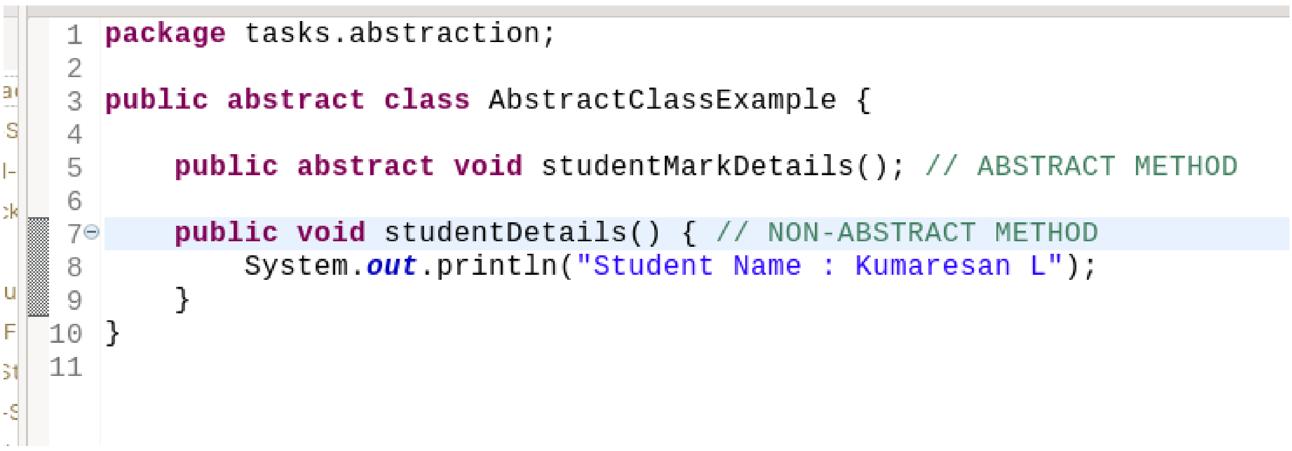
Points to remember:

1-An abstract class can have abstract and/or non-abstract methods.

Ans : Yes, abstract class have abstract and / non abstract methods.

Eg code.,

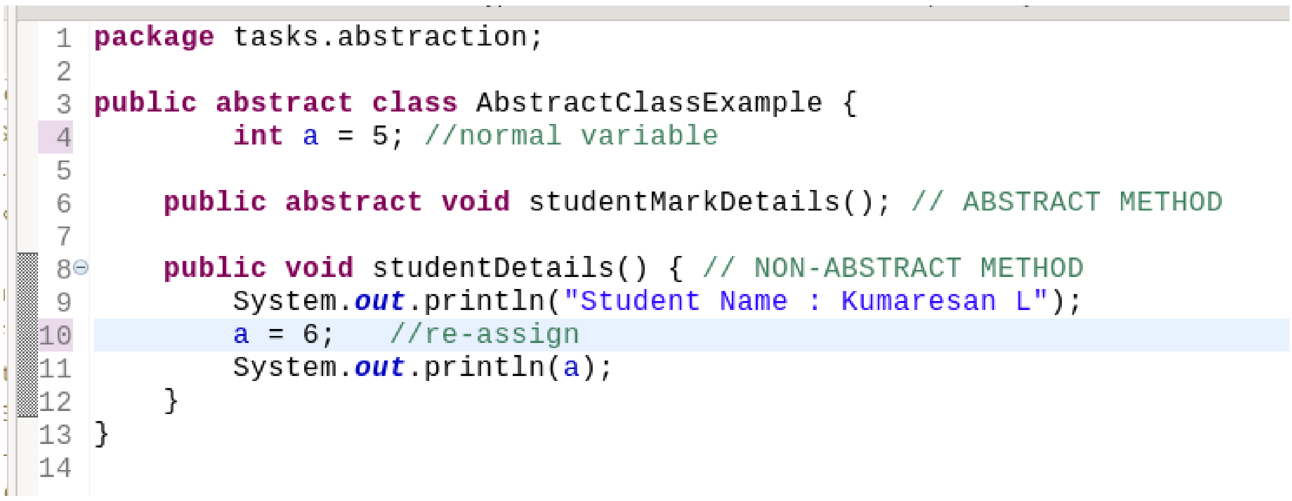


```java
package tasks.abstraction;

public abstract class AbstractClassExample {

    public abstract void studentMarkDetails(); // ABSTRACT METHOD

    public void studentDetails() { // NON-ABSTRACT METHOD
        System.out.println("Student Name : Kumaresan L");
    }
}

```

2-The abstract may or may not contain the final variables.

