Inheritance

—oop continuation

$$\begin{tikzcd}
\text{Company 1} & \text{Project} \arrow[l] \arrow[dl] & \text{Client 1} \arrow[l] \\
\text{Company 2} & & \text{Client 2} \arrow[ul]
\end{tikzcd}$$

1 & 2

1 p1

```
class Cricket {
    name
    age
    height
    weight
    matches played
    strike rate
    100's
    50's
    wickets
    category
    man_of_the match
}
```

3 cr

6 months

Company 1

2 P2

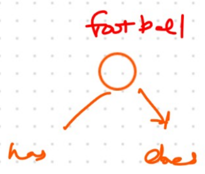
```
class football {
    name
    age
    height
    weight
    matches Played
    goals
    position
    redcard
    penalty
    offside
}
```

3 cr

6 months

Total Duration & ROI

1 year — 6 cr's



Company 2

PC

```
class Human {
    name
    age
    height
    weight
}
```

2 months

2 cr

```
class Cricket {
    matches played
    strike rate
    100's
    50's
    wickets
    category
    man_o the match
}
```

2 cr

3 months

P2

```
class football {
    matches Played
    goals
    position
    redcard
    penalty
    offside
}
```

3 months

$\boxed{6\ \text{cr}}$

Total duration & ROI

8 months : $\underline{10\ \text{cr}}$

Inheritance : It is a mechanism where one object acquires features

{ properties i.e fields & behaviour i.e methods } of the parent object

Types of Inheritance :

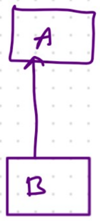
(1) Single

(2) multilevel

(3) Hierarchical

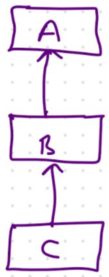
Possibilities

Single



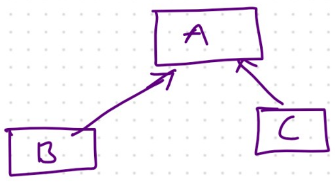
```
class A {
====
}
class B extends A {
...
====
}
```

Multi_level



```
Class A {

}

Class B extends A {

}

Class C extends B {

}
```

Hierarchical

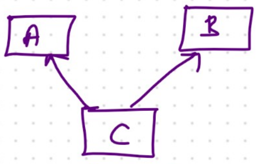


```
Class A {

}

Class B extends A {

}

Class C extends A {

}
```

Not Possible

Multiple

X

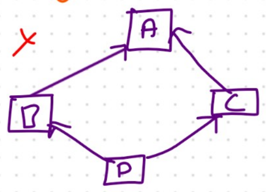


```
Class A {    class B {    class C extends A, B {
}            }            }
```

X

Hybrid

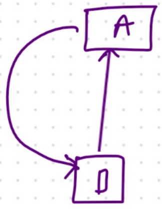
X



```
Class A {

}

Class B extends A {

}

Class C extends A {

}

Class D extends B, C {

}
```

X

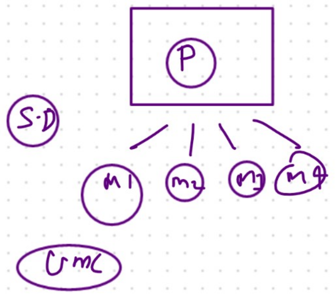
Cyclic



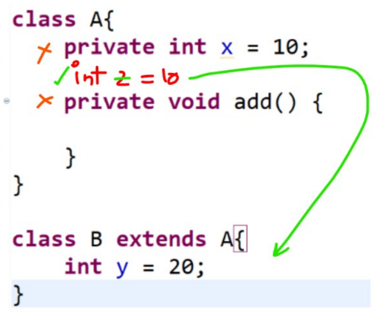
```
class A extends B {

}
class B extends A {

}
```

Rules of Inheritance:

(1) We have to use "extends" keywords to inherit the properties & behaviour from Super class [parent] to Subclass [child class]



(2) "private members" do not participate in heritance



```
class A{
  ✗ private int x = 10;
  ✓int z = 10
  ✗ private void add() {

    }
}

class B extends A{
    int y = 20;
}
```

(3) In case of constructors, those are not allowed to participate in heritance

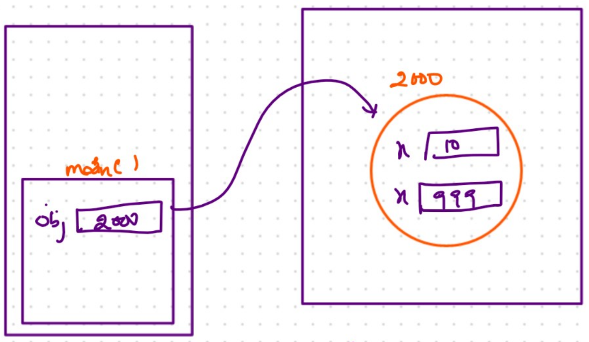
(4) Multiple Inheritance is not allowed for classes in Java

(5) Cyclic Inheritance is not allowed in java

(6) We can have variables [instance variables] of the same name in super class as well as subclass, if we create a object for that subclass then memory will be allocated for both the variables within the object

```
class A {                class B extends A {          class Main {

    int x = 10               int x = 999;                 p s v main(...) {

}                        }                                B obj = new B()

                                                          }
```

Memory Mapping



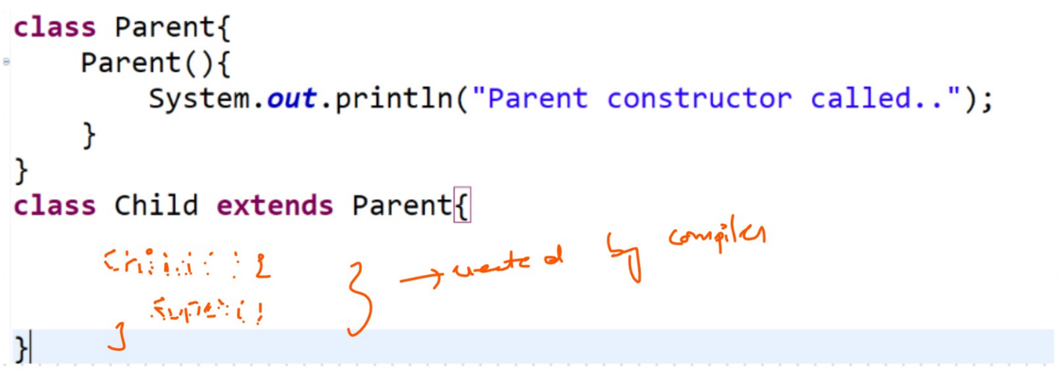
both variables are present

→ to access the variable present in the super class, we have to make use of "super" keyword & to access the variable present in subclass we can make use of "this" keyword

(7) Constructor do not participate in heritance, but the parent class constructor can be called from the child class constructor with the help of super()

```
class Parent {

    Parent() {
        S.O.P("parent Constructor");
    }

}


class Child extends Parent {

    Child() {
        super()
        S.O.P ("Child Constructor")
    }
}
```

(8) If there is no constructor in Child, it will automatically call the default constructor [which is added by compiler] and default super() call is made to parents because it extends parent class



```
class Parent{
    Parent(){
        System.out.println("Parent constructor called..");
    }
}
class Child extends Parent{

    Child(){
      super();
    }
}
```

→ created by compiler