

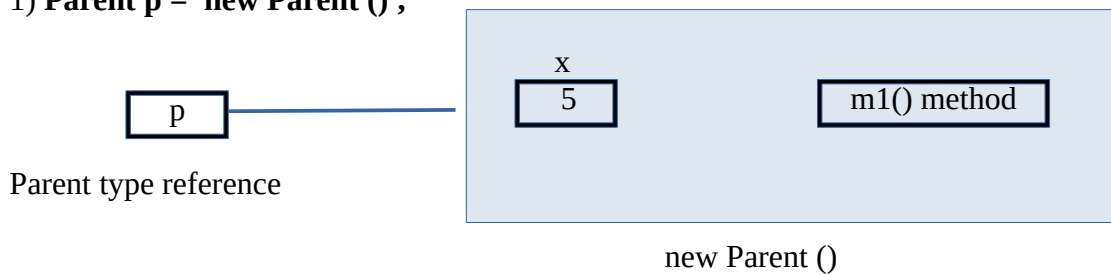
## Object and Reference

```

class Parent
{
int x=10;
void m1()
{
System.out.println("parent method");
}
}
class Child extends Parent
{
int y=20;
void m2()
{
System.out.println("child method");
}
}

```

1) **Parent p = new Parent () ;**



**// Valid operation**

```

system.out.println(p.x) ;
p.m1();

```

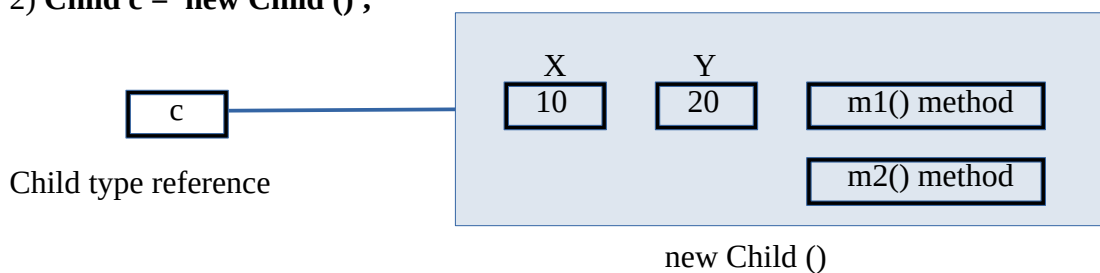
**// Invalid operation**

```

system.out.println(p.y) ;
p.m2();

```

2) **Child c = new Child () ;**



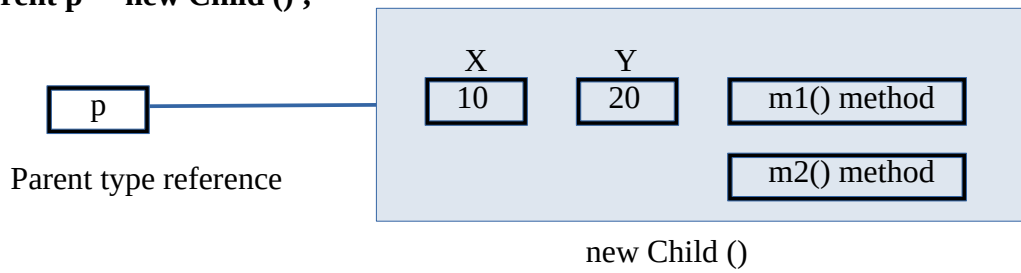
**// Valid operation**

```

system.out.println(c.x) ;
c.m1();
system.out.println(c.y) ;
c.m2();

```

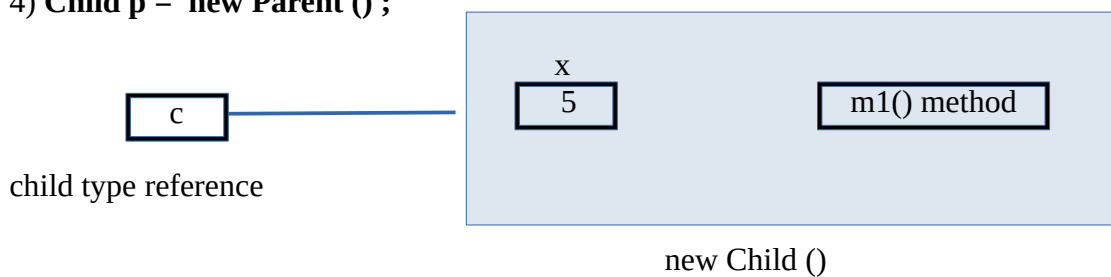
3) **Parent p = new Child () ;**



Point to parent's member only

<p><b>// Valid operation</b>  <code>system.out.println(p.x) ;</code>  <code>p.m1();</code></p>	<p><b>// Invalid operation</b>  <code>system.out.println(p.y) ;</code>  <code>p.m2();</code></p>
--	--