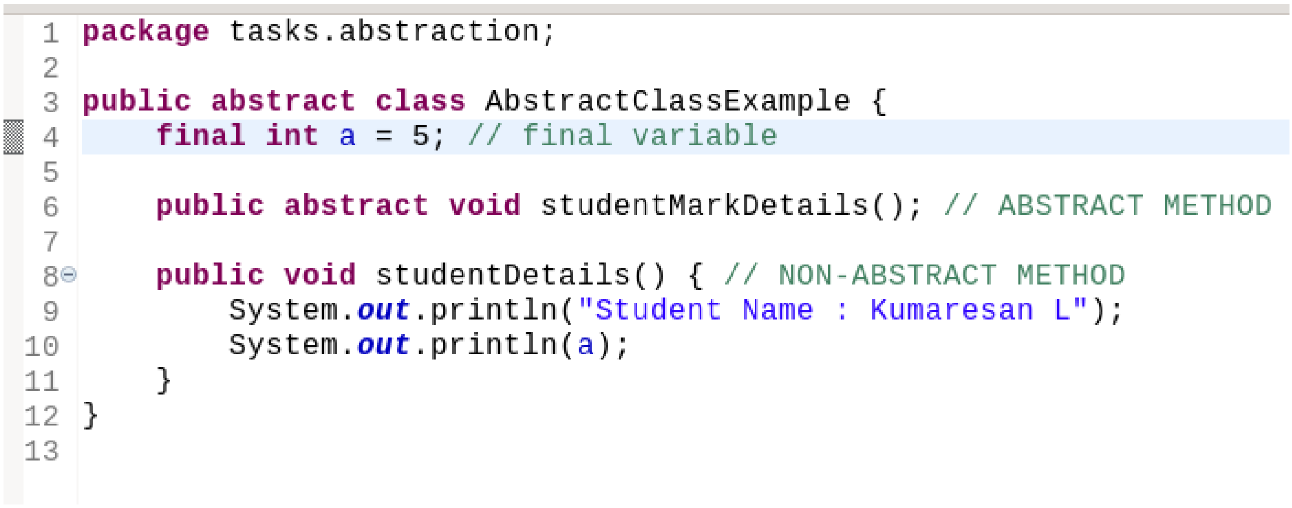
Ans : Yes its true, abstract may contain final variables or may not.

<u>**Eg code., contain normal variable:-**</u>



```java
package tasks.abstraction;

public abstract class AbstractClassExample {
        int a = 5; //normal variable

    public abstract void studentMarkDetails(); // ABSTRACT METHOD

    public void studentDetails() { // NON-ABSTRACT METHOD
        System.out.println("Student Name : Kumaresan L");
        a = 6;    //re-assign
        System.out.println(a);
    }
}

```

<u>Eg code., contain final variables</u>



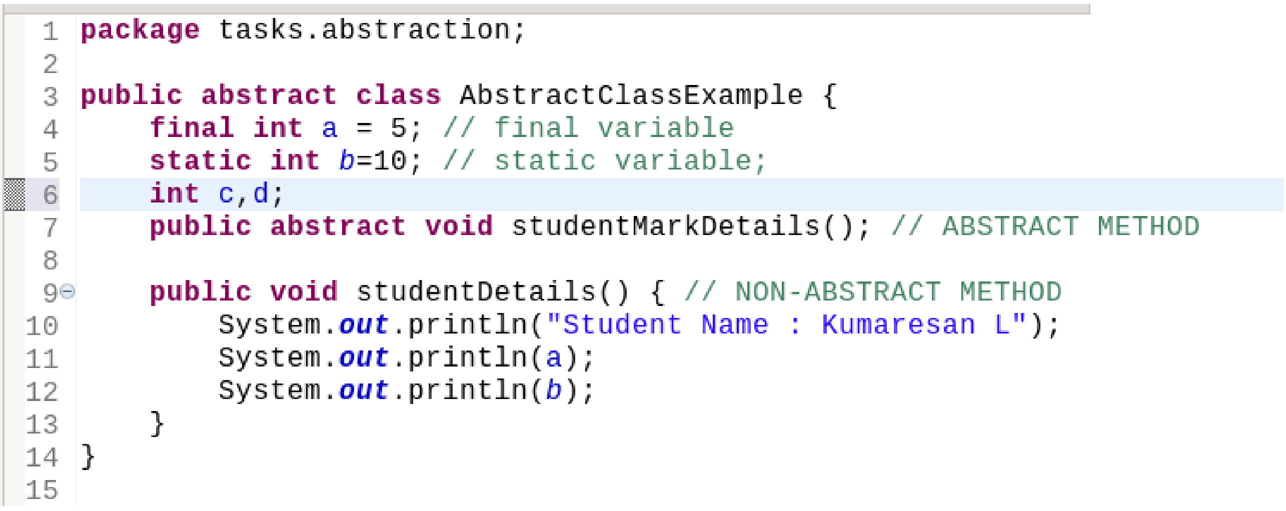
```java
package tasks.abstraction;

public abstract class AbstractClassExample {
    final int a = 5; // final variable

    public abstract void studentMarkDetails(); // ABSTRACT METHOD

    public void studentDetails() { // NON-ABSTRACT METHOD
        System.out.println("Student Name : Kumaresan L");
        System.out.println(a);
    }
}

```

3-An abstract class can have final, static or non-static or non-final variables.

Ans: Above all statements are true

*** Abstract class have final variable and static variable or may not.**

Eg code:-



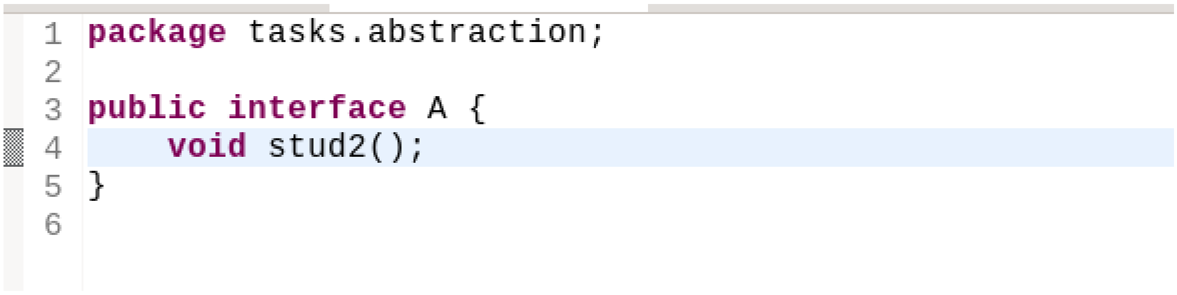
```java
package tasks.abstraction;

public abstract class AbstractClassExample {
    final int a = 5; // final variable
    static int b=10; // static variable;
    int c,d;
    public abstract void studentMarkDetails(); // ABSTRACT METHOD

    public void studentDetails() { // NON-ABSTRACT METHOD
        System.out.println("Student Name : Kumaresan L");
        System.out.println(a);
        System.out.println(b);
    }
}

```

4-An abstract class may provide interface implementation.

