the user to input values to calculate range.

The program should not crash no matter the user input. Deal with all unexpected input.

