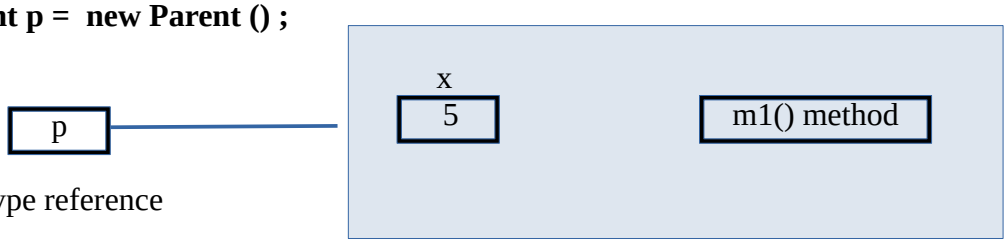
Object and Reference

```
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 () ;**

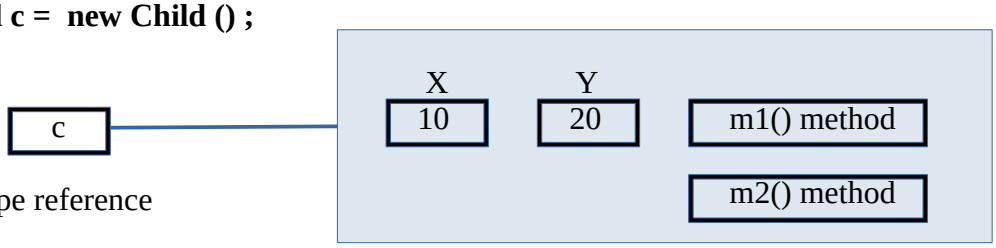


Parent type reference

new Parent ()

// Valid operation	// Invalid operation
system.out.println(p.x) ; p.m1();	system.out.println(p.y) ; p.m2();

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

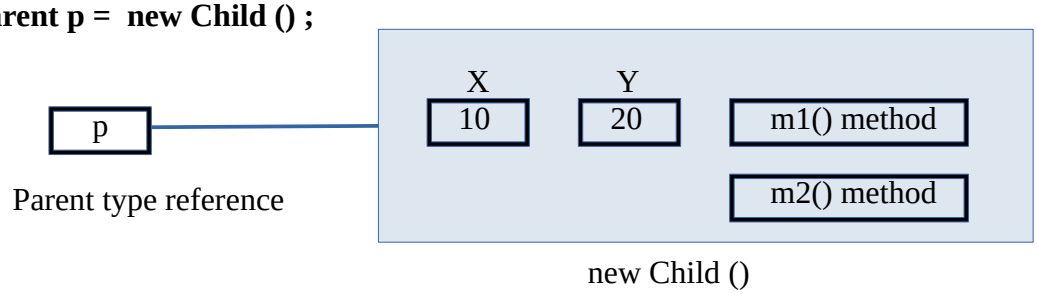


Child type reference

new Child ()

