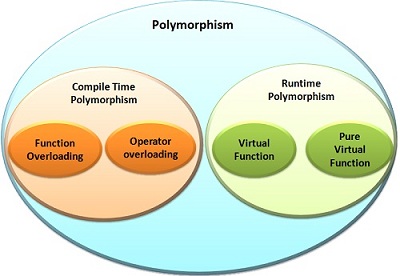
Difference Between Virtual and Pure Virtual Function

April 30, 2016 [4 Comments](https://techdifferences.com/difference-between-virtual-function-and-pure-virtual-function.html#comments)

The virtual function and pure virtual function both are the concepts of run time polymorphism. Polymorphism is supported by both the languages C++ and Java. In Java, the term “overriding” is used instead of ‘virtual function’, as the virtual function is the term of C++. The main difference between ‘virtual function’ and ‘pure virtual function’ is that ‘virtual function’ has its definition in the base class and also the inheriting derived classes redefine it. The pure virtual function has no definition in the base class, and all the inheriting derived classes has to redefine it.

Content: Virtual Function Vs Pure Virtual Function

1. [Comparison Chart](https://techdifferences.com/difference-between-virtual-function-and-pure-virtual-function.html#ComparisonChart)
2. [Definition](https://techdifferences.com/difference-between-virtual-function-and-pure-virtual-function.html#Definition)
3. [Key Differences](https://techdifferences.com/difference-between-virtual-function-and-pure-virtual-function.html#KeyDifferences)
4. [Conclusion](https://techdifferences.com/difference-between-virtual-function-and-pure-virtual-function.html#Conclusion)

Comparison Chart

| **BASIS FOR COMPARISON** | **VIRTUAL FUNCTION** | **PURE VIRTUAL FUNCTION** |
| --- | --- | --- |
| Basic | 'Virtual function' has their definition in the base class. | 'Pure Virtual Function' has no definition in the base class. |
| Declaration | virtual funct\_name(parameter\_list) {. . . . .}; | virtual funct\_name(parameter\_list)=0; |
| Derived class | All derived classes may or may not override the virtual function of the base class. | All derived classes must override the virtual function of the base class. |
| Effect | Virtual functions are hierarchical in nature; it does not affect compilation if any derived classes do not override the virtual function of the base class. | If all derived classes fail to override the virtual function of the base class, the compilation error will occur. |
| Abstract class | No concept. | If a class contains at least one pure virtual function, then it is declared abstract. |

Definition of Virtual Function

The virtual function is the member function of the base class, and it is redefined by the derived classes which inherit’s the base class. It is not necessary that all the inheriting derived classes must redefine the virtual function, it is only redefined by those derived classes which may require its functioning.  A virtual function is created by the declaring the function in the base class preceded with the keyword ‘virtual’.  
Declaration :

1. class base{
2. public :
3. virtual type **funt\_name**( parameter-list)
4. {
5. .
6. .
7. .
8. }
9. };

The inheriting derived classes can redefine the virtual function without any ‘virtual’ keyword. Derived classes redefine the virtual function to accomplish its task. As the virtual function is redefined in derived classes, we have multiple forms of the same function, now, which version of the function is being called, depends on the what kind of object is referred to invoke that function.

In multilevel inheritance, where a derived class that has inherited the virtual function from its base class, when itself is used as base class for another derived class, the virtual function still can be overridden. So, when a virtual function is inherited its virtual nature is also inherited.

Virtual functions are also hierarchical in nature .i.e. if a derived class do not override/redefine the virtual function inherited from the base class and when derived class’s object invokes that virtual function, then the virtual function defined by the base class is invoked.

Definition of Pure Virtual Function

As seen above if derived class do not override the virtual function then the virtual function defined by the base class is used. But, what if the base class itself doesn’t define the virtual function. Many times, the base class has no definition for the virtual function, or sometimes you want that all derived classes must override the virtual function.

To handle these above two situations, C++ supports the concept of “Pure Virtual Function”. A ‘pure virtual function’ is declared in base class but do not have its definition in the base class. The pure virtual function is declared as follows.

1. virtual type **funct\_name**(parameter\_list)=0;

Whenever a virtual function in the base class is made “pure”, then each derived class must mandatorily override the pure virtual function of the base class. If derived class fails to override the pure virtual function of the base class, it will result in compilation error. The class that contains at least one pure function is called ‘abstract class’.

No objects of abstract classes can be created, but you can create references and pointers to abstract classes. Members of the abstract classes can be accessed through the object of the derived classes which inherits the abstract base class. A class which you want to declare abstract, use the keyword ‘abstract’ in front of the ‘class’ keyword.

1. //for example
2. abstract class class-name{
3. .
4. .
5. virtual type **funct\_name**(parameter\_list)=0;
6. .
7. .
8. };

 Key Differences Between Virtual Function and Pure Virtual Function

1. Virtual functions are definitely defined in the base class and redefined (override) in the derived class. On the other hand, pure virtual function the base class is particularly not defined in the base class
2. Derived class if required redefine(override) the virtual function whereas, in case of pure virtual function derived class has to definitely redefine the pure virtual function.
3. If the derived class fails to redefine (override) the virtual function it can use the virtual function of the base class. On the other hand, if derived class fails to override pure virtual function then compilation error occurs.
4. Base class containing the virtual function can be instantiated i.e. its object can be created but, the base class containing pure virtual function i.e. abstract class can not be instantiated as an abstract class is not fully defined.

Note:

The prototype of the ‘virtual function’ and ‘pure virtual function’ remains same throughout the program.

Conclusion:

The ‘virtual functions’ and ‘pure virtual function’ both have their importance, as in ‘virtual functions’, all derived class do not need to redefine the virtual function and where we want that all derived class should redefine the virtual function, pure virtual function best applies there.