26/7/2024 Assignment - 2(a) P. Balanagasimha

C.S.A 0980

Inheritance :-

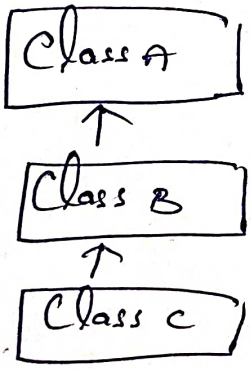
In heritance means creating new classes based an existing ones. Inheritance in Java is a key feature of object-oriented programming that follows one class of inherit the properties (Fields) and behaviours (methods) of another class.

Types of Inheritances :

1. Single Inheritance :- A class inherits from one Superclass

Class A. ⟶ Class B
(parent) (child)

2. multilevel inheritance :- A class is derived from a class, which is also derived from another class

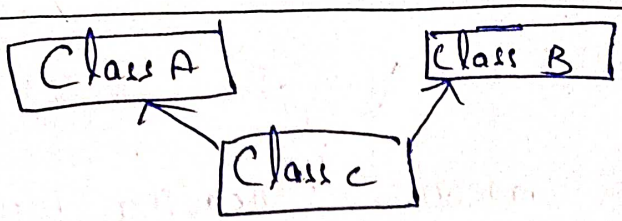


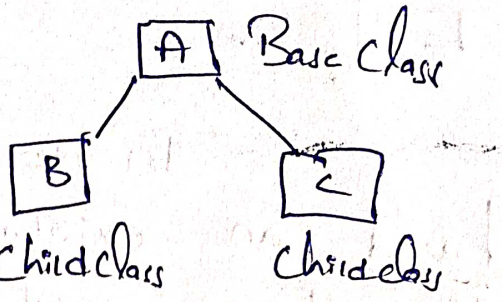
Superclass Ex:- parent

Sub class 1 Ex:- Child

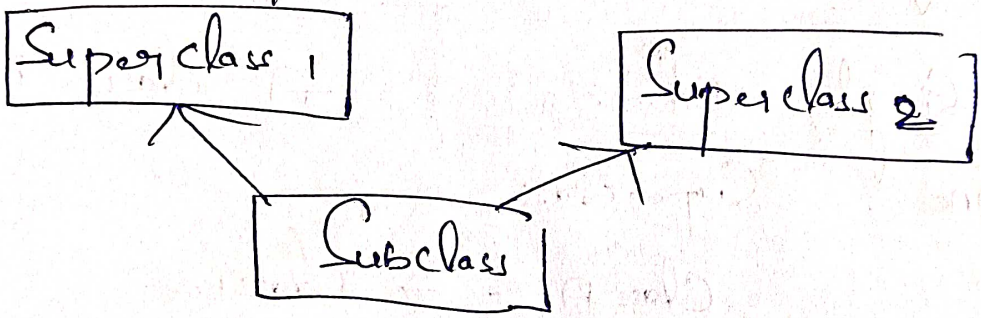
Sub class 2 Ex:- grandchild.

3. Hiegarchial Inheritance :- multilevel classes inherit from a Single Superclass

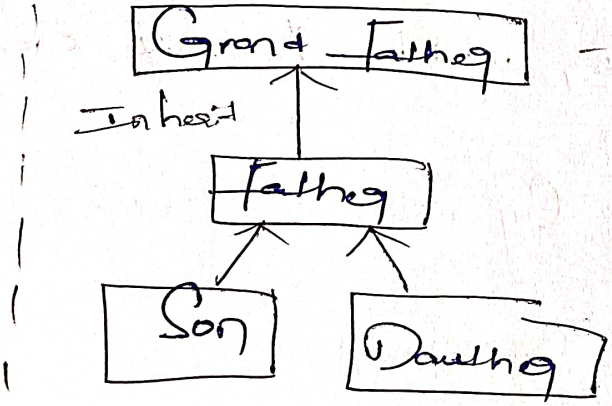




4. Multiple Inheritance :- where a class can inherit properties and methods from more than one Superclass



5. Hybrid Inheritance :- it is a Combination of two or more type of inheritance.



Single Inheritance

```
Class A {
    int a;
    Void display A() {
        System.out.println("a =" +a);
```

```
}
Class B extends A {
    int b;
    Void display B() {
        System.out.println("b = "+b);
    }
Pabic class Single inheritance Example {
    Pubic Static Void main (String [] args){
        B obj = new B();
        obj.a = 20;
        obj.b = 30;
        obj.display A();
        obj.display B();
    }
}
```

output : a=20, b=30

Multilevel inheritance :-

```
Class A {
    Pubic Void display A() {
        System.out.println("Inside display A");
    }
}
Class B extends A {
    Pubic Void display B() {
        System.out.println("Inside display B");
    }
}
```

```
Class C extends B {
    Pubic void display C () {
    System. out. println ("Inside display C");
}
}

    pubic class main {
        Pubic Static void main (String[] args){
        C. obj = new C();
        obj. display A();
        obj. display B();
        obj. display C();
    }
}
```

