

# Abstract Class

---

- Abstract classes can't be instantiated, i.e., we cannot create an object of this class.
- However, we can derive a class from it and instantiate the object of the derived class.
- An Abstract class has at least one pure virtual function.

## Properties of the abstract classes:

- It can have normal functions and variables along with pure virtual functions.
- Prominently used for upcasting (converting a derived-class reference or pointer to a base-class).
- In other words, upcasting allows us to treat a derived type as a base type), so its derived classes can use its interface.
- If an abstract class has a derived class, they must implement all pure virtual functions, or they will become abstract.
- If we do not override the pure virtual function in the derived class, then the derived class also becomes an abstract class.
- We cannot create objects of an abstract class.
- However, we can derive classes from them and use their data members and member functions (except pure virtual functions).

```
#include<iostream>
using namespace std;
class Base {
    public:
        virtual void s() = 0; // Pure Virtual Function
};

class Derived: public Base {
    public:
        void s() {
            cout << "Virtual Function in Derived_class";
        }
};

int main() {
    Base *b;
    Derived d_obj;
    b = &d_obj;
    b->s();
}
```

Output

Virtual Function in Derived\_class



