BSc (Hons) in Information Technology
Year 2

Tutorial 4

IT2030 – Object Oriented Programming **Semester 1, 2023**

Exercise 1

You are going to develop a system to manage different types of employees in a company. Refer to the below abstract class called Employee.

```
public abstract class Employee {
      protected String name;
      protected int employeeId;

      public Employee(String name, int employeeId){
          this.name = name;
          this.employeeId = employeeId;
      }

      public abstract double calculateSalary();
      public abstract void displayDetails();
   }
```

Create two subclasses of Employee using the appropriate attributes:

a) **Manager**: This subclass should implement the "`calculateSalary()`" method to calculate the manager's salary based on a fixed salary amount. Additionally, it should implement the "`displayDetails()`" method to display the manager's name, employee ID, and position as Manager.

b) **Developer**: This subclass should implement the "`calculateSalary()`" method to calculate the developer's salary based on an hourly wage and the number of hours worked. Additionally, it should implement the "`displayDetails()`" method to display the developer's name, employee ID, hourly wage, and hours worked.

c) Create a separate class called `EmployeeApp` with the main method. create objects of both Manager and Developer, calculate their salaries, and display their details.

Tutorial 4

IT2030 – Object Oriented Programming **Semester 1, 2023**

Exercise 2

You are going to develop an application which will be using in a supermarket. Refer to the below class called “ShoppingCart”.

```
public class ShoppingCart {
    private double total;
    private int itemCount;

    public ShoppingCart() {
        total = 0.0;
        itemCount = 0;
    }

    public void addItem(double price) {
      //Implement adding an item to the shopping cart and handle
exceptions
    }

    public double calculateTotalPrice() {
       //Implement calculating the total price and handle exceptions
        return 0.0;
    }
}
```

a) Add items to the shopping cart by specifying their prices. Handle `IllegalArgumentException` when attempting to add items with negative prices or prices greater than $1,000.

b) Calculate the total price of the items in the cart.

c) Handle `ArithmeticException` when attempting to calculate the total price if the cart is empty (if `itemCount` is 0).

d) Display error messages for each type of exception

Exercise 3

a) Create a custom exception class called **AgeValidationException** that extends the Exception class.

b) Inside the AgeValidationException class, implement a constructor that takes a message to describe the age validation error.

c) Create another class called **AgeValidationDemo.** Inside this class implement a method called “`validateAge`()” that checks whether an age is within the valid range (0 to 120). If the age is outside this range, it should throws the custom AgeValidationException with an error message.

d) Implement a main method inside AgeValidationDemo class and call the validateAge method within a try-catch block to handle the thrown Exception properly.