```
// Valid operation
system.out.println(c.x) ;
c.m1();
system.out.println(c.y) ;
c.m2();
```

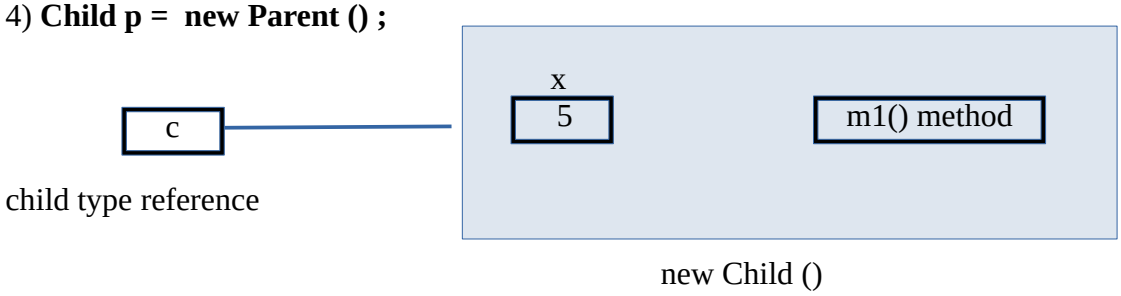
3) **Parent p = new Child () ;**



Point to parent's member only

// Valid operation system.out.println(p.x) ; p.m1();	**// Invalid operation** system.out.println(p.y) ; p.m2();

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

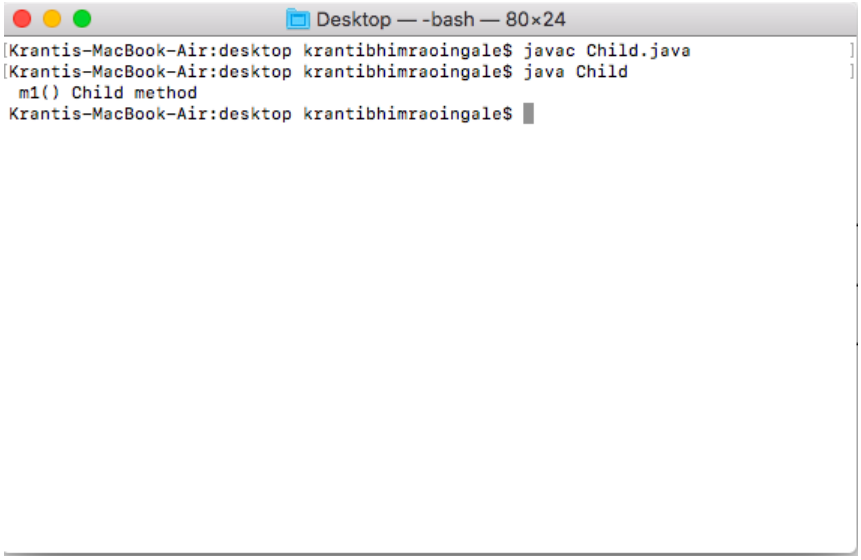


Not valid as memory for Child class is not allocated.

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:

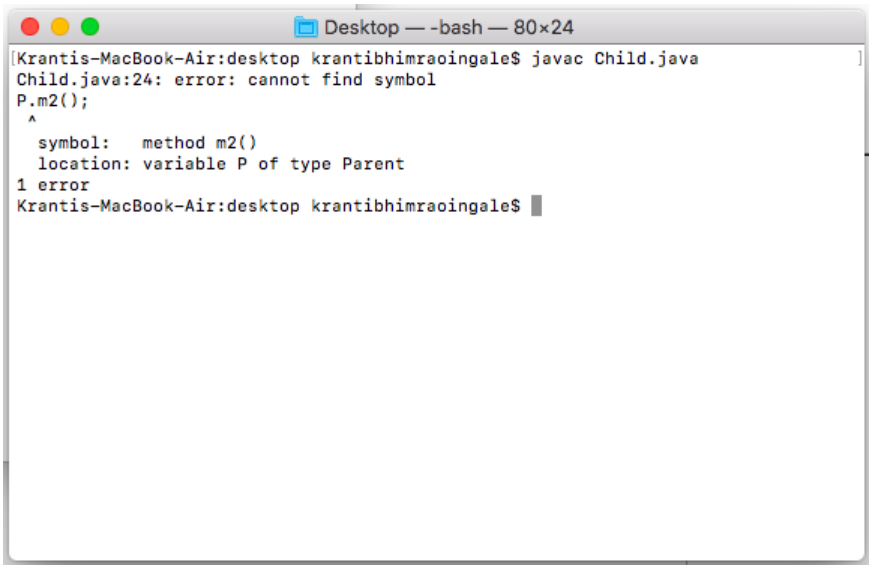


At Compile time compiler checks m1() of parent but RunTime Child object is created.therefore m1() of Child executed.

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:



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

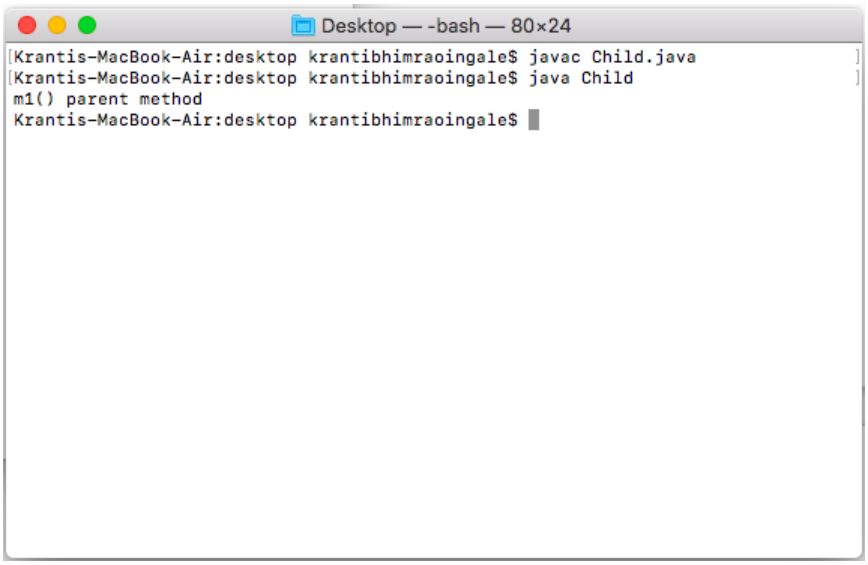
Example 3.

```
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:



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