```
#include <iostream>
#include <time.h>
using namespace std;

struct NODE
{
   int data;
   NODE *pNext;
};
struct LIST
{
   NODE *Head;
   NODE *Tail;
};
NODE *CreateNode (int x)
{
   NODE *p = new NODE;
    if (p==NULL)
      return NULL;
    p->data=x;
    p->pNext=NULL;
    return p;
}
void CreateList (LIST &l)
{
   l.Head=NULL;
   l.Tail=NULL;
}
void AddTail (LIST &l,NODE *p)
```

```
{
   if (l.Head==NULL)
   {
      l.Head=l.Tail=p;
   }
   else
   {
      l.Tail->pNext=p;
      l.Tail=p;
   }
}
void Print_BK (LIST &l)
{
   for (NODE *p=l.Head;p!=NULL;p=p->pNext)
      cout << p->data << "   ";
   cout << endl;
}
//================================HASH TABLE=========================//
#define M 7 // =>SIZE của bảng băm
//Buoc 1: Tạo cấu trúc bảng băm
struct HASHTABLE
{
   LIST bucket [M];
};
//Bước 2: Khởi động bảng băm / các "bucket"
void InitBuckets(HASHTABLE &h)
{
   for (int i=0;i<M;i++)
   {
```

```
        CreateList(h.bucket[i]);
    }
}
//Giả sử chúng ta chọn hàm  băm dạng %: f(key)=key % M.
//h(k) =k mod m
int hashfunc(int data)
{
    return data % M;
}
void Insert(HASHTABLE &h,int data)
{
    //B1:Băm để xác định vị trí
    int b=hashfunc(data);
    //b2:Thêm vào
      NODE *p =CreateNode (data);
      AddTail(h.bucket[b],p);
    //Hoặc :
     //AddTail(h.bucket[b],CreateNode(data));
}
void show_HashTable (HASHTABLE h)
{
    for (int i=0;i<M;i++)
    {
        cout << "[BUCKET #" << i << "]:\t";
        //Print_BK(h.bucket[i]);
        LIST l=h.bucket[i];
        for (NODE *i=l.Head;i!=NULL;i=i->pNext)
        {
            cout << i->data << "\t";
```

```
        }
        cout << endl;
    }
}
void input_array(HASHTABLE &h,int a[],int n)
{
    for (int i=0;i<n;i++)
    {
        Insert(h, a[i]);
    }
}

void NhapThuCong(HASHTABLE &h)
{
    int x;
    do
    {
        int chon = 0;
        cout << "Nhap phan tu de gan vao bang bam: ";
        cin >> x;
        if(x != 0)
        {
            Insert (h,x);
        }
        else
        {
            cout << "Nhap phim 1 de nhap 0 vao bang bam, nhan phim 0 de ket thuc nhap: ";
            cin >> chon;
            if (chon != 0)
```

```
        {
            Insert(h,0);
            NhapThuCong(h);
        }
        else
            return;
      }
   }while(x != 0);
}

void auto_input_array(HASHTABLE &h)
{
   cout << "So phan tu duoc tao ngau nhien: ";
     srand(time(NULL));
     int n = 45 + rand() % 51;
   cout << n << endl;
    for(int i = 0; i < n; i++)
   {
      double x = 856 + rand() % 133;
       Insert (h,x);
   }
}
NODE *Search_X(HASHTABLE h,int x)
{
   //B1:Băm để tìm bucket
   int iBucket = hashfunc(x);
   LIST l = h.bucket[iBucket];
    //B2:Xử lý
    for (NODE *p=l.Head;p!=NULL;p=p->pNext)
```

```
    {
        if (x == p->data)
            return p;
    }
    return NULL;
}
/*
Câu 5: Xóa giá trị trong bang băm
*/
void delete_Node(HASHTABLE &h, int x)
{
    int i = hashfunc(x);
    if(h.bucket[i].Head == NULL)
        {
            return; // Gía trị ko tồn tai
        }

    // giá trị ở vị trí đầu tiên
    if(h.bucket[i].Head->data == x)
    {
        h.bucket[i].Head = h.bucket[i].Head->pNext;
        return;
    }

    NODE *q = h.bucket[i].Head;
    for(NODE *p = h.bucket[i].Head; p!=NULL; p=p->pNext)
    {
        if(p->data == x)
        {
```

```
            q->pNext = p->pNext;
            delete p;
            return;
        }
        q = p;
    }
}
int Tong_Le(HASHTABLE h)
{
    int s = 0;
    for (int i = 0; i < M; i++)
    {
        LIST l = h.bucket[i];
        for (NODE *p = l.Head; p != NULL; p = p->pNext)