Ans : Yes abstract class provide interface implementation but instance of the interface will be not in abstract class, it may be in normal class or in another interfaces.

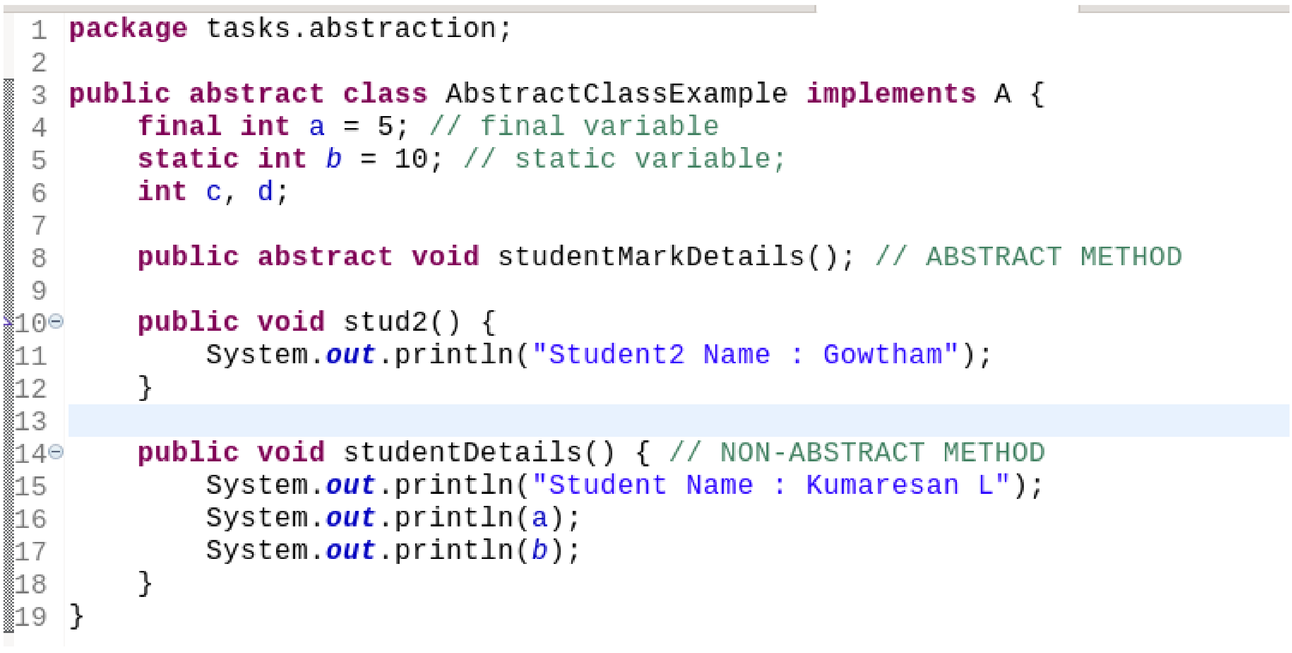
Eg code.,

interface A :-



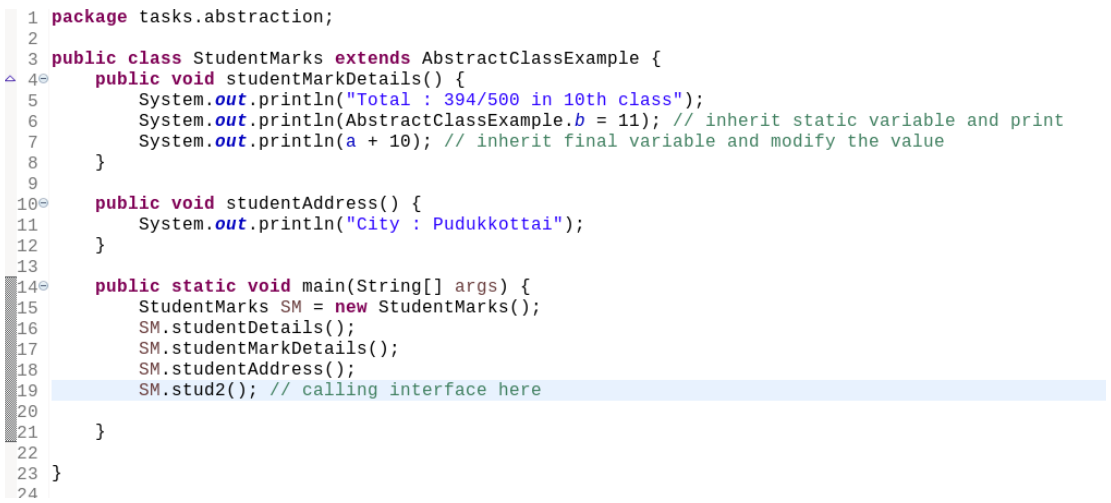
```java
package tasks.abstraction;

public interface A {
    void stud2();
}

