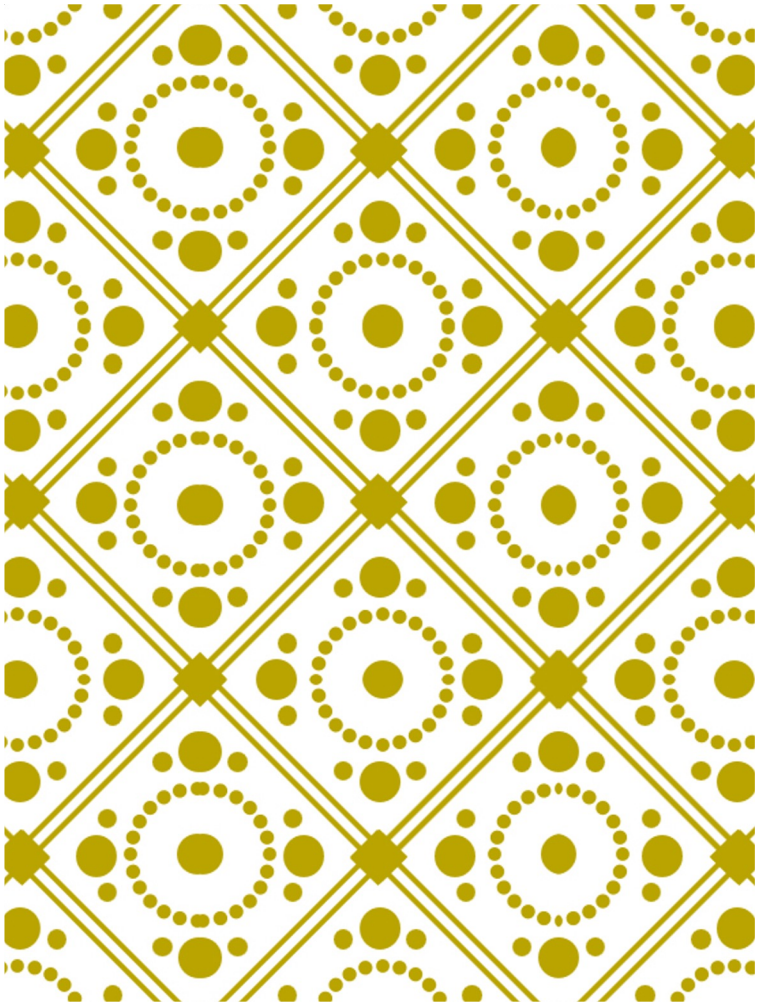

WORKSHOP 6

Understanding Java Exceptions

CSX3004 Programming Languages
Kwankamol Nongpong

ACTIVITY: UNDERSTANDING JAVA EXCEPTIONS

Time Limit: 60 minutes

TASK 1: CONCEPT CHECK

1. What happens if an exception is thrown but not caught in Java? Provide a sample Java program for such a case.
2. What is the difference between `RuntimeException` and `IOException`?

TASK 2: EXCEPTION HANDLING

```
1. public class Task2 {
2.     public static void main(String[] args) {
3.         int[] numbers = {2, 10, 8};
4.         try {
5.             System.out.println("Number: " + numbers[5]);
6.             int result = divide(numbers[1], 0);
7.             System.out.println("Result: " + result);
8.         } catch (ArrayIndexOutOfBoundsException e) {
9.             System.out.println("Array error: " + e.getMessage());
10.        } catch (ArithmeticException e) {
11.            System.out.println("Math error: " + e.getMessage());
12.        } finally {
13.            System.out.println("Execution finished.");
14.        }
15.    }
16.    public static int divide(int a, int b) {
17.        return a / b;
18.    }
19. }
```

1. Without running the code, identify where the exceptions occur and state the type of each exception.
2. Modify the program so the exceptions are handled separately.
3. Submit Task2.java

TASK 3: CUSTOM EXCEPTION (1)

Create a class called Task3.

1. Refer to your answer from Task 2.
2. Create a custom exception class for dividing by zero.
3. Update the main program so that the custom exception class is used.

Expected Output

```
Array error: Out of bounds! Accessing index 5 for length 3.
Custom math error: Cannot divide by zero!
```

TASK 4: CUSTOM EXCEPTION (2)

Create a class called Task4

1. Define a `NegativeNumberException` class.
2. Write a method `mySqrt(int n)` that throws `NegativeNumberException` if `n < 0`.
3. Implement a main program that tries to calculate a square root of a negative number and handles the exception gracefully.

TASK 5: EXCEPTION PROPAGATION

Create a class called Task5.

Implement three methods: `main()`, `methodA()` and `methodB()`

1. Make the call sequence: `main()` → `methodA()` → `methodB()`.
2. `methodB()` throws an exception (any exception is fine).
3. `methodA()` doesn't handle but propagate the exception.
4. `main()` catch and display the exception.