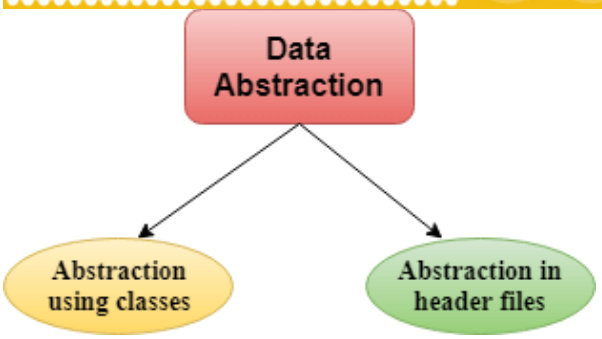
Data Abstraction in C++

- Data Abstraction is a process of providing only the essential details to the outside world and hiding the internal details, i.e., representing only the essential details in the program.
- Data Abstraction is a programming technique that depends on the seperation of the interface and implementation details of the program.
- Let's take a real life example of AC, which can be turned ON or OFF, change the temperature, change the mode, and other external components such as fan, swing. But, we don't know the internal details of the AC, i.e., how it works internally. Thus, we can say that AC seperates the implementation details from the external interface.
- C++ provides a great level of abstraction. For example, pow() function is used to calculate the power of a number without knowing the algorithm the function follows.

In C++ program if we implement class with private and public members then it is an example of data abstraction.

Data Abstraction can be achieved in two ways:

- Abstraction using classes
- Abstraction in header files.



Abstraction using classes: An abstraction can be achieved using classes. A class is used to group all the data members and member functions into a single unit by using the access specifiers. A class has the responsibility to determine which data member is to be visible outside and which is not.

Abstraction in header files: An another type of abstraction is header file. For example, pow() function available is used to calculate the power of a number without actually knowing which algorithm function uses to calculate the power. Thus, we can say that header files hides all the implementation details from the user.

Access Specifiers Implement Abstraction:

- **Public specifier:** When the members are declared as public, members can be accessed anywhere from the program.
- **Private specifier:** When the members are declared as private, members can only be accessed only by the member functions of the class.

Let's see a simple example of abstraction in header files.

// program to calculate the power of a number.

```
#include <iostream>
#include<math.h>
using namespace std;
int main()
{
 int n = 4;
  int power = 3;
  int result = pow(n,power);        // pow(n,power) is the  power function
  std::cout << "Cube of n is : " <<result<< std::endl;
  return 0;
}
```

Output:

```
Cube of n is : 64
```

In the above example, pow() function is used to calculate 4 raised to the power 3. The pow() function is present in the math.h header file in which all the implementation details of the pow() function is hidden.

Let's see a simple example of data abstraction using classes.

```
#include <iostream>
using namespace std;
 class Sum
{
private: int x, y, z; // private variables
public:
```

```
void add()
{
cout<<"Enter two numbers: ";
cin>>x>>y;
z= x+y;
cout<<"Sum of two number is: "<<z<<endl;
}
};
int main()
{
Sum sm;
sm.add();
return 0;
}
```

Output:

```
Enter two numbers:
3
6
Sum of two number is: 9
```

In the above example, abstraction is achieved using classes. A class 'Sum' contains the private members x, y and z are only accessible by the member functions of the class.

Advantages Of Abstraction:

- Implementation details of the class are protected from the inadvertent user level errors.
- A programmer does not need to write the low level code.
- Data Abstraction avoids the code duplication, i.e., programmer does not have to undergo the same tasks every time to perform the similar operation.
- The main aim of the data abstraction is to reuse the code and the proper partitioning of the code across the classes.
- Internal implementation can be changed without affecting the user level code.

← Prev Next →

AD

Youtube For Videos Join Our Youtube Channel: Join Now

Feedback

- Send your Feedback to feedback@javatpoint.com

Help Others, Please Share

Learn Latest Tutorials

Splunk tutorial
Splunk

SPSS tutorial
SPSS

Swagger tutorial
Swagger

T-SQL tutorial
Transact-SQL

Tumblr tutorial
Tumblr

React tutorial
ReactJS

Regex tutorial
Regex

Reinforcement learning tutorial
Reinforcement Learning

R Programming tutorial

RxJS tutorial
RxJS

React Native tutorial

Python Design Patterns

R Programming

React Native

Python Design Patterns

Python Pillow tutorial

Python Pillow

Python Turtle tutorial

Python Turtle

Keras tutorial

Keras

Preparation

Aptitude

Aptitude

Logical Reasoning

Reasoning

Verbal Ability

Verbal Ability

Interview Questions

Interview Questions

Company Interview Questions

Company Questions

Trending Technologies

Artificial Intelligence

Artificial Intelligence

AWS Tutorial

AWS

Selenium tutorial

Selenium

Cloud Computing

Cloud Computing

Hadoop tutorial

Hadoop

ReactJS Tutorial

ReactJS

Data Science Tutorial

Data Science

Angular 7 Tutorial

Angular 7

Blockchain Tutorial

Blockchain

Git Tutorial

Git

Machine Learning Tutorial

Machine Learning

DevOps Tutorial

DevOps

B.Tech / MCA

DBMS tutorial
DBMS

Data Structures tutorial
Data Structures

DAA tutorial
DAA

Operating System
Operating System

Computer Network tutorial
Computer Network

Compiler Design tutorial
Compiler Design

Computer Organization and Architecture
Computer Organization

Discrete Mathematics Tutorial
Discrete Mathematics

Ethical Hacking
Ethical Hacking

Computer Graphics Tutorial
Computer Graphics

Software Engineering
Software Engineering

html tutorial
Web Technology

Cyber Security tutorial
Cyber Security

Automata Tutorial
Automata

C Language tutorial
C Programming

C++ tutorial
C++

Java tutorial
Java

.Net Framework tutorial
.Net

Python tutorial
Python

List of Programs
Programs

Control Systems tutorial
Control System

Data Mining Tutorial
Data Mining

Data Warehouse Tutorial
Data Warehouse

AD
eCruit
a new way to hire talent

Interfaces in C++ (Abstract Classes)

Abstract classes are the way to achieve abstraction in C++. Abstraction in C++ is the process to hide the internal details and showing functionality only. Abstraction can be achieved by two ways:

1. **Abstract class**
2. **Interface**

Abstract class and interface both can have abstract methods which are necessary for abstraction.

C++ Abstract class

In C++ class is made abstract by declaring at least one of its functions as <>strong>pure virtual function. A pure virtual function is specified by placing "= 0" in its declaration. Its implementation must be provided by derived classes.

Let's see an example of abstract class in C++ which has one abstract method draw(). Its implementation is provided by derived classes: Rectangle and Circle. Both classes have different implementation.

```
#include <iostream>
using namespace std;
 class Shape
{
   public:
   virtual void draw()=0;
};
 class Rectangle : Shape
{
   public:
    void draw()
   {
       cout < <"drawing rectangle..." < <endl;
   }
};
class Circle : Shape
{
   public:
```

```
    void draw()
    {
        cout <<"drawing circle..." < <endl;
    }
};
int main( ) {
    Rectangle rec;
    Circle cir;
    rec.draw();
    cir.draw();
   return 0;
}
```

Output:

```
drawing rectangle...
drawing circle...
```

← Prev Next →

AD

For Videos Join Our Youtube Channel: Join Now