```

A implememts in abstract class and also defining stud2() :-



```java
package tasks.abstraction;

public abstract class AbstractClassExample implements A {
    final int a = 5; // final variable
    static int b = 10; // static variable;
    int c, d;

    public abstract void studentMarkDetails(); // ABSTRACT METHOD

    public void stud2() {
        System.out.println("Student2 Name : Gowtham");
    }

    public void studentDetails() { // NON-ABSTRACT METHOD
        System.out.println("Student Name : Kumaresan L");
        System.out.println(a);
        System.out.println(b);
    }
}
```

But calling will be in studentMarks class that inherits AbstractClassExample that implements A so the think happened :-



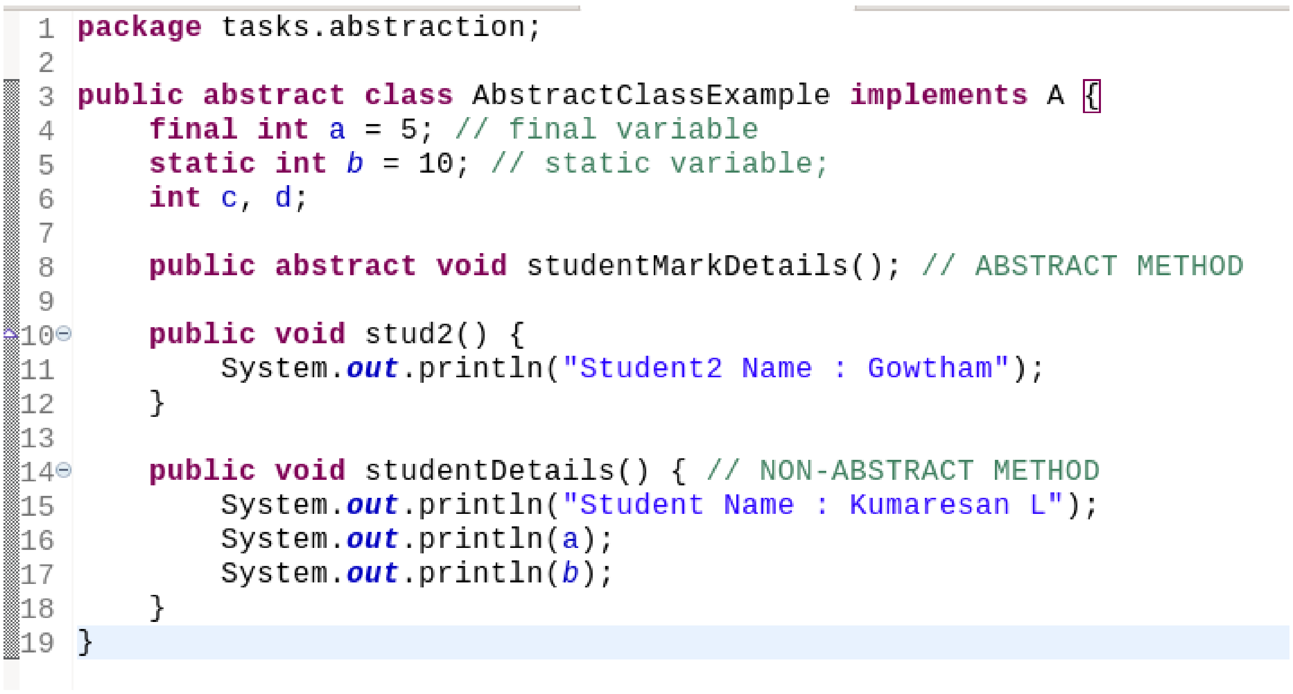
```java
package tasks.abstraction;

public class StudentMarks extends AbstractClassExample {
    public void studentMarkDetails() {
        System.out.println("Total : 394/500 in 10th class");
        System.out.println(AbstractClassExample.b = 11); // inherit static variable and print
        System.out.println(a + 10); // inherit final variable and modify the value
    }

    public void studentAddress() {
        System.out.println("City : Pudukkottai");
    }

    public static void main(String[] args) {
        StudentMarks SM = new StudentMarks();
        SM.studentDetails();
        SM.studentMarkDetails();
        SM.studentAddress();
        SM.stud2(); // calling interface here

    }

}

```

5-An abstract class is inherited using the “extends” keyword.

Ans : Yes abstract class inherited using extends keyword in another class.

