OODP Part 1 Theory

- ▼ What are the four basic components of object-oriented model?
  - Objects
  - Messages
  - Methods
  - Classes
- ▼ What are the four main concepts/features of OOM?
  - Abstraction
  - Inheritence
  - Polymorphism
  - Encapsulation/Information hiding
- ▼ What is abstraction?
- ▼ What is encapsulation?
  - Builds a barrier to protect an object's private data.
  - Access to private data can be done through the public methods of the object's class
  - Only need to know what a class does, and how to implement, and do not need to know the implementation details
- ▼ Benefits of encapsulation?
  - Simplicity
  - Extensibility
- ▼ Difference between an object and a class?
  - A class is a blueprint for creating objects. It contains data properties and methods.

- An object is a specific instance of a class. Each object has its own state and behavior.

▼ What is a constructor?

- Used for initialising object data

▼ What are the 2 ways to send message?

- bill.comeDown()
- bill.comeDown(params)

▼ What does the keyword 'this' do?

- It refers to the receiver object with which you call the method (or access the object's variable.
- It is the object reference that stores the receiver object

▼ What is an accessor?

- A get method
- Responsible for returning the value of a data property

▼ What is a mutator?

- A set method
- Responsible for change the values of a data property

▼ What are static variables?

- Created when program starts and destroyed when program stops
- One copy of each class variable per class, irregardless of number of objects created

▼ What about instance variables?

- Created when object is created with new and destroyed when object is destroyed.
- Hold values referenced by more than one method

▼ What is object composition?

- An object can include other object as its data member

- ▼ What are the 4 naming conventions?
    - Lowercase - package
    - Uppercase - constances, enums
    - CamelCase - classes and interfaces
    - Mixed case - methods, variables
- ▼ What is inheritance?
    - Allow us to derive new classes from existing classes by
        - Absorbing their attributes and behaviours
        - Add new capabilities
- ▼ What is method overloading?
    - Happens in the same class
    - When a method is overloaded, it is designed to perform differently when it is supplied with different signatures
    - Same method name but
        - different number of parameters
        - different parameter types
- ▼ Advantage of method overloading?
    - Allows methods to have similar tasks, but differ only in the number of parameters required by the method or the data types of the parameters
    - If no overloading, need to come up with different method names for similar tasks instead of just one
- ▼ What is method overriding?
    - Happens in different class
    - When a subclass alters a method it inherited from a superclass
    - Exactly same signature as the method in the superclass
- ▼ What are the 2 ways of method overriding?

- Refinement
    - Reuse implementation of superclass method with some refinement using the super keyword
- Replacement
    - Replace method completely

▼ How does Java resolve method calls?

- Search for a matching method begins at the class of the object
- If not found, search continues to the immediate superclass
- Proceeds through each immediate superclass until
    - A matched method is found; or
    - No superclass remain, error

▼ What are the 3 types of visibility modifiers?

- public
    - Visible (accessible) anywhere in an application
- private
    - within that class's implementation
- protected
    - methods of the class, methods of subclasses, or any classes in the same package

▼ What is a package?

- Set of classes that are grouped together in the same directory

▼ What is a final method?

- Method cannot be overridden in subclasses

▼ What is a final class?

- Class cannot be superclass. Will not have subclasses.
- Methods in this class is implicitly final

▼ Benefits of final classes and methods?

- Improve security
    - Ensures behaviour of the method will not be changed by subclass
- Improve efficiency
    - Compile-time type checking and binding can be made instead of waiting till runtime

▼ What is abstraction?

- Superclass should contain general features that can be shared by subclasses

▼ What is an abstract class?

- When superclass is too general, and no meaningful object can be created from it

▼ What is an abstract method?

- No implementation, implementation must be provided by the subclass

▼ Does Java support multiple inheritance?

- No. But it supports multiple implementations.

▼ What is an interface?

