**Exercise 1**

**Pre-Assessment Day Task**

**Encapsulation**

The method by which variables and the code acting on them is contained together, also known as data hiding. When utilised, the variables within a class will be hidden from other classes, and can only be accessed through their current class.

By declaring variables privately, they can only be accessed by specific public get methods, and written by specific public set methods.

This is especially useful when handling sensitive data such as names or financial information.

**Inheritance**

The method by which one class can acquire the methods and fields of another class. The class which inherits the properties of another class is called a subclass or child class, and the class whose properties are inherited is known as a superclass or parent class.

There are different types of inheritance in Java. Single inheritance is when single subclasses inherit the properties of one superclass, so the properties of class A are passed on to class B.

Multilevel inheritance is similar, but with more classes deriving from class B. So, class C inherits from class B, and D from C and so on.

Hierarchical inheritance is when multiple subclasses all inherit their properties from the same superclass, so B C D etc all share inheritance from A.

Multiple inheritance is when one subclass inherits from more than one superclass, and gains the properties of all its parent classes. However, this is not supported by Java with classes, only through interfaces.

Finally, there is hybrid inheritance, which is a combination of multiple types of inheritance.

**Polymorphism**

Polymorphism is a concept in Object Orientated Programming which allows users to perform a single action in multiple ways. One type of polymorphism in Java is runtime polymorphism or dynamic polymorphism, which is where the way the action is performed is decided while the program is running. An example of this is method overriding, which is when a method is declared in a subclass as well as a parent class. This can be useful since the method can be implemented in the subclass without changing the code in the parent class

Another type of polymorphism is compile time polymorphism or static polymorphism. This is where the way an action is performed is decided while the program is compiling. An example of this is method overloading, which is when a single class has multiple methods sharing the same name if their arguments lists are different.

**Abstraction**

The process by which the details and implementation of how the program functions are hidden from the user, where the user only knows *what* the program does and not *how* it does it. This is achieved in Java by using abstract classes and interfaces. Abstract classes can be created by adding the keyword ‘abstract’ in the class declaration. A user could use features of a complex abstract class without changing the way the abstract class works since they wouldn’t be able to access or change it.