        {
            if(p->data % 2 == 1)
                s+=p->data;
        }
    }
    return s;
}
bool IsEmpty (HASHTABLE h)
{
    for (int i = 0; i < M; i++)
    {
        LIST l = h.bucket[i];
        for (NODE *p = l.Head; p != NULL; p = p->pNext)
        {
            if(p->data != 0)
```

```
            return false;
        }
    }
    return true;
}

void KiemTraChanLe(HASHTABLE h)
{
    int chan = 0;
    int le = 0;
    for (int i = 0; i < M; i++)
    {
        LIST l = h.bucket[i];
        for (NODE *p = l.Head; p != NULL; p = p->pNext)
        {
            if(p->data % 2 == 1)
                le++;
            else
                chan++;
        }
    }
    cout << "Bang bam co " << chan << " phan tu chan, va co " << le << " phan tu le." <<endl;
}

int main()
{
    HASHTABLE h;
    InitBuckets(h);
```

```
//    Insert (h, 50);
//    Insert (h, 700);
//    Insert (h, 76);
//    Insert (h, 85);
//    Insert (h, 92);
//    Insert (h, 73);
//    Insert (h, 101);
//    cout << "h1: " <<endl;
//    show_HashTable(h);
//
//    cout << "==============================" << endl;


  while (true)
  {
     cout << endl;
     cout <<"====================MENU==================="<<endl;
     cout <<">>\t1.Nhap tu dong cho bang bam."<<endl;
     cout <<">>\t2.Tao du lieu cho bang bam tu mang 1 chieu"<<endl;
     cout <<">>\t3.Tao du lieu thu cong nhap tu ban phim." << endl;
     cout <<">>\t4.Print HashTable." << endl;
     cout <<">>\t5.Xoa gia tri trong bang bam."<<endl;
     cout <<">>\t6.Tim 1 gia tri trong bang bam."<<endl;
     cout <<">>\t7.Tong cac gia tri le."<<endl;
     cout <<">>\t8.Check HashTable empty."<<endl;
     cout <<">>\t9.(Tuy chon).Kiem tra chan le."<<endl;
     cout << endl;
     cout <<"============================================="<<endl;
```

```
  cout << "Xin moi lua chon : ";

  int chon;
  cin >> chon;

switch (chon)
{
  case 0:
     {
     cout << "Chuong trinh ket thuc."<<endl;
     return 0;
     }
  case 1:
   {
      cout << "Cac gia tri duoc nhap tu dong:"<<endl;
       auto_input_array(h);
       //show_HashTable(h);
       break;
   }
  case 2:
   {
      cout << "Tao du lieu cho bang bam tu 1D."<<endl;
      int arr[]={50,700,76,85,92,73,101};
       input_array(h,arr,7);
      cout << "Da tao thanh cong."<<endl;
       //show_HashTable(h);
       break;
   }
   case 3:
```

```
{
    cout << "Nhap thu cong tu ban phim:\n";
    NhapThuCong(h);
    break;
}
case 4:
{
   cout << "Print HashTable." << endl;
    show_HashTable(h);
    break;
}
case 5:
{
    cout << "Nhap gia tri muon xoa trong bang bam: ";
    int x;
    cin >> x;
    delete_Node(h,x);
    cout << "Gia tri: " << x <<  " da xoa khoi bang bam." << endl;
    cout << "=======HashTable duoc cap nhat lai========== " <<endl;
    show_HashTable(h);
    break;
}
case 6:
{
    cout << "Tim 1 gia tri :"<<endl;
    int x;
    cin >> x;
    NODE *p=Search_X(h,x);
    if (p == NULL)
```

```
            cout << "Khong tim thay!";
        else
            cout << "Tim thay bucket!" << endl;
            cout << "Bucket "<< x << " co dia chi la: " << p << endl;
        break;
      }
    case 7:
      {
        cout << "Tong gia tri le la " << Tong_Le(h) << endl;
        break;
      }
    case 8:
      {
          cout <<"Check HashTable empty"<<endl;
        if (IsEmpty(h))
          cout << "HashTable is Empty."<<endl;
        else
           cout << "Not empty."<< endl;
       break;
      }
    case 9:
        KiemTraChanLe(h);
        break;
   }

 }
   return 0;
}
```