**Array of Objects**

When a class is defined, only the specification for the object is defined; no memory or storage is allocated. To use the data and access functions defined in the class, you need to create objects.

**Syntax:**

ClassName ObjectName[number of objects];

The Array of Objects stores *objects*. An array of a class type is also known as an array of objects.

**Example#1:**
Storing more than one Employee data. Let’s assume there is an array of objects for storing employee data emp[50].



Below is the C++ program for storing data of one Employee:

**C++**

// C++ program to implement

// the above approach

#include<iostream>

**using** **namespace** std;

**class** Employee

{

  **int** id;

  **char** name[30];

  **public**:

  **void** getdata();//Declaration of function

  **void** putdata();//Declaration of function

};

**void** Employee::getdata(){//Defining of function

  cout<<"Enter Id : ";

  cin>>id;

  cout<<"Enter Name : ";

  cin>>name;

}

**void** Employee::putdata(){//Defining of function

  cout<<id<<" ";

  cout<<name<<" ";

  cout<<endl;

}

**int** main(){

  Employee emp; //One member

  emp.getdata();//Accessing the function

  emp.putdata();//Accessing the function

  **return** 0;

}

Let’s understand the above example –

* In the above example, a class named Employee with id and name is being considered.
* The two functions are declared-
	+ **getdata():** Taking user input for id and name.
	+ **putdata():** Showing the data on the console screen.

This program can take the data of only one Employee. What if there is a requirement to add data of more than one Employee. Here comes the answer Array of Objects. An array of objects can be used if there is a need to store data of more than one employee. Below is the C++ program to implement the above approach-

**C++**

|  |
| --- |
| // C++ program to implement // the above approach#include<iostream>**using** **namespace** std; **class** Employee{  **int** id;  **char** name[30];  **public**:     // Declaration of function  **void** getdata();     // Declaration of function  **void** putdata();}; // Defining the function outside // the class**void** Employee::getdata(){   cout << "Enter Id : ";  cin >> id;  cout << "Enter Name : ";  cin >> name;} // Defining the function outside // the class**void** Employee::putdata(){  cout << id << " ";  cout << name << " ";  cout << endl;} // Driver code**int** main(){  // This is an array of objects having  // maximum limit of 30 Employees  Employee emp[30];   **int** n, i;  cout << "Enter Number of Employees - ";  cin >> n;     // Accessing the function  **for**(i = 0; i < n; i++)     emp[i].getdata();     cout << "Employee Data - " << endl;     // Accessing the function  **for**(i = 0; i < n; i++)     emp[i].putdata();} |

**Output:**



**Explanation:**
In this example, more than one Employee’s details with an Employee id and name can be stored.

* Employee emp[30] – This is an array of objects having a maximum limit of 30 Employees.
* Two for loops are being used-
	+ First one to take the input from user by calling emp[i].getdata() function.
	+ Second one to print the data of Employee by calling the function emp[i].putdata() function.

**Example#2:**

**C++**

|  |
| --- |
| // C++ program to implement// the above approach#include<iostream>**using** **namespace** std;**class** item{  **char** name[30];  **int** price;  **public**:  **void** getitem();  **void** printitem();};  // Function to get item details**void** item::getitem(){  cout << "Item Name = ";  cin >> name;  cout << "Price = ";  cin >> price;    } // Function to print item// details**void** item ::printitem(){  cout << "Name : " << name <<           "\n";  cout << "Price : " << price <<           "\n";} **const** **int** size = 3; // Driver code**int** main(){  item t[size];  **for**(**int** i = 0; i < size; i++)  {    cout << "Item  : " <<             (i + 1) << "\n";    t[i].getitem();  }     **for**(**int** i = 0; i < size; i++)  {    cout << "Item Details : " <<              (i + 1) << "\n";    t[i].printitem();  }} |

**Output:**



**Advantages of Array of Objects:**

1. The array of objects represent storing multiple objects in a single name.
2. In an array of objects, the data can be accessed randomly by using the index number.
3. Reduce the time and memory by storing the data in a single variable.