**Task 11**

1. **What are the four access modifiers in java. What is the significance in class, method and variable accessibility?**

There are four types of access modifiers in java namely,

* Public
* Protected
* Default
* Private

1. **Public**

* Class: A public class can be accessible from any other class.
* Method: A public method can be called from any other class
* Variable: A public variable can be accessed from any other class.

Significance: A public variable can be accessed from any other code, making them available globally.

1. **Protected:**

* Class: The protected modifier is not applicable to classes. (classes can have protected constructor and nested classes)
* Method: A protected method is accessible within the same package and by subclasses in other packages.
* Variable: A protected variable is accessible within the same package and subclasses in other packages.

1. **Default:**

* Class: A class with no access modifiers is accessible only with its own package.
* Method: A default method is accessible only with its own package.
* Variable: A default variable is accessible only with its own package.

1. **Private:**

* Class: The private modifier is not applicable to the top-level classes.
* Method: A private method is accessible only within the class defined in.
* Variable: A private variable is accessible only within the class it is defined in.

**Example syntax:**

// Public class example

public class PublicClass {

// Public variable

public int publicVariable;

// Protected variable

protected int protectedVariable;

// Default (package-private) variable

int defaultVariable;

// Private variable

private int privateVariable;

// Public method

public void publicMethod() {

// Accessible anywhere

}

// Protected method

protected void protectedMethod() {

// Accessible within package and subclasses

}

// Default method

void defaultMethod() {

// Accessible within package

}

// Private method

private void privateMethod() {

// Accessible within this class only

}

}

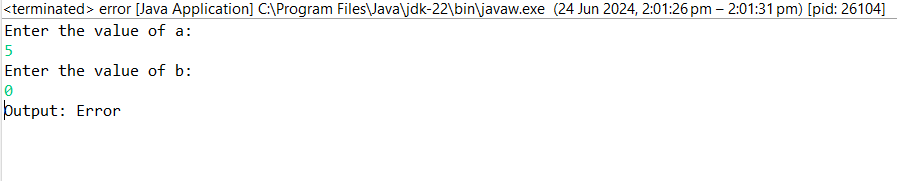
1. **Difference between Exception and error.**

|  |  |  |
| --- | --- | --- |
| Feature | Exception | Error |
| Definition | An event that disrupts the normal flow of the program but usually handled. | A serious problem indicating an issue with run time environment (JVM) cannot handle. |
| Handling | Must be explicitly handled (try catch or throws) | Cannot be caught or handled by the program |
| Checked/Un-checked | Both (checked and Unchecked) | Always Unchecked |
| Cause | Typically caused by developer code issues | Beyond developer control |
| Examples | NullPointerException,  ArrayIndexOfboundsException | OutofMemory error,StackoverflowError |

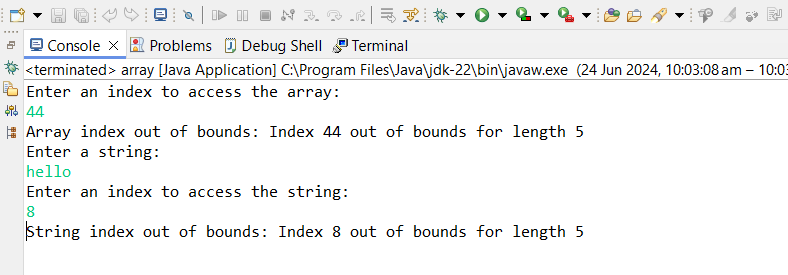
1. **Difference between Checked exception and unchecked exception.**

|  |  |  |
| --- | --- | --- |
| Feature | Checked Exception | Unchecked Exception |
| Definition | Exception that are checked during compling time | Exceptions that are not checked during the compile tie |
| Hierarchy | Sub class of Exception excluding “Run Time exception” | Sub class of Run time Exception |
| Compile Time | Checked by the compiler | Not checked by the compiler |
| Common Use Cases | Used for recoverable conditions that a program should handle, such as file I/O operations or database access | Used for programming errors that are typically caused by bugs in the code, such as dereferencing a null pointer or accessing an array out of bounds. |
| Compile Time Checking | Yes, checked by the compiler to ensure proper handling. | No, not checked by the compiler. |

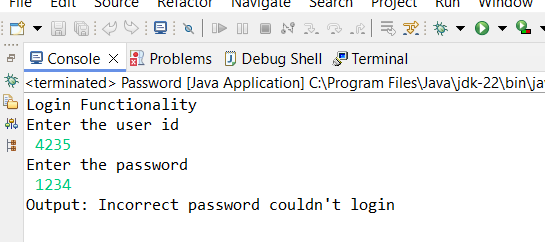
1. **Read two integers from user and perform division. Give second integer as zero. Display error message to user.**

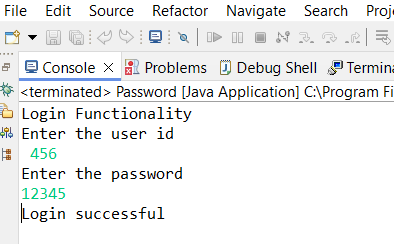
****

1. **Code of array&string index out of bound exception.**

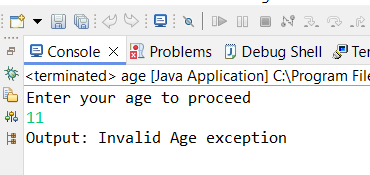


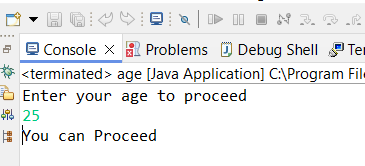
1. **Login Functionality**





1. **In-valid age exception.**





1. **File not found exception.**