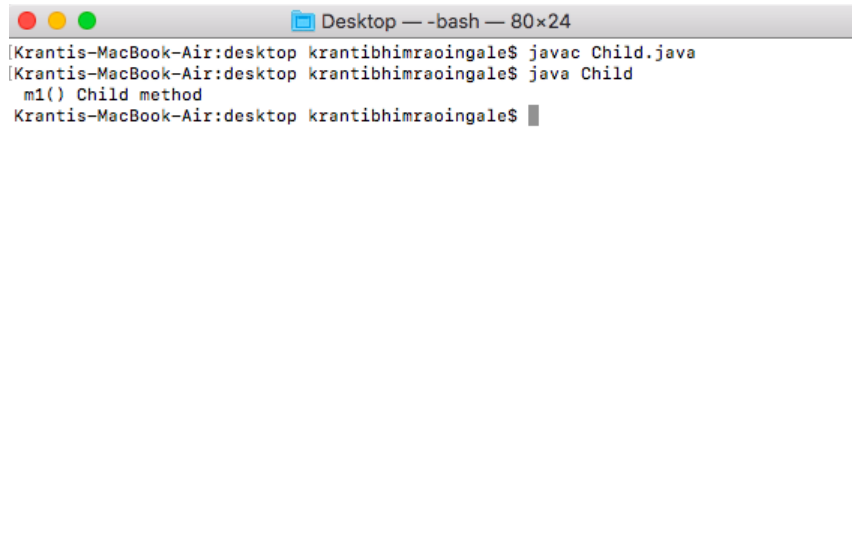
4) **Child p = new Parent () ;**



Not valid as memory for Child class is not allocated.

**Example 1.**

```
class Parent
{
void m1()
{
System.out.println("m1() parent method");
}
}
class Child extends Parent
{
void m1()
{
System.out.println(" m1() Child method");
}
void m2()
{
System.out.println("m2() child method");
}
public static void main(String arg[])
{
Parent P=new Child();
P.m1();
}
}
```

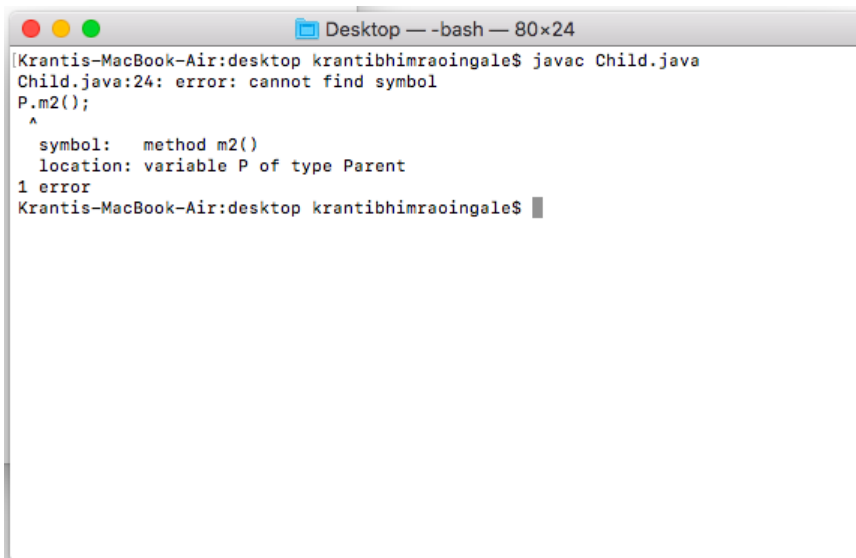
**Output:**

```
Desktop — -bash — 80x24
Krantis-MacBook-Air:desktop krantibhimraoingale$ javac Child.java
Krantis-MacBook-Air:desktop krantibhimraoingale$ java Child
m1() Child method
Krantis-MacBook-Air:desktop krantibhimraoingale$
```

At Compile time compiler checks **m1() of parent** but RunTime Child object is created.therefore **m1() of Child executed.**

**Example 2.**

```
class Parent
{
void m1()
{
System.out.println("m1() parent method");
}
}
class Child extends Parent
{
void m1()
{
System.out.println(" m1() Child method");
}
void m2()
{
System.out.println("m2() child method");
}
public static void main(String arg[])
{
Parent P=new Child();
P.m1();
P.m2();
}
}
```

**Output:**A screenshot of a terminal window titled "Desktop — -bash — 80x24". The terminal shows the command "javac Child.java" being executed. The output is a compilation error: "Child.java:24: error: cannot find symbol", followed by a line of code "P.m2();" with a caret under the 'm', and then "symbol: method m2()", "location: variable P of type Parent", and "1 error". The prompt "Krantis-MacBook-Air:desktop krantibhimraoingale\$" is visible at the bottom.

```
Desktop — -bash — 80x24
Krantis-MacBook-Air:desktop krantibhimraoingale$ javac Child.java
Child.java:24: error: cannot find symbol
P.m2();
  ^
  symbol:   method m2()
  location: variable P of type Parent
1 error
Krantis-MacBook-Air:desktop krantibhimraoingale$
```

At Compile time compiler checks **m2() of parent.** But Parent don't have m2() method there will get Compile Time Error

**Example 3.**

```
class Parent
{
    static void m1()
    {
        System.out.println("m1() parent method");
    }
}

class Child extends Parent
{
    static void m1()
    {
        System.out.println(" m1() Child method");
    }
    public static void main(String arg[])
    {
        Parent P=new Child();
        P.m1();
    }
}
```

**Output:**A screenshot of a terminal window titled "Desktop — -bash — 80x24". The terminal shows the following commands and output:

```
Krantis-MacBook-Air:desktop krantibhimraoingale$ javac Child.java
Krantis-MacBook-Air:desktop krantibhimraoingale$ java Child
m1() parent method
Krantis-MacBook-Air:desktop krantibhimraoingale$
```

static methods are specific to **class** , not with Object . Therefore parent class m1() method called.