Agenda

- Revision
- Singleton class
- Hirerachy(Association & Inheritance)
- super keyword
- Types of Inheritance
- Method Overriding
- ~~polymorphism (DMD)~~
- ~~upCasting~~
- ~~Downcasting~~
- ~~instanceof Operator~~
- ~~final (method,class)~~

Singleton Design Pattern (Demo01)

- It is a desing Pattern
- It alllows to cretae only a single object of the class across the entire program
- Steps for creating singleton class ->
- 1. To make a class as singleton the first step to make all the provided constructors as private.
- 2. Provide a static field of the same type as that of class.
- 3. initialize this field with the class object inside static block
- 4. provide a getter method which will return this singleton object.

Hirerachy

- It is the 4th Major pillar of OOP
- Their are two types of relationships
    - 1. has-a relationship
        - For has-a relationship we use Association
        - eg->
            - Car has-a engine
            - Human has-a heart
            - Room has-a wall
            - Employee has-a vehicle
        - It has two sub-types
            - 1. Composition
            - 2. Aggegration
    - 2. is-a relationship
        - For is-a relationship we use Inheritance
        - eg->
            - Employee is-a Person
            - Circle is-a Shape
            - Car is-a Vehicle
            - Apple is-a Fruit

Association (Demo02)

- If has-a relationship exists between two entities we use association.
- Their are two types of associations
    - 1. Composition
        - If the two entities are tightly coupled we use composition
        - It is also considered as a part-of relationship
        - eg ->
            - heart is part-of Human
            - Wall is part-of Room
            - DateofJoining is part-of Employee
    - 2. Aggegration
        - If the two entities are loosely coupled we use Aggegration.
        - eg ->
            - Room has-a window
            - Employee has-a vehicle

Inheritance (Demo03)

- If is-a relationship exists between two entities then we use inheritance
- In inheritance we have A parent class and a Child class.
- Parent class is also called as Super class and Child class is also called as sub class in java.
- We perform inheritance in java using extends keyword.
- a class can extend only one class
- a class cannot extends multiple classes.
- Multiple implementation inheritance (class inheritance) is not supported in java
- However multiple interface inheritance is allowed in java.

```

```

super keyword

- Super is a keyword in java.
- It is used to invoke the super class methods.
- It is used to invoke the constructor of super class from sub class constructor
- super statement must be the first statement inside the constructor
- super is also used to invoke the super class methods when the methods of superclass are hidden inside the subclass.
- If the superclass and the subclass method is same(method overriding is done) then to invoke the hidden method of superclass inside subclass we should use this super keyword.

Method Overriding

- when we define the superclass method once again in the subclass with same name and signature we call it as method overriding.
- Method overriding is done when
    - 1. super class method is 100% incomplete
    - 2. super class method is partaial complete

    - 3. if we require the implemention of the superclass method inside subclass with complete different requirement
- Rules of method overiding
    - 1. the name of super class and subclass method should be same
    - 2. no of parameters and type of parameters should also be same (signature should be same)
    - 3. The return type of the subclass method can be the subclass type of the return type of superclass method
    - 4. The exception list in subclass method should be same or it can be subset of exception list mentioned for the superclass method.
    - 5. We can change the access modifier for the overriden method in subclass, however it should be of wider visibility type that that of superclass method.
- In java5, the annotation @override was introduced
- this annotation is used to check for the above rules of method overiding.
- using the annotation override is completely optional, however it should be used to avoid human errors at the time of overriding
- As the rules of method overriding is not followed such methods will be termed as new methods of the subclass and not the overriden methods.

Types of Inheritance (Demo04)

```
// Single Inheritance
class A{

}
class B extends A{

}
```

```
// Multiple Inheritance
class A{

}
class B{

}
class C extends A,B {}// NOT Allowed in Java

interface A{

}

interface B{

}

interface C {
```

```
}

class D implements A,B,C // Allowed
{

}
```

```
// Multilevel Inheritance
class A{

}
class B extends A{

}

class C extends B{

}
```

```
// Hirerachical Inheritance
class A{

}
class B extends A{

}
class C extends A{

}
```

- If you mix any two interitances it is hybrid inhetitance

Homework

- Solve the previous assignments if remaning
- Revise the concepts of OOP