Activity 6.1: Exception Handling

- Problem Statement:
  - In the Hangman game, Peter wants to implement exception handling, such that if a user enters a menu input other than 1, 2, or 3 in the game menu option, an appropriate user-defined exception should be generated. In addition, a user-defined exception should be generated if the player inputs multiple characters, instead of a single character to identify the possible alphabet in the word.

Activity 6.1: Exception Handling (Contd.)

- Solution: To perform the activity, refer the steps given in the embedded document.

Microsoft Word Document

Introduction to Java

Using the assert Keyword

- Assertions are statements in Java that enable you to test any assumptions that you make regarding a program during its execution.
- The assumptions in a program can be simple facts, such as the range of a number should be between 10 and 20 or a number cannot be greater than 100.
- You can implement assertions by using the `assert` keyword provided in Java.

Understanding Assertions

- Assertions are used during the testing of a program.
- You can test this by asserting that the value being passed to that particular method is greater than zero by using an `assert` keyword.
- Assertions in Java provide the following benefits:
  - By using assertions, data can be validated easily.
  - By using assertions, you can confirm whether the program is working as expected.
  - By using assertions, the task of debugging is simplified as assertions can easily indicate the source and the reason for an error.

Introduction to Java

Implementing Assertions

- The `assert` keyword is used to implement assertions.
- You can declare an assertion by using the following syntax:

```
assert expressionA;
```

Implementing Assertions (Contd.)

- In order to place an assertion to check that the value of age is greater than `0`, but less than `130`, you need to use the following code:

```
public class ValidateAge
{
public static void main(String[] args)
{
int age;
Scanner obj1 = new Scanner(System.in);
System.out.println("Enter the age: ");
age = obj1.nextInt();
assert(age>0)&&(age<130);
System.out.println("The entered age is: " +age);
}
}
```

Implementing Assertions (Contd.)

- You can also declare assertion by using the following syntax:

```
assert expression1 : expression2;
```

Introduction to Java

Just a minute

- Which one of the following errors is generated if an assertion does not hold true?
  - `AssertionError`
  - `IllegalAccess`
  - `TypeMismatch`
  - `IllegalAssertion`

Introduction to Java

Just a minute (Contd.)

- Solution:
  - `AssertionError`

Activity 6.2: Implementing Assertions

- Problem Statement:
  - Create a calculator application in Java that will accept two numbers. Further, the calculator application should be able to perform the following operations one at a time on the two numbers:
    - Addition
    - Subtraction
    - Multiplication
    - Division

    You need to implement assertions and assert that both the numbers should be greater than 0. Further, the operator used to perform the calculations should only be +,-, *, or, /.

Activity 6.2: Implementing Assertions (Contd.)

- Solution: To perform the activity, refer the steps given in the embedded document.

Microsoft Word Document

Summary

- In this session, you learned that:
  - When a run-time error occurs, an exception is thrown by the JVM which can be handled by an appropriate exception handler.
  - To deal with these exceptions, Java provides various built-in exception classes.
  - The `Throwable` class is the base class of exceptions in Java.
  - The `Exception` class represents the conditions that a program should handle.
  - The `Error` class defines the exceptions related to the Java run-time environment.
  - Java exceptions are categorized into the following types:
    - Checked exceptions
    - Unchecked exceptions

Summary (Contd.)

- You can implement exception handling in a program by using the following keywords and blocks:
  - `try`
  - `catch`
  - `throw`
  - `throws`
  - `finally`
  - `try`-with-resources
- A `try` block encloses the statements that might raise an exception and defines one or more exception handlers associated with it.
- In Java, the `catch` block is used as an exception handler.
- A `try` block must have at least one `catch` block that follows the `try` block, immediately.

Summary (Contd.)

- You can throw an exception explicitly, by using the `throw` keyword.
- The `throws` keyword is used by a method to specify the types of exceptions that the method can throw.
- The statements specified in the `finally` block are executed after the control has left the try-catch block.
- The `try`-with-resources block ensures that one or more system resources are released when they are no longer required.
- In addition to the built-in exceptions, you can create customized exceptions, as per the application requirements.
- Assertions are statements in Java that enable you to test any assumptions that you make regarding a program during its execution.
- You can implement assertions by using the `assert` keyword provided in Java.