1. Initialize the size (say **n**) of vectors.
2. While adding an element, go through the following steps:
   1. Get the value of x=”[value] MOD **n**“.
   2. Push back this element’s value in the v[x].
3. For Deletion, we follow these steps:
   1. Get the value of x=”[value] MOD **n**“.
   2. Scan for the element to be deleted in v[x].
   3. If found, remove the element using [erase()](https://www.geeksforgeeks.org/vector-erase-and-clear-in-cpp/) method.
   4. If element to be deleted is not found, print “Element not found!”
4. Print the hash.

**Implementation**

| // C++ Program to implement Separate  // Chaining in C++ STL without  // the use of pointers    #include <bits/stdc++.h>  using namespace std;    // Container for our data-set  class SeperateHash {        // Data members are kept private      // since they are accessed using methods  private:      int n;      vector<vector<int> > v;        // Methods are kept public  public:      // Initialising constructors as the      // minimum required memory will be      // allocated after which compiler      // will not report flag error      SeperateHash(int n)      {          // Constructor to initialize          // the vector of vectors          v = vector<vector<int> >(n);            // Here, we will allocate          // memory of the unnamed\_memory          // to the object. This snippet          // of code won't work for now.          // Program will work whenever          // constructor gets initialized          this->n = n;      }        int getHashIndex(int x)      {          return x % n;      }        void add(int x)      {          // Adding the element according          // to hash index          v[getHashIndex(x)].push\_back(x);      }        void del(int x)      {          int i = getHashIndex(x);            // Scan for the element to be removed          for (int j = 0; j < v[i].size(); j++) {                // Erase if present otherwise              // print no element found!              if (v[i][j] == x) {                  v[i].erase(v[i].begin() + j);                  cout << x << " deleted!" << endl;                  return;              }          }            cout << "No Element Found!" << endl;      }      void displayHash()      {          // Display the contents          for (int i = 0; i < v.size(); i++) {              cout << i;              for (int j = 0; j < v[i].size(); j++)                  cout << " -> " << v[i][j];              cout << endl;          }      }  };    // Driver Code  int main()  {      // Array to be used      int arr[] = { 12, 3, 23, 4, 11,                    32, 26, 33, 17, 19 };        // Sending the size      // for separate chaining      SeperateHash obj(10);        // Inserting elements in      // the container accordingly      for (int i = 0; i < 10; i++)          obj.add(arr[i]);        // Display the initial hash      cout << "Initial Hash:\n";      obj.displayHash();        // Removing the element      // from the container      cout << "\nRemoving 23 from Hash: ";      obj.del(23);      cout << endl;        // Display the final hash      cout << "Final Hash:\n";      obj.displayHash();      return 0;  } |
| --- |