# Task D

## Questions

1. Describe the processes and techniques related to object-oriented programming. In your answer, include how to design class with properties and methods and instantiate objects in your applications in Kotlin.
2. Describe the process for developing small-size applications. In your answer, describe the steps of how-to development, e.g., how to start a project, code and debug using an IDE.
3. Explain terms below:

* Syntax language rules, data types structures
* Primitive variables and instance variables
* Class variables
* Polymorphism and inheritance
* Debugging and testing approaches and techniques
* Constructors
* Object aggregation
* Sequence, selection, and iteration constructs
* Organisational documentation

1. Explain design quality metrics such as coupling and cohesion. Explain why coupling is bad and cohesion is preferred.
2. Explain design refinement techniques. How do you detect errors and debug?
3. Describe programming design principles. In your answer, include inheritance, composition and aggregation, encapsulation, and polymorphism.
4. Describe different programming methodologies and various software developmental life cycle options, such as waterfall, spiral, proto typing and agile.

## Answers

1. Designing a class requires you to lists the properties of said class, such as Name, Health point, Damage point, etc. and the method, behaviours which a particular class can do, such as Hit enemy, Defend, Rest, etc.

Using Class Diagram will reduce clutter during designing the class.

Creating a class file during the programming process can be done by right clicking the folder which the Main.kt is saved and choose Kotlin class/File.

Once you have created a class in the class file with its properties and methods, you can instantiate an object in the Main function by setting a variable to the said class.

For example var m = Monster()

If we have set a method to print the Health Point of the first monster in the Monster class

fun displayHealthPoint() {

println(“Monster HP: $healthPoint”)

then you can call the value in Main by using m[1].displayHealthPoint()

1. First, create a Use Case Diagram to learn the potential scope of the application needed. Then, by using Class Diagram, we will be able to design classes that is necessary for the client. Creating the Communication Diagram afterwards will help us to visualise the relation between each class. Fourth, create the Activity Diagram to display the operation of the application.

After the preparation is completed, we can start creating the project file. I would suggest using IntelliJ due to its ease of use and the community version is free. Code in the classes and the main function designed earlier. Do not forget to debug and do a Test Report before shipping the application to the client.

1. Syntax is a set of key words’ arrangement that can be understood by a computer.

Data types is the most common classification of data, e.g int, float, string. They hold the less amount of data than Data structures.

Data structures is a collection of different data types, e,g stacks, queues. They are way more complex than the usual data types.

Primitive variables are Boolean and numeric data types. They constitute the most basic data types.

Instance variables are variables defined in a class, but outside of a method. Usually preceded by an access modifier. Different object can have different instance variables

Class variables are also variables defined in a class, but outside of a method. However, class variables are preceded by static which means all object created from the said class has the same class variable value.

Polymorphism is creating subclasses which have different implementation of the particularly same method. While Inheritance allows for those subclasses to inherit the properties and methods from another class.

Debugging is the process of identifying errors and rectifying them. Printing out variables, and error trapping can be used as a debugging technique. The other common technique is utilising the debugging tool provided by the IDE currently used.

Constructors are special methods that is utilised to create an object.

Object aggregation is when the object created from the superclass are comprised of object from a subclass which cannot exists by itself.

Sequence constructs is when an application works by following a set of instructions from the first to the last.

Selection constructs involves a selection which is facilitated by IF-ELSE statement, or SWITCH-CASE statement.

Iteration constructs involves a repetition which is facilitated by FOR loops

Organisational documentation is the manual of an application created by us.

1. Coupling focuses more on dependencies between parts of an application, while cohesion is about the interconnectedness of each parts of an application. Cohesion is preferred than coupling because an application with a high degree of Cohesion can run better with a missing part than an application with a high degree of coupling. Application that relies on coupling is akin to a chain while an application that relies on cohesion is akin to a spiderweb.
2. Different techniques to produce correct application and simplifying existing application.

Printing out variables, and error trapping can be used as a debugging technique. The other common technique is utilising the debugging tool provided by the IDE currently used.

1. Don’t Repeat Yourself. If there is a code that repeats, put it into a method. If there is a hard-coded value, set it as a public constant.

Encapsulates codes that is bound to be changed in the future.

Classes, methods, or functions should be open for extension and closed for modification.

A Class should only handle one functionality.

High-level modules should not import from low-level modules and abstraction should not depends on details. This helps reduce the coupling of components, allow for a more cohesive application.

Favouring Composition over Inheritance provides higher flexibility and polymorphic application which is more robust. Inheritance also breaks encapsulation because if the superclass is modified, then the subclass will also be affected.

A superclass should be able to be replaced by a subclass without breaking the application.

No code should be forced to depends on a method it does not use.

1. Waterfall model is the earlies SDLC approach. It is a linear-sequential life cycle model. Waterfall mode is used when the project is small.

Spiral model is the combination of sequential and prototype model. Spiral mode is used when the project is large, and the software needs continuous risk evaluation.

Prototyping model is used when the client does not know the exact project requirements beforehand. The developers send small updates to the client to get feedback, which is used to develop the prototype and to be tested by the client.

Agile model is like a combination of waterfall and prototyping. Large projects are separated into smaller parts and those parts will be dealt with in sequential manner, like waterfall model. However, the development of each individual part is similar to prototyping model.