Date ..../..../....

1a) Class

- A class is a blueprint or template
- It defines the structure and behavior (fields and methods) that the objects created from the class will have.
- It does not consume memory until an object is created from it.
- Define using class keyword.

b) Object

- An object is an instance of a class.
- It is created at runtime and occupies memory.
- You can create multiple objects from one class, each with its own state (data).
- Created using the new keyword.

2a) Instance Variable

- Declared inside a class, outside methods, and without the static keyword.
- Each object gets its own copy.
- Used to store object-specific data in Heap memory
- It belongs to the instance (object)
- It is created when object is created.

Date ..../..../....

b) Static Variable

- Declared using the static keyword
- Shared across all instance of the class.
- Used for class-level data stored in Method area.
- It belong to the class itself.
- It is created at class loading time.

3. a) Class-level Access Modifier

- public - can be accessed from any other class.
- default (no modifire) - accessible only within the same package

Method-level access Modifire

public
private - accessible only within same class
protected - accessible within the same class and subclasses [even if in diff. packages]
default

- Non-access Modifiers

- Static
- final
- abstract
- Strictfp

☆ Private, protected, Static only nested class.

Date ____/____/____

4 a) default

- No keyword used, i.e., the member is not explicitly marked with any access modifier
- Accessible only within the same package.
- Not accessible from outside the package, even by subclasses.

b) protected

- Accessible within the same package.
- Accessible in subclasses, even if those subclasses are in diff. packages.
- More permissive than default, but still not as permissive as public.

5) Default Constructor

- It is a special type of method that takes no arguments.
- It is automatically provided by the Java compiler if no other constructor is explicitly defined in the class.
- If you define any constructor, the default constructor will not be automatically provided unless you explicitly defined.
- In any constructor the 1st line implicitly call to super(), unless you explicitly called another constructor of the superclass.

Date ____/____/____

Purpose

- Initialization of object with default values, i.e, When you create an object of a class using the defaut constructor, Java will intialize the instance variable to their default values.
- In default constructor you can create an object without specifying any arguments.
- In case of inheritance, a subclass may rely on the defaut constructor of a superclass. If the super class has no explicit constructor, the defaut constructor is automatically invoked.
- It can be replaced with a user-define constructor if defined.

6 a) Constructor

- A constructor is used to intialize a new object of a class. It is clled automatically when an object is created.
- The name of the constructor must be same as the class name.
- It does not have a return type, not even void.
- Construct can be over loaded
- It is not inherited by subclass, but a subclass can call a superclass constructor using super().

Date ____/____/____

b) Method

- A method is used to define behaviors or functions that can be called on objects of the class.
- The method name can be anything.
- A method must have a return type (can be void).
- It can be overloaded
- Methods are inherited by subclasses.

7) NO, Constructors do not return any value, not even void. It is a special type of method that is to initialize an object when it is created, no to return any value.

9) Yes, there is a constructor class in Java, but it is a part of the reflection API.
- java.lang.reflect.
- Methods
  - newInstance() - Invoke constructor dynamically
  - getName() - Return name of constructor.
  - getParameterTypes() - Returns the types of parameters accepted by the constructor.

Date ____/____/____

10) Purpose of the constructor class

- The constructor class allows you to be inspect the metadata of constructors (such as their parameters, modifiers, and annotations) at runtime.
- Constructor class enables dynamic instantiation of objects, meaning you can invoke constructors at runtime
- Can be used to access private or protected constrors, even though they would be typically be accessible.
- Many framework (like Spring, hibernate) uses reflection and the constructor class to dynamically instantiate objects.

11) You can't directly access a proted data members outside the package.
You can only access it by using :-

- Inheritance
- Reflection

Date / /

12) a) Abstract Class

- Abstract class is declared with abstract keyword.
- It can have both abstract and concrete method.
- It can have constructors
- It can have instance variable
- It supports single inheritance
- Can have any access modifire

b) Interface

- Interface is declared with Interface keyword.
- It can have default, static and abstract methods.
- Cannot have constructors
- Variable are public static final by default
- It supports multiple inheritance.
- All methods are public abstract by default.

Date ..../..../....

13 a) Abstraction

- Abstraction hides complex implementation details and showing only the essential features of an object.
- It is use to Simplify complexity by hiding internal logic.
- It can be achieved using abstract classes, interfaces or method overriding.
- It focuses on what an object does.

b) Encapsulation

- It wrapps data and methods that operate on that data into single unit, and restricts access to some of the object's components.
- It is used to protect the objects state and ensure it's used correctly.
- It can be achived using access modifiers, getters/setters.
- It focusses on how the object's data is protected, who can access it