Abstract:

The abstract is the modifier applicable for classes and methods but not for variables.

Abstract method

- A method declared as abstract and did not have an implementation (body) is known as an abstract method. i.e. abstract method can have only method declaration but not implementation. Hence every abstract method declaration should compulsorily end with ;(semicolon).
- The child class is responsible for providing the implementation of parent class abstract methods.
- By declaring abstract methods in parent class, we define guidelines for the child classes, which describe the methods to be compulsorily implemented by the child class.

```
abstract void addData(int x,int y);
```

Example abstract method

```
1. abstract void display();   //valid-only declaration, nobody defined.
2. abstract void display()  { }  //invalid
```

Abstract class in Java

- A class that is declared with an abstract keyword is known as an abstract class in java.
- It can have an abstract method (method without body) and non-abstract methods (method with the body).
- It needs to be extended and its method should be implemented.
- It cannot be instantiated.

Example abstract class

```
1. abstract class A{}
```

Case 1:

```
1
2  abstract class Upes {
3      public static void main(String args[])
4      {
5  // Creating object of class inside main() method
6          Upes u =new Upes();
7      }
8  }
```

Output:

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.22000.778]
(c) Microsoft Corporation. All rights reserved.

D:\1 UPES Data aug 2022\Java CSF\abstract>javac Upes.java
Upes.java:6: error: Upes is abstract; cannot be instantiated
                Upes u =new Upes();
                       ^
1 error
```

Case 2:

```
1
2   class Upes
3  {   // Method of this class
4      public void methodOne();
5  }
```

Output:

```
D:\1 UPES Data aug 2022\Java CSF\abstract>javac Upes.java
Upes.java:4: error: missing method body, or declare abstract
   public void methodOne();
               ^
1 error
```

Case 3:

```
 class Upes
{   /// Method of this class
    public abstract void method1() {}
}
```

Output:

```
D:\1 UPES Data aug 2022\Java CSF\abstract>javac Upes.java
Upes.java:2: error: Upes is not abstract and does not overrid
e abstract method method1() in Upes
 class Upes
 ^
Upes.java:4: error: abstract methods cannot have a body
   public abstract void method1() {}
                        ^
2 errors
```

Case 4:

```
 class Upes
{   // Method of this class
    public abstract void method1();
}
```

Output:

```
D:\1 UPES Data aug 2022\Java CSF\abstract>javac Upes.java
Upes.java:2: error: Upes is not abstract and does not overrid
e abstract method method1() in Upes
 class Upes
 ^
1 error
```

Example:

```
abstract class Parent
{
        public abstract void m1();
        public abstract void m2();
}
class TestChild extends Parent
        {
        public  void m1(){   //body of m1()}
        }
```

Output:

```
F:\Java Code 2020>javac TestChild.java
TestChild.java:6: error: TestChild is not abstract and does
not override abstract method m2() in Parent
class TestChild extends Parent
^
1 error
```

We can handle this error either by declaring the child class as abstract (case 1) or by providing an implementation for m2() (case 2).

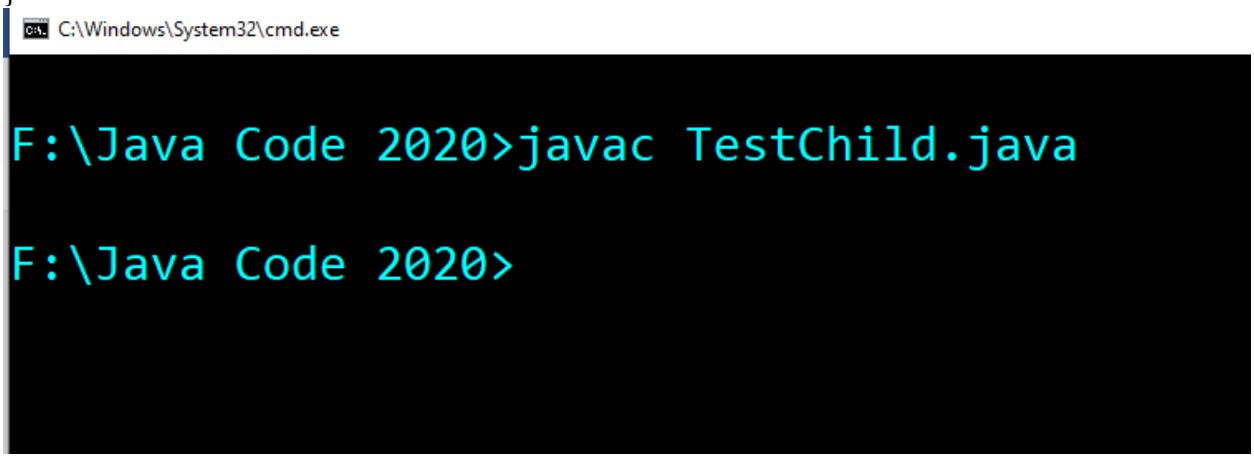
case 1:

```
abstract class Parent
{
public abstract void m1();
public abstract void m2();
}
abstract class TestChild extends Parent
{
public  void m1(){}
}
```

case 2

```
abstract class Parent
{
public abstract void m1();
public abstract void m2();
}
 class TestChild extends Parent
{
```

```
public  void m1(){}
public  void m2(){}
}
```



Example of an abstract class that has an abstract method

In this example, Bike is the abstract class that contains only one abstract method run. It implementation is provided by the Honda class.

```
1.  abstract class Bike
2.  {
3.  abstract void run();  //abstract method
4.  void speed(){// non abstract method
5.  }
6.  }
7.
8.  class Honda extends Bike
9.  {
10. void run()
11. {
12. System. out.println("Overridden Method run()");
13. }
14. public static void main(String args[])
15. {
16.  //Bike obj = new Bike(); //compile time error
17.  Honda obj= new Honda()
18.  obj.run();
19. }

}
```

Output:

```
Overridden Method run()
```

Abstraction in Java

Abstraction is a process of hiding the implementation details and showing only functionality to the user. Another way, it shows only important things to the user and hides the internal details for example sending an email, you just type the text and send the message. You don't know the internal processing of the message delivery.

Ways to achieve Abstraction

There are two ways to achieve abstraction in java

1. Abstract class (0 to 100%)
2. Interface (100%)

Examples:

```
abstract class  A—0%
{
m1(){stmts…..}—0%
m2(){stmts…..}—0%
m3() {stmts…..}--0%
m4() {stmts…..}--0%
m5(){stmts…..}--0%
}
```

```
abstract class A—60%
{
m1(){stmts…..}—0%
m2(){stmts…..}—0%
abstract m3();--20%
abstract m4();--20%
abstract m5();--20%
}
```

```
abstract class A—100%
{
abstract m1();—20%
abstract m2();—20%
abstract m3();--20%
abstract m4();--20%
abstract m5();--20%
}
```

Note: The use of abstract methods, abstract class and interfaces are recommended, and it is always good programming practice.