Heiqaqachial Inhegitance;

```
    Class Animal {
        Void Cat() {
        System. out. println ("this animal eat food");
    }
}
    class. Dog extends Animal {
        Void bark() {
            System. out. println ("The dog barks");
    }
}
    Class Cat extends Animal {
        Void meow() {
            System. out. println ("The cat meaws");
    }
}
    Pubic class main {
        pubic Static void main [String [] args) {
```

```
Dog dog = new dog();
dog.eat();
dog.bark();
Cat cat = new Cat();
Cat.eat();
Cat.meow();
}
}
```

Output :-
This animal eats food
This dog barks
This animal eat food
The Cat meow

Multiple Inheritance :

```
Class A {
Void method A() {
System.out.println("method from class A");
}
}
Interface B {
Void method B();
}
Interface C {
Void method C();
}
Class D extends A implements B,C {
Pubic Void method B() {
System.out.println("method from interface B");
}
pubic Void method() {
System.out.println("method from interface C");
}
}
```

```
Pubic Class multipleinheritance Example {
    Pubic Static Void main (String[] args) {
        D obj = new D ();
        obj. method A();
        Obj. method B();
        obj. method C();
    }
}
```

Output :-

Method from Class A
Method from Interface B
Method from Interface C

Hybrid Inheritance ;

```
Class Grandfather {
    Pubic Void showG () {
        System.out.println ("He is grandfather");
    }
}

Class Father extends Grandfather {
    Pubic Void showF () {
        System.out.println ("He is father");
    }
}

Class Son extends Father {
    Pubic Void show S () {
        System.out.println ("He is son");
    }
}

pubic Class Daughter Extend Father {
    Pubic Void show D () {
```

System.out.println("she is daughter");

}

Public static void main(String args[]){

Son obj = new Son();

obj.showS();

obj.showF();

obj.showG();

daughter obj2 = new daughter();

obj2.showD();

obj2.showF();

obj2.showG();

}

}

Output:

He is son
He is father
He is grandfather
She is daughter
He is grandfather

2. Exception Handling

Exception is an error that occurs during the Execution of program.

Key components of Exception handling

1. Try Block - This is where you write the code that might throw an exception. If Exception occurs, the Execution of try block stop and control is transferred catch block.

2. Catch Block — this is where you handle Exception Block of Code to be Executed Error occurs to try block.

3. Finally Block - This block Contains Code that is Executed Regardless of a whether an Exception was thrown or not.

1. Try. Catch block

```
class main {
    Pubic Static Void main (String[] args);
    try {
        int a = 5/0;
        System.out.println ("Res of Code in try block");
    }
    Catch (Arithmetic Exception e) {
        System.out.println ("Arithmetic Exception => " + e.getMessage());
    }
    Catch (Exception e) {
        System.out.println ("Exception => " + e.getMessage());
    }
}
```

Output: Arithmetic Exception => / by Zero

2. Try - Catch block

```
Pubic class main {
    Pubiic Static Void main (String [] args) {
        try {
            int a [] = {10, 20, 30};
            System.out.println (a[10]);
        }
        Catch (ArrayIndexOutOfBoundsException e)
        {
        System.out.println ("ArrayIndex out of bound
                Exception =>" + e.
                getMessage ());
        }
    }
```

Output: ArrayIndexOutOfBoundsException
=> Index 10 out of bounds for length 3

3) Finally block

```
Class main {
    Pubiic Static Void main (String [] args) {
        try {
            int divideByZero = 5/0;
        }
        Catch (ArithmeticException e) {
```

System.out.println("Arithmetic Exception => "+e.
getMessage());
}
finally {
System.out.println("This is finally block");
}
}

Output:
Arithmetic Exception => / by zero
This is the finaly block.

4) Throw Vote:

Public class main {
static void checkAge(int age) throws
ArithmeticException
{
if (age < 18) {
throw new ArithmeticException("you are
not eligble");
}
else {
System.out.println("you are eligble to vote");
}
}
public static void main(String[] args) {
checkage(15);
}

Output :- / You are not eligble.

Nested try block

```
Public class Excep-test {
    Public Static void main (string args[]){
try {
    int a [] = {1,2,3}
try {
    int b = 1/10;
}
  Catch (Exception e) {
      System.out.println ("Exception thrown" + e.get
                     message());
  }
  System.out.println(a[4]);
}
Catch (ArrayIndexOutOfBoundsException e) {
    System.out.println("Exception" + e.get
                     message());
}
  System.out.println("out of block");
} }
```

Output: Exception thrown / by Zero
Exception thrown. Index 4 out of
band for length 3 out of block