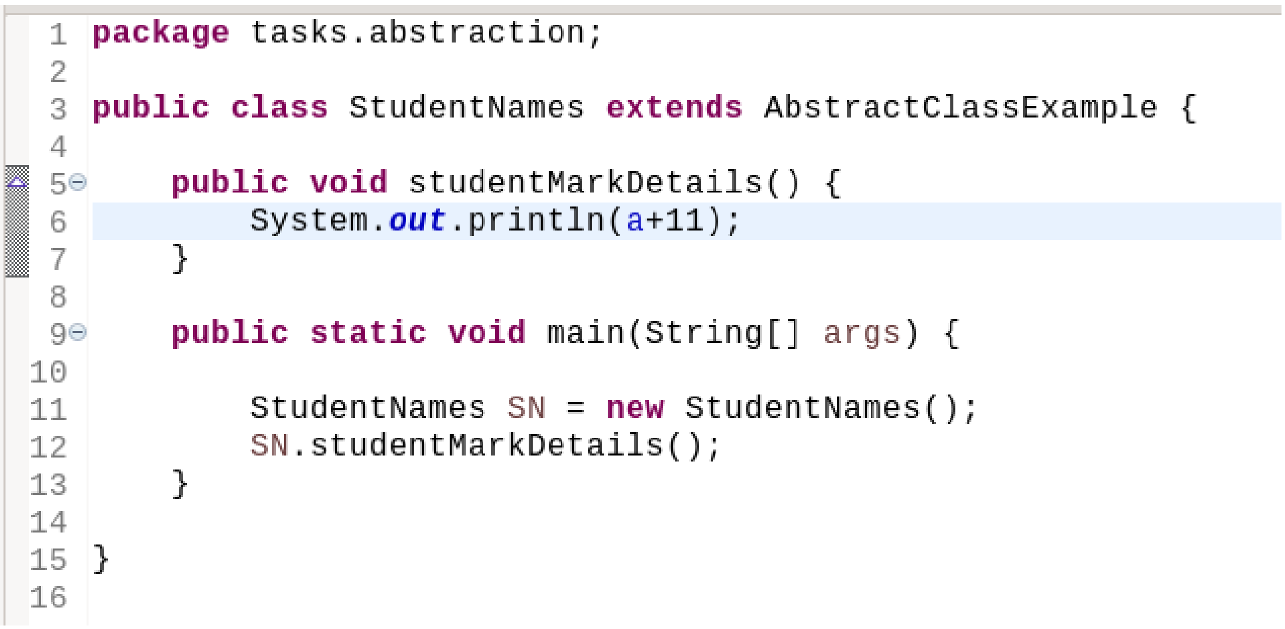
Eg.,

Abstract class :-



```
package tasks.abstraction;

public abstract class AbstractClassExample implements A {
    final int a = 5; // final variable
    static int b = 10; // static variable;
    int c, d;

    public abstract void studentMarkDetails(); // ABSTRACT METHOD

    public void stud2() {
        System.out.println("Student2 Name : Gowtham");
    }

    public void studentDetails() { // NON-ABSTRACT METHOD
        System.out.println("Student Name : Kumaresan L");
        System.out.println(a);
        System.out.println(b);
    }
}
```

That is inherited in StudentNames class :-



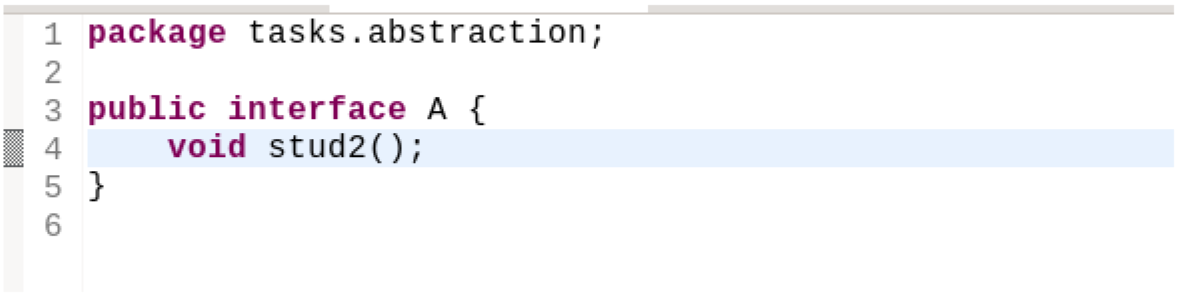
```
package tasks.abstraction;

public class StudentNames extends AbstractClassExample {

    public void studentMarkDetails() {
        System.out.println(a+11);
    }

    public static void main(String[] args) {

        StudentNames SN = new StudentNames();
        SN.studentMarkDetails();
    }

}

```

6-An abstract class can extend other classes or implement multiple interfaces.

Ans : Yes, absrtact class can extend other classes alone. (without interface) and also implement multiple interfaces.

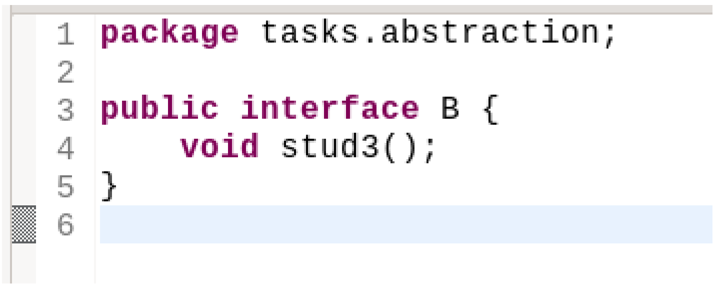
Eg., Abstract class implement multiple interface.

