**DSA Lab 11**

**Name:** Hafsa Salman

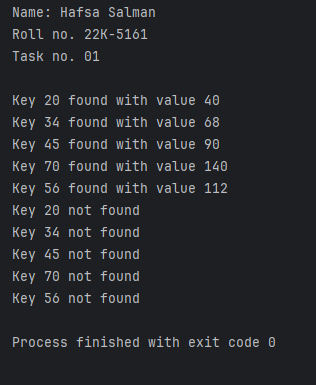
**Roll no.** 22K-5161

**Task no. 01**

Code:

//Hafsa Salman  
//22K-5161  
//Task no. 01  
  
public class Task\_01  
{  
 public static void main(String[] args)  
 {  
 System.*out*.println("Name: Hafsa Salman");  
 System.*out*.println("Roll no. 22K-5161");  
 System.*out*.println("Task no. 01");  
 System.*out*.println();  
  
 int[] keys = {20, 34, 45, 70, 56};  
  
 HashTable hashTable = new HashTable(10);  
  
 // inserting elements into the table  
 for (int key : keys)  
 {  
 hashTable.insert(key, key \* 2);  
 }  
  
 // searching and displaying elements  
 for (int key : keys)  
 {  
 Node result = hashTable.search(key);  
 if (result != null)  
 {  
 System.*out*.println("Key " + key + " found with value " + result.value);  
 }  
  
 else  
 {  
 System.*out*.println("Key " + key + " not found");  
 }  
 }  
  
 // deleting elements  
 for (int key : keys)  
 {  
 hashTable.delete(key);  
 }  
  
 // searching after deletion  
 for (int key : keys)  
 {  
 Node result = hashTable.search(key);  
 if (result != null)  
 {  
 System.*out*.println("Key " + key + " found with value " + result.value);  
 }  
  
 else  
 {  
 System.*out*.println("Key " + key + " not found");  
 }  
 }  
 }  
}  
  
class Node  
{  
 int key;  
 int value;  
 Node next;  
  
 public Node(int key, int value)  
 {  
 this.key = key;  
 this.value = value;  
 this.next = null;  
 }  
}  
  
class LinkedList  
{  
 Node head;  
  
 public void insert(int key, int value)  
 {  
 Node newNode = new Node(key, value);  
  
 newNode.next = head;  
 head = newNode;  
 }  
  
 public Node search (int key)  
 {  
 Node current = head;  
  
 while (current != null)  
 {  
 if (current.key == key)  
 {  
 return current;  
 }  
  
 current = current.next;  
 }  
  
 return null;  
 }  
  
 public void delete(int key)  
 {  
 Node current = head;  
 Node prev = null;  
  
 while (current != null && current.key != key)  
 {  
 prev = current;  
 current = current.next;  
 }  
  
 if (current != null)  
 {  
 if (prev != null)  
 {  
 prev.next = current.next;  
 }  
  
 else  
 {  
 head = current.next;  
 }  
 }  
 }  
}  
  
class HashTable  
{  
 private LinkedList[] table;  
 private int size;  
  
 public HashTable(int size)  
 {  
 this.size = size;  
 table = new LinkedList[size];  
  
 for (int i = 0; i < size; i++)  
 {  
 table[i] = new LinkedList();  
 }  
 }  
  
 private int hash (int key)  
 {  
 return key % size;  
 }  
  
 public void insert(int key, int value)  
 {  
 int index = hash(key);  
 table[index].insert(key, value);  
 }  
  
 public Node search(int key)  
 {  
 int index = hash(key);  
  
 return table[index].search(key);  
 }  
  
 public void delete(int key)  
 {  
 int index = hash(key);  
 table[index].delete(key);  
 }  
}

Output:

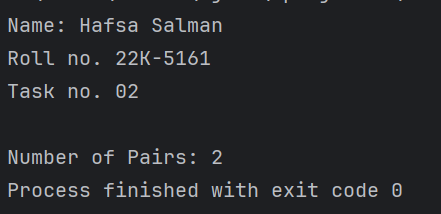


**Task no. 02**

Code:

//Hafsa Salman  
//22K-5161  
//Task no. 02  
  
public class Task\_02  
{  
 public static void main(String[] args)  
 {  
 System.*out*.println("Name: Hafsa Salman");  
 System.*out*.println("Roll no. 22K-5161");  
 System.*out*.println("Task no. 02");  
 System.*out*.println();  
  
 int[] arr = {1, 5, 7, 1};  
  
 int k;  
  
 k = 6;  
  
 HashLL hash = new HashLL(10);  
  
 for (int i=0;i<arr.length;i++)  
 {  
 hash.insert(arr[i]);  
 }  
 System.*out*.print("Number of Pairs: ");  
 System.*out*.print(hash.getNumOfPairs(arr,k));  
 }  
}  
  
class HashNode  
{  
 int key, value;  
 HashNode next;  
  
 public HashNode(int value)  
 {  
 this.value = value;  
 this.next = null;  
 }  
}  
  
class HashLL  
{  
 int size = 10;  
 HashNode[] ht;  
  
 public HashLL(int size)  
 {  
 this.size = size;  
 this.ht = new HashNode[size];  
 }  
  
 public void insert(int k)  
 {  
 int v = hashFunc(k);  
  
 HashNode n = new HashNode(k);  
  
 if(ht[v]==null)  
 {  
 ht[v] = n;  
 }  
  
 else  
 {  
 HashNode curr = ht[v];  
  
 while(curr.next!=null)  
 {  
 curr=curr.next;  
 }  
 curr.next = n;  
 }  
 }  
  
 public int hashFunc(int k)  
 {  
 return Math.*abs*(k) % size;  
 }  
  
 public boolean searchKey (int v)  
 {  
 boolean found;  
  
 found = false;  
  
 for(int i=0;i<ht.length;i++)  
 {  
 if(ht[i] == null)  
 {  
 continue;  
 }  
  
 else  
 {  
 HashNode curr = ht[i];  
  
 while(curr != null)  
 {  
 if(curr.value == v)  
 {  
 found=true;  
 }  
  
 curr=curr.next;  
 }  
 }  
 }  
  
 return found;  
 }  
  
 public int getNumOfPairs (int[] arr, int k)  
 {  
 int count;  
  
 count = 0;  
  
 for (int i = 1; i < arr.length; i++)  
 {  
 if (k - arr[i] > 0)  
 {  
 if (searchKey(k - arr[i]))  
 {  
 count++;  
 }  
 }  
 }  
  
 return count;  
 }  
}

Output:

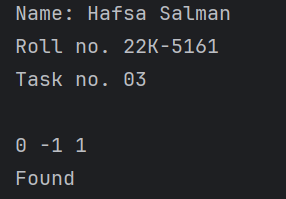


**Task no. 03**

Code:

//Hafsa Salman  
//22K-5161  
//Task no. 03  
  
public class Task\_03  
{  
 public static void main(String[] args)  
 {  
 System.*out*.println("Name: Hafsa Salman");  
 System.*out*.println("Roll no. 22K-5161");  
 System.*out*.println("Task no. 03");  
 System.*out*.println();  
  
 Hashing beep = new Hashing(10);  
  
 int[] arr={0, -1, 2, -3, 1};  
  
 for(int i=0;i<arr.length;i++)  
 {  
 beep.insert(arr[i]);  
 }  
  
 if(beep.triplets(arr,0)==1)  
 {  
 System.*out*.println("Found");  
 }  
  
 else  
 {  
 System.*out*.println("Not found");  
 }  
 }  
}  
  
class HashNodee  
{  
 int key, value;  
 HashNode next;  
  
 public HashNodee (int value)  
 {  
 this.value = value;  
 this.next = null;  
 }  
}  
  
class Hashing  
{  
 int size = 10;  
 HashNode[] ht;  
  
 public Hashing (int size)  
 {  
 this.size = size;  
 this.ht = new HashNode[size];  
 }  
  
 public void insert(int k)  
 {  
 int v = hashFunc(k);  
  
