

|  |
| --- |
| #include<bits/stdc++.h>  using namespace std;    class Hash  {      int BUCKET;    // No. of buckets        // Pointer to an array containing buckets      list<int> \*table;  public:      Hash(int V);  // Constructor        // inserts a key into hash table      void insertItem(int x);        // deletes a key from hash table      void deleteItem(int key);        // hash function to map values to key      int hashFunction(int x) {          return (x % BUCKET);      }        void displayHash();  };    Hash::Hash(int b)  {      this->BUCKET = b;      table = new list<int>[BUCKET];  }    void Hash::insertItem(int key)  {      int index = hashFunction(key);      table[index].push\_back(key);  }    void Hash::deleteItem(int key)  {    // get the hash index of key    int index = hashFunction(key);      // find the key in (inex)th list    list <int> :: iterator i;    for (i = table[index].begin();             i != table[index].end(); i++) {      if (\*i == key)        break;    }      // if key is found in hash table, remove it    if (i != table[index].end())      table[index].erase(i);  }    // function to display hash table  void Hash::displayHash() {    for (int i = 0; i < BUCKET; i++) {      cout << i;      for (auto x : table[i])        cout << " --> " << x;      cout << endl;    }  }    // Driver program  int main()  {    // array that contains keys to be mapped    int a[] = {15, 11, 27, 8, 12};    int n = sizeof(a)/sizeof(a[0]);      // insert the keys into the hash table    Hash h(7);   // 7 is count of buckets in                 // hash table    for (int i = 0; i < n; i++)      h.insertItem(a[i]);      // delete 12 from hash table    h.deleteItem(12);      // display the Hash table    h.displayHash();      return 0;  } |

**Output:**

0

1 --> 15 --> 8

2

3

4 --> 11

5

6 --> 27