Interface A -



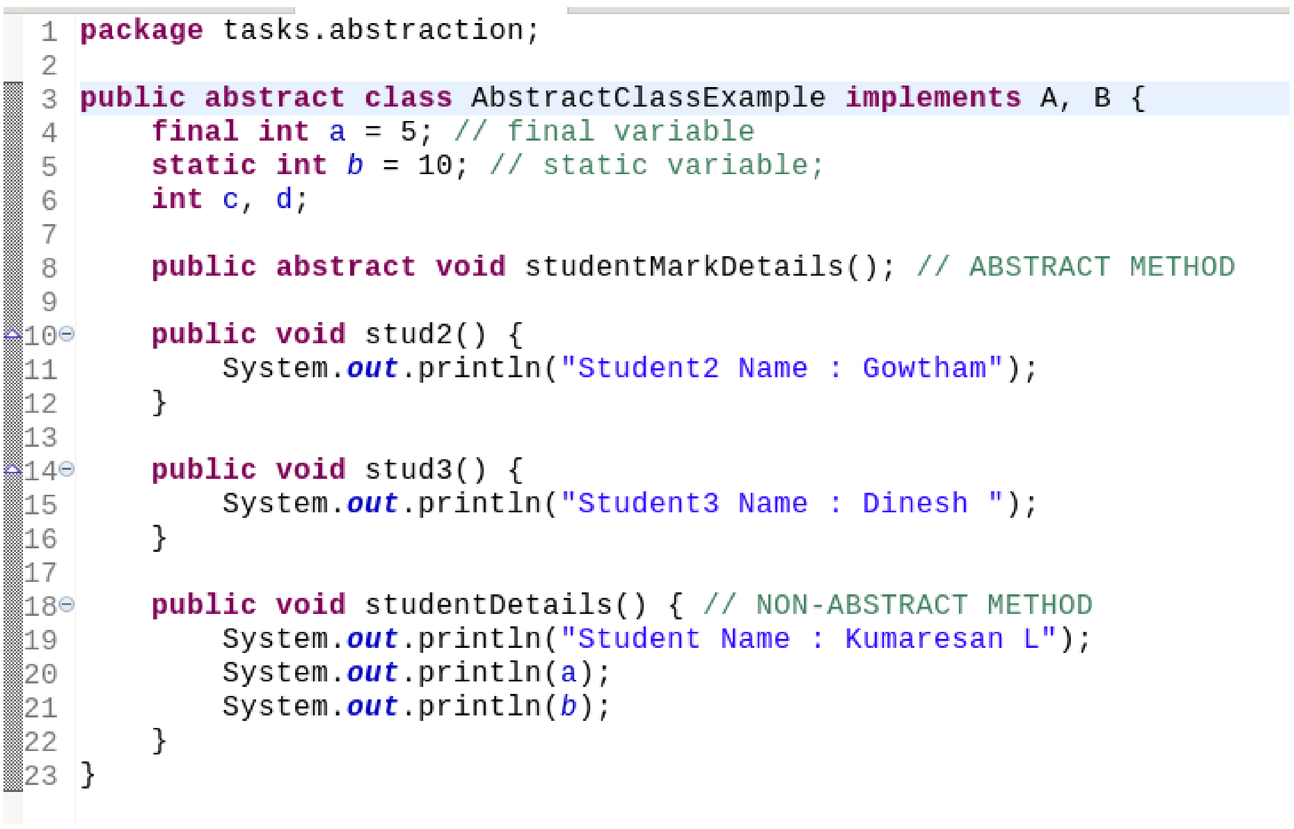
```
package tasks.abstraction;

public interface A {
    void stud2();
}

```

Interface B -



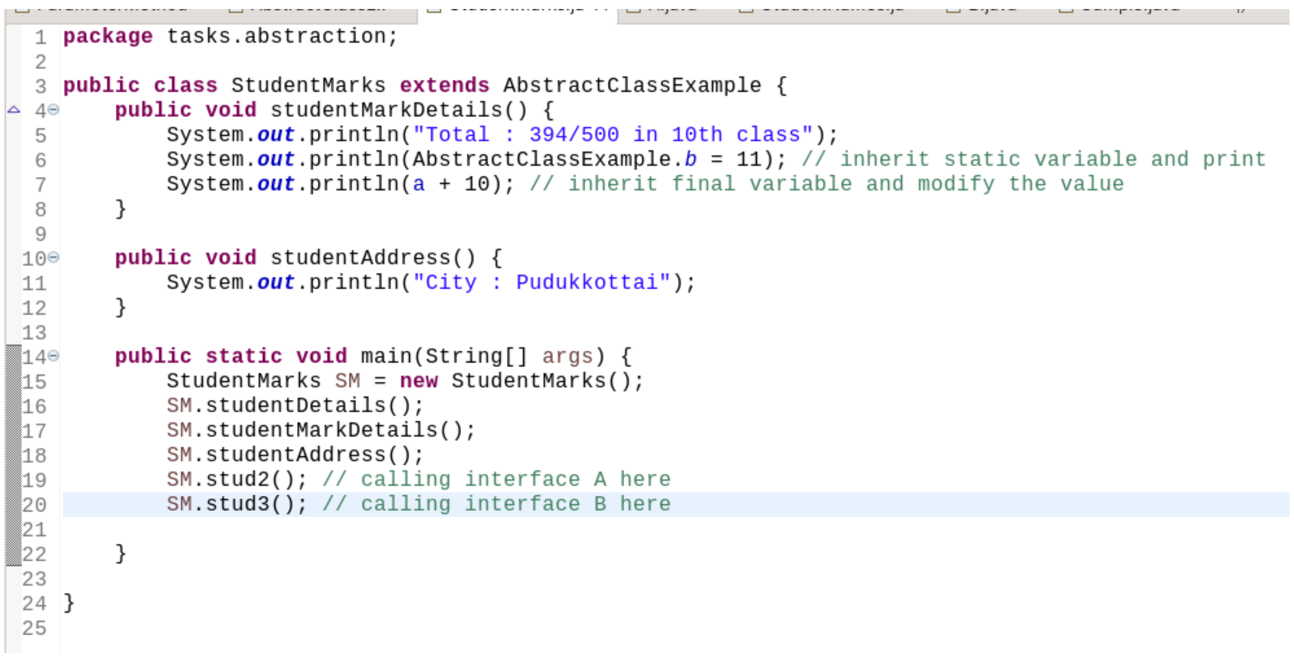
```
package tasks.abstraction;

public interface B {
    void stud3();
}