 HashNode n = new HashNode(k);  
  
 if(ht[v]==null)  
 {  
 ht[v] = n;  
 }  
  
 else  
 {  
 HashNode curr = ht[v];  
  
 while(curr.next!=null)  
 {  
 curr=curr.next;  
 }  
 curr.next = n;  
 }  
 }  
  
 public int hashFunc(int k)  
 {  
 return Math.*abs*(k) % size;  
 }  
  
 public boolean searchKey (int v)  
 {  
 boolean found;  
  
 found = false;  
  
 for(int i=0;i<ht.length;i++)  
 {  
 if(ht[i] == null)  
 {  
 continue;  
 }  
  
 else  
 {  
 HashNode curr = ht[i];  
  
 while(curr != null)  
 {  
 if(curr.value == v)  
 {  
 found=true;  
 }  
  
 curr=curr.next;  
 }  
 }  
 }  
  
 return found;  
 }  
  
 public int triplets (int[] arr, int sum)  
 {  
 for (int i=0; i<arr.length-2; i++)  
 {  
 int currSum = sum - arr[i];  
  
 for (int j=i+1; j<arr.length; j++)  
 {  
 int third = currSum-arr[j];  
  
 if (searchKey(third))  
 {  
 System.*out*.println(arr[i] + " " + arr[j] + " " + third);  
  
 return 1;  
 }  
 insert(arr[j]);  
 }  
 }  
 return 0;  
 }  
}

Output:



**Task no. 04**

Code:

//Hafsa Salman  
//22K-5161  
//Task no. 04  
  
public class Task\_04  
{  
 public static void matchNutAndBolt(char[] nuts,char[] bolts)  
 {  
 HashLLChar n = new HashLLChar(5);  
 HashLLChar b= new HashLLChar(5);  
  
 if (nuts.length == bolts.length)  
 {  
 for (int i = 0; i < nuts.length; i++)  
 {  
 n.insert(nuts[i]);  
 }  
  
 for (int i = 0; i < bolts.length; i++)  
 {  
 if (n.searchKey(bolts[i]))  
 {  
 nuts[i] = bolts[i];  
 }  
 }  
  
 System.*out*.println("Nuts:");  
  
 for (int i=0; i < nuts.length; i++)  
 {  
 System.*out*.print(nuts[i]+ " ");  
 }  
  
 System.*out*.println();  
  
 System.*out*.println("\nBolts:");  
 for (int i= 0; i < bolts.length; i++)  
 {  
 System.*out*.print(bolts[i]+ " ");  
 }  
  
 System.*out*.println();  
 }  
 }  
  
 public static void main(String[] args)  
 {  
 System.*out*.println("Name: Hafsa Salman");  
 System.*out*.println("Roll no. 22K-5161");  
 System.*out*.println("Task no. 04");  
 System.*out*.println();  
  
 char[] nuts={'@','%', '$', '#', '^'};  
 char[] bolts={'%','@', '#', '$', '^'};  
  
 *matchNutAndBolt*(nuts,bolts);  
 }  
}  
  
class HashNodeChar  
{  
 int key;  
 char value;  
 HashNodeChar next;  
  
 public HashNodeChar (char value)  
 {  
 this.value = value;  
 this.next = null;  
 }  
}  
  
class HashLLChar  
{  
 int size=10;  
 HashNodeChar[] ht;  
  
 public HashLLChar (int size)  
 {  
 this.size=size;  
 this.ht = new HashNodeChar[size];  
 }  
  
 public void insert(char k)  
 {  
 int v;  
  
 v = hashFunc(k);  
  
 HashNodeChar n = new HashNodeChar(k);  
  
 if(ht[v] == null)  
 {  
 ht[v] = n;  
 }  
  
 else  
 {  
 HashNodeChar curr = ht[v];  
  
 while(curr.next!=null)  
 {  
 curr=curr.next;  
 }  
  
 curr.next= n;  
 }  
 }  
  
 public boolean searchKey(char v)  
 {  
 boolean found;  
  
 found = false;  
  
 for(int i=0; i<ht.length; i++)  
 {  
 if (ht[i] == null)  
 {  
 continue;  
 }  
  
 else  
 {  
 HashNodeChar curr = ht[i];  
  
 while(curr!=null)  
 {  
 if(curr.value==v)  
 {  
 found=true;  
 }  
  
 curr=curr.next;  
 }  
 }  
 }  
  
 return found;  
 }  
  
 public void search(char v)  
 {  
 if (searchKey(v))  
 {  
 System.*out*.println("Found "+v);  
 }  
  
 else  
 {  
 System.*out*.println("Not found "+v);  
 }  
 }  
  
 public int hashFunc(char k)  
 {  
 return Math.*abs*(k)%size;  
 }  
}

Output:

