Abstract Keyword

1) Hiding the internal implementation and highlighting the set of services that process is called abstraction.

Ex:-
a. Bank ATM Screens (Hiding thee internal implementation and highlighting set of services like withdraw, money transfer, mobile registration).
b. Mobile phones (The mobile persons are hiding the internal circuit implementation and highlighting touch screen).
c. Syllabus copy (the institutions persons just highlighting the set of contents that persons provided the persons are not highlighting the whole content).

2) The way of representation the methods are divided into two types
a**) Normal methods** **b) Abstract methods**

a) Normal methods

Normal method is a method which contains declaration as well as implementation.

```
Ex:-    Void m1()
{
---------
--------  body;
---------
}
```

b)Abstract methods

- **The method which is having declaration but not implementations such type of methods are called abstract Method. Hence every abstract method should end with ";".**
- The child classes are responsible to provide implementation for parent class abstract methods.

Ex: - void m1 (); ----------abstract method

Based on above representation of methods , the classes are divided into two types
1) Normal classes **2) Abstract classes**

a) Normal Classes

Normal class is a java class it contains only normal methods**.**

Example :

```
Class Test
{
void m1()
{
........ body;
}
void m2()
{
........ body;
}
.
.
}
```

b)Abstract Classes

- Abstract class is a java class which contains at least one **abstract method.**
- To specify the particular class is abstract and particular method is abstract method , the compiler use abstract modifier.
- For the abstract classes **it is not possible to create an object**. Because it contains the unimplemented methods.
- For any class if we don't want instantiation then we have to declare that class as abstract i.e., for abstract classes instantiation (creation of object) is not possible.
- Abstract class can be inherited .when abstract class is inherited then all abstract methods must
  be overridden in a child class/sub class .

Example :

```
// At least one abstract method
Abstract class Test
{
void m1()
{
........ body;
}
void m2()
{
........ body;
}

Abstract void m3();

}
```

```
// Abstract class may have abstract methods or
may not.
Abstract  class Test
{
void m1()
{
........ body;
}
void m2()
{
........ body;
}
void m3()
{
........ body;
}
}
```

Example 1 :

```
abstract class Test
{
abstract void m1();
abstract void m2();
abstract void m3();
}
class AbstractDemo extends Test
{
void m1()
{
        System.out.println("m1-method");
}
void m2()
{
System.out.println("m2-method");
}
void m3()
{
        System.out.println("m3-method");
}
public static void main(String[] args)
{
AbstractDemo ad=new AbstractDemo();
ad.m1();
ad.m2();
ad.m3();
}
}
```

output:

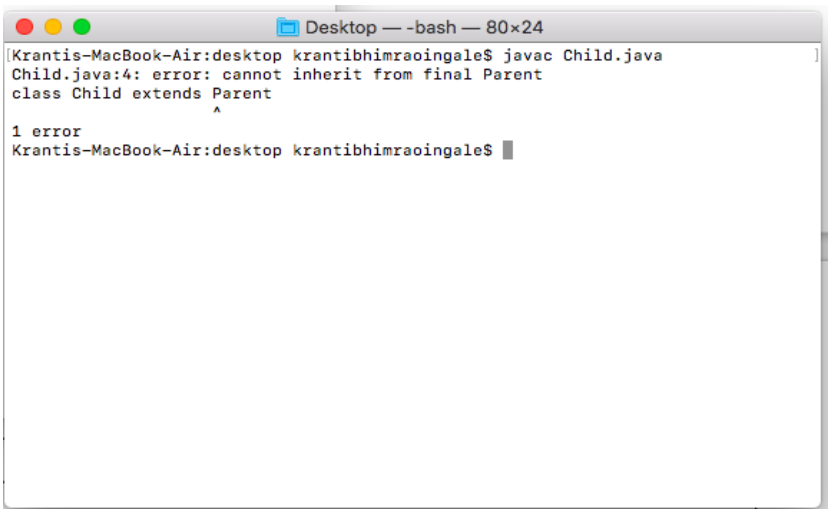
```
Desktop — -bash — 80×24
Krantis-MacBook-Air:desktop krantibhimraoingale$ javac AbstractDemo.java
Krantis-MacBook-Air:desktop krantibhimraoingale$ java AbstractDemo
m1-method
m2-method
m3-method
Krantis-MacBook-Air:desktop krantibhimraoingale$
```

if the child class is unable to provide the implementation for parent class abstract methods at that situation we can declare that class is an abstract then take one more child class in that class provide the implementation for remaining methods.

Example 2:

```
abstract class Test
{
abstract void m1();
abstract void m2();
abstract void m3();
}
abstract class AbstractDemo1 extends Test
{
void m1()
{
        System.out.println("m1-method");
}
void m2()
{
        System.out.println("m2-method");
}
}
class AbstractDemo extends AbstractDemo1
{
void m3()
{
        System.out.println("m3-method");
}
public static void main(String[] args)
{
AbstractDemo ad=new AbstractDemo();
ad.m1();
ad.m2();
ad.m3();
}
}
```

Output:



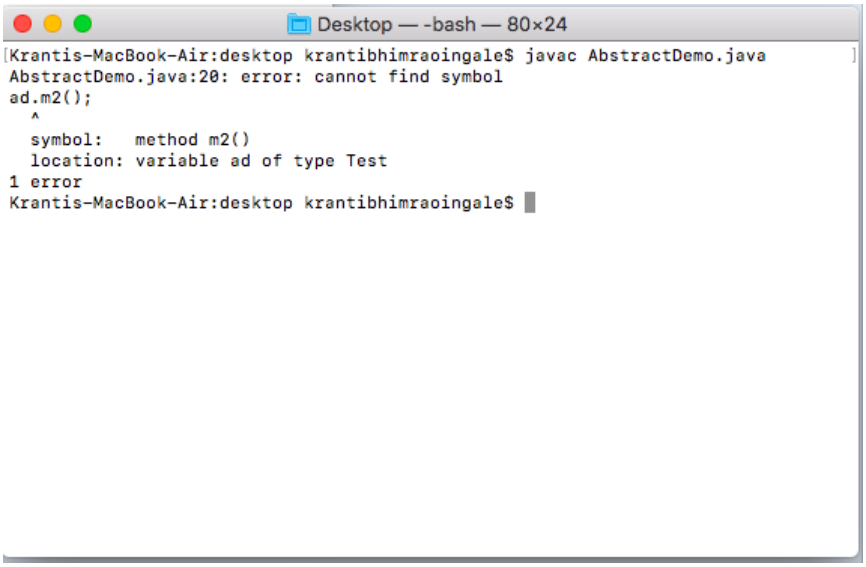
Example 3 :

```
abstract class Test
{
abstract void m1();
}
class AbstractDemo extends Test
{
void m1()
{
        System.out.println("m1-method");
}
void m2()
{
System.out.println("m2-method");
}

public static void main(String[] args)
{
Test ad=new AbstractDemo();
ad.m1();
ad.m2();  // Error

}
}
```

Output:

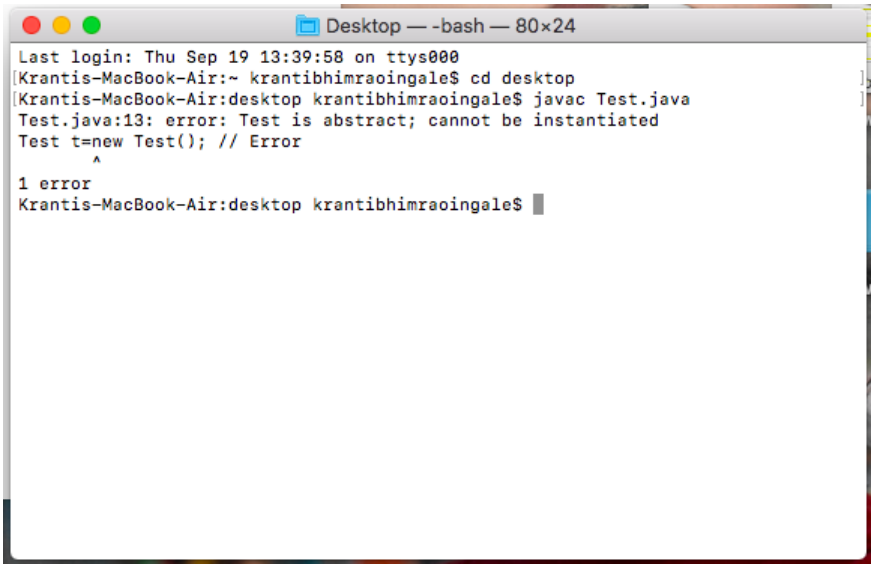


At compilation time Test class reference checks m2() method in Test Class. As m2() is not available will Compilation Error.

Example 4 :

```
abstract class Test
{
        void m1()
        {
        System.out.println("m1-method");
        }
        void m2()
        {
        System.out.println("m1-method");
        }
        public static void main(String[] args)
        {
        Test t=new Test(); // Error
        t.m1();
        t.m2();
        }
}
```

Output:



abstract classes is java class it contains zero number of abstract methods. Even though class does not contain any abstract method still we can declare the class as abstract i.e. abstract class can contain zero number of abstract methods