```

Implements A and B in abstract class named AbstractClassExample -



```java
package tasks.abstraction;

public abstract class AbstractClassExample implements A, B {
    final int a = 5; // final variable
    static int b = 10; // static variable;
    int c, d;

    public abstract void studentMarkDetails(); // ABSTRACT METHOD

    public void stud2() {
        System.out.println("Student2 Name : Gowtham");
    }

    public void stud3() {
        System.out.println("Student3 Name : Dinesh ");
    }

    public void studentDetails() { // NON-ABSTRACT METHOD
        System.out.println("Student Name : Kumaresan L");
        System.out.println(a);
        System.out.println(b);
    }
}
```

But calling will be in studentMarks class that inherits AbstractClassExample that implements A and B so the think happened :-

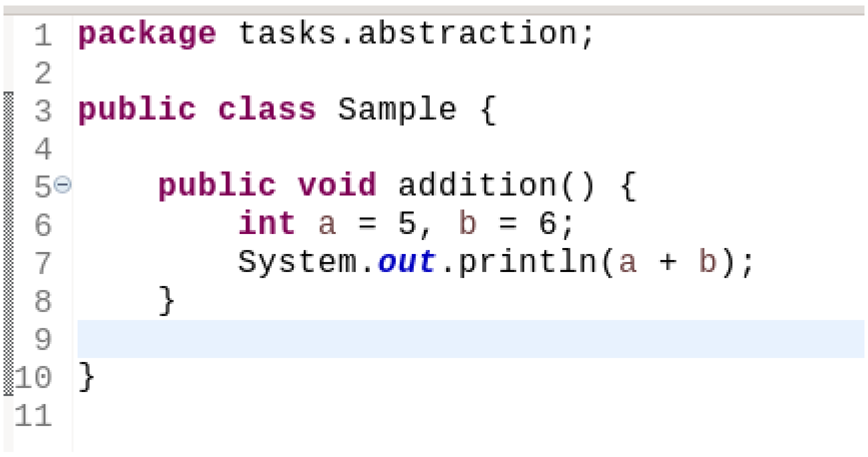


```java
package tasks.abstraction;

public class StudentMarks extends AbstractClassExample {
    public void studentMarkDetails() {
        System.out.println("Total : 394/500 in 10th class");
        System.out.println(AbstractClassExample.b = 11); // inherit static variable and print
        System.out.println(a + 10); // inherit final variable and modify the value
    }

    public void studentAddress() {
        System.out.println("City : Pudukkottai");
    }

    public static void main(String[] args) {
        StudentMarks SM = new StudentMarks();
        SM.studentDetails();
        SM.studentMarkDetails();
        SM.studentAddress();
        SM.stud2(); // calling interface A here
        SM.stud3(); // calling interface B here

    }

}
```

Eg., An abstract class can extend other classes -

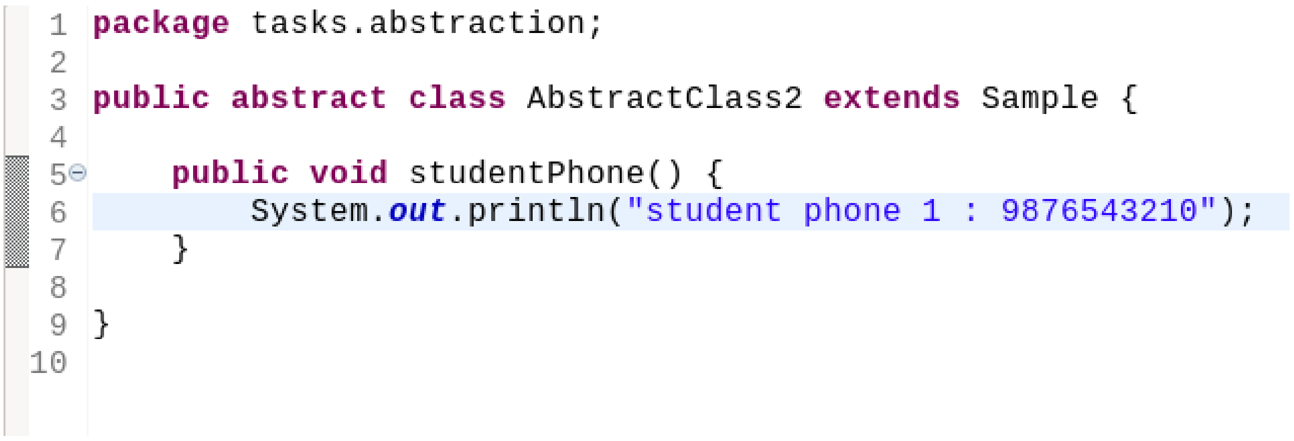
Now, here abstract class extends sample class:

sample.java -



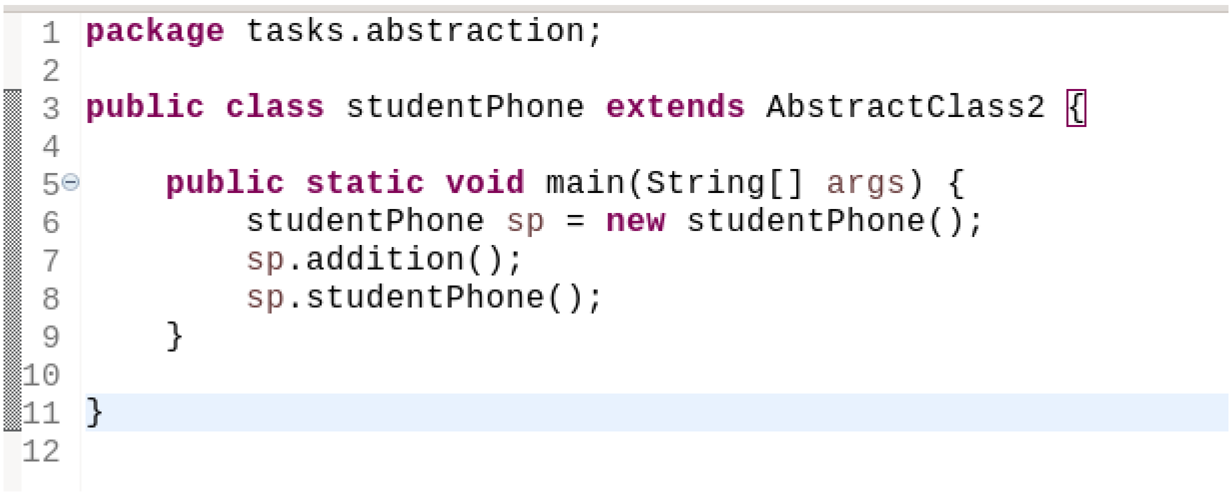
```
package tasks.abstraction;

public class Sample {

    public void addition() {
        int a = 5, b = 6;
        System.out.println(a + b);
    }

}

```

Abstract class named AbstractClass2 extends Sample class:



```
package tasks.abstraction;

public abstract class AbstractClass2 extends Sample {

    public void studentPhone() {
        System.out.println("student phone 1 : 9876543210");
    }

}

```

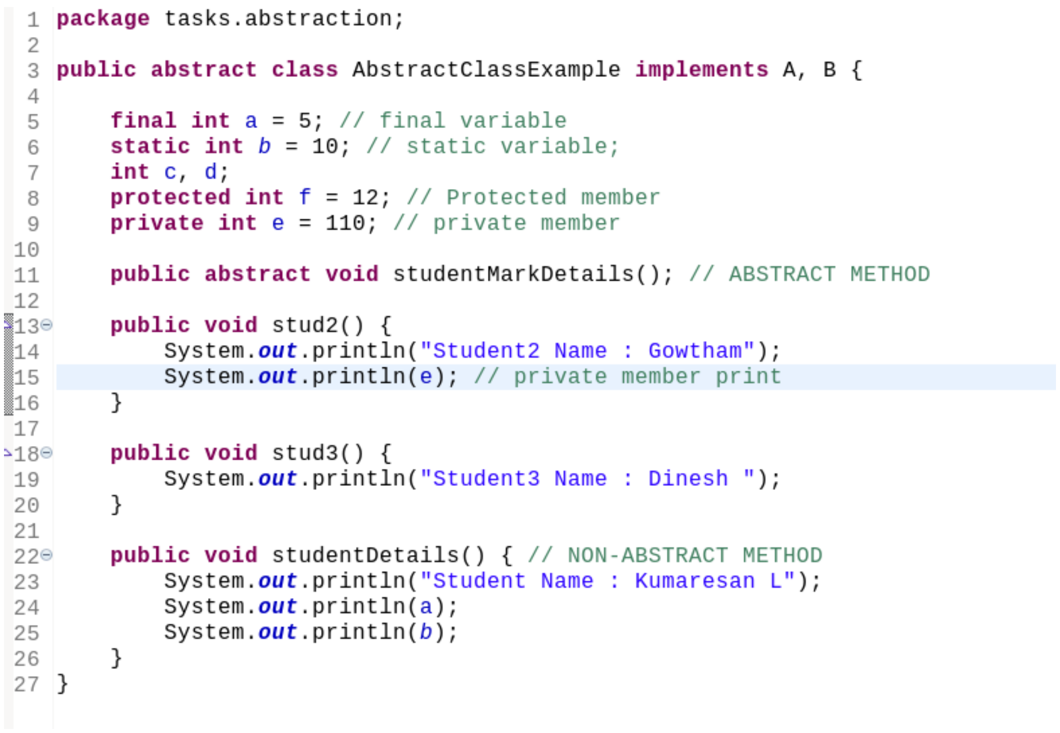
But calling will be in studentPhone class that inherits AbstractClass2 that extends Sample so the think happened :-


```java
package tasks.abstraction;

public class studentPhone extends AbstractClass2 {

    public static void main(String[] args) {
        studentPhone sp = new studentPhone();
        sp.addition();
        sp.studentPhone();
    }

}
```

7-An abstract class can have private or protected data members apart from public members.

Ans :- Yes, both the above statements are true. Abstract class can have private or protected members.

Eg.,


```java
package tasks.abstraction;

public abstract class AbstractClassExample implements A, B {

    final int a = 5; // final variable
    static int b = 10; // static variable;
    int c, d;
    protected int f = 12; // Protected member
    private int e = 110; // private member

    public abstract void studentMarkDetails(); // ABSTRACT METHOD

    public void stud2() {
        System.out.println("Student2 Name : Gowtham");
        System.out.println(e); // private member print
    }

    public void stud3() {
        System.out.println("Student3 Name : Dinesh ");
    }

    public void studentDetails() { // NON-ABSTRACT METHOD
        System.out.println("Student Name : Kumaresan L");
        System.out.println(a);
        System.out.println(b);
    }
}
```