303626-2-6E AID:258164 | 22/08/2020

**Depreciation:**

Abstraction separates code into interface and implementation. So while design the component, you must keep interface independent of the implementation so that if changes are underlying implementation then interface would remain intact.

In this case whatever programs are using these interfaces, they would not be impacted and would just need a recompilation with the latest implementation.

For example ,in database system , hide the details of how data is to stored and create and maintained.

**Abstraction:** Abstraction mean hide some data .It refers to act of showing the essential features or data without involving there main background details or explanation. Class uses list of abstract attributes. Abstraction is eliminate the unwanted irrelevant of the essentials.

It hide implementation details while just presenting the features to the outside world. The main advantage is reduce code complexity , hiding details and exposing the essential parts.

It focus on what the object does ,outer layout used in terms of design.

Abstract modifier can be applicable for method ,class and variable.

**Abstract class:** It is one whose instances cannot be created. Any class that has at least one abstract method has to be compulsory declared as an abstract class. Abstract class can contain both abstract and non abstract method.

**Abstract method:** Abstract method has only declaration not definition. It has no body. It can declared as “ abstract void m(); ” . Child class needs to override the definition of all abstract method.

**Example of abstract method :**

Class parent1

{

Abstract void m1();

{

System. out. print(“parent1”);

}

}

Class child1 extents parent1

{

Void m1()

{

System. out. print(“child1”);

}

}

Class abstract test

{

Public static void main.(string arg[])

{ child1 c1 = new child1();

c1 . m1();

c1 . m2();

}

}

**Output of code :**

child1

parent 1