- Just like an abstract class, except it only contains abstract methods and constants with static final

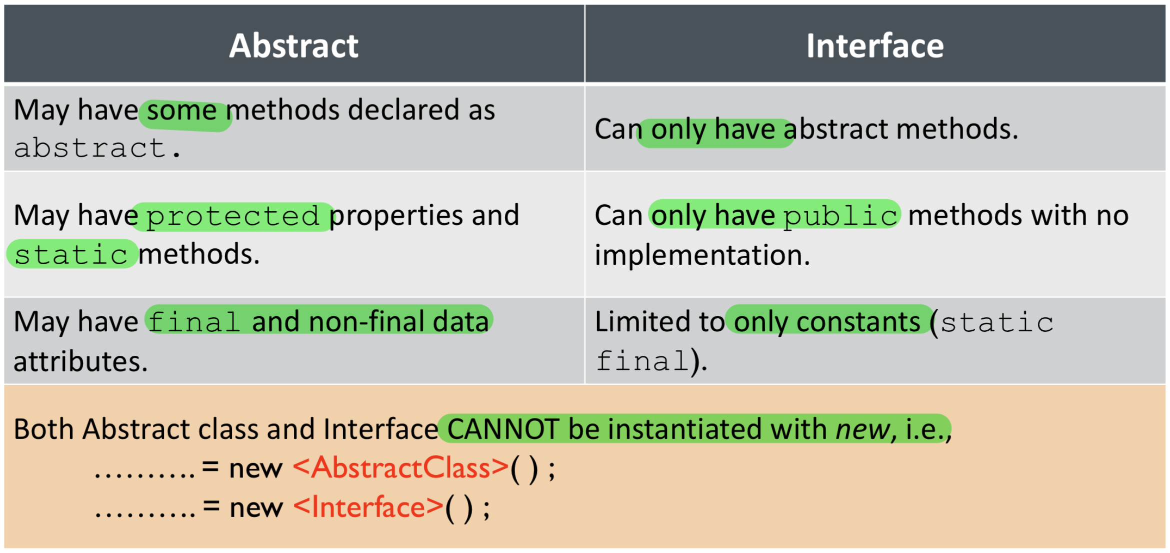
▼ What happens if a subclass do not implement all the abstract methods in an interface?

- It becomes an abstract class

▼ What are interfaces useful for?

- Assigning common functionality to possibly unrelated classes
    - E.g. ship, person, pet and building all have names

▼ Abstract vs Interfaces



Abstract	Interface
May have some methods declared as `abstract`.	Can only have abstract methods.
May have `protected` properties and `static` methods.	Can only have `public` methods with no implementation.
May have `final` and non-final data attributes.	Limited to only constants (`static final`).
Both Abstract class and Interface CANNOT be instantiated with *new*, i.e., ………. = new <AbstractClass>( ) ; ………. = new <Interface>( ) ;	

▼ What does Java Packages do?

Group together java classes into different directories according to their functionality, usability as well as category they should belong to

▼ What is polymorphism?

- Ability of an object reference (superclass) being referred to different types

▼ What is a necessary tool for polymorphism?

- Overriding
    - Subclass can override a method in parent class by defining a method with exactly the same signature and return type
    - Subclass can replace or refine method in the parent class

▼ What is binding?

- Refers to which method to be called at a given time

▼ What are the 2 types of binding

- Static binding
    - Occurs when method call is "bound" at compile time
- Dynamic binding
    - Select of method to be executed is delayed until run time

        - Default for java, except for private, final, static methods

▼ What is upcasting?

- Object of a subclass is assigned to a variable of its superclass

▼ What is downcasting

- Object of superclass is assigned to a variable of its subclass

▼ If downcasting is done incorrectly, what kind of error?

- Run-time error
- Check if downcast is legit
    - object instanceof ClassName

▼ When is downcasting useful?

- To compare one object to another

▼ Benefits of polymorphism?

- Simplicity
    - If you need to write code that deals with a family of types, the code can ignore type-specific details and just interact with the base type of the family
    - Even though the code thinks it is using an object of the base class, the object's class could actually be the base class or any one of its subclasses
    - Easier to write, and easier for others to understand
- Extensibility
    - Programs become extensible
    - Can add new functionality by creating new classes inherited from an off the shelf base class without modifying the base class and the other classes derived from the base class

▼ What are the 3 ways of method overriding?

- Methods of subclass override methods of superclass
- Methods of subclass implements abstract methods of an abstract class
- Methods of concrete class implements the methods of an interface