```
/******************************************************************************

                            Online C++ Compiler.
                Code, Compile, Run and Debug C++ program online.
Write your code in this editor and press "Run" button to compile and execute it.

*******************************************************************************/
/*
Subject : DSA Laboratory
Class   : SE Computer Engineering
Practical No: 02 - "A C++ Program to implement Set an ADT using Hash Table"
                a) Insert Elements in Set
                b) Search Elements in Set
                c) Delete Elements from Set
                d) Union of two Sets
                e) Intersection of two Sets
                f) Difference of two Sets
*/

                        //....Header Files
#include <iostream>
using namespace std;

                        //....Create Hash Table to store Set Elements
int HT1[10];                    //....for Set-A
int HT2[10];                    //....for Set-B
int size = 10;
                        //....to initialise Hash Tables/Set-A,B
void init ()
{
  for (int i = 0; i < size; i++)
    {
      HT1[i] = 0;
      HT2[i] = 0;
    }
}

void display ()
{
  int i;

  cout << "\n\t ------Hash Table - 01 ------";
  cout << "\n\t Index - Key";
  for (int i = 0; i < size; i++)
    {
      cout << "\n\t " << i << "\t " << HT1[i];
    }

  cout << "\n\t ------Hash Table - 02 ------";
  cout << "\n\t Index - Key";
  for (int i = 0; i < size; i++)
    {
```

```
      cout << "\n\t " << i << "\t " << HT2[i];
    }
}


void insert_SetA (int key)
{
  int index;

  index = key % size;              //....Hash Function: Division Method

  HT1[index] = key;
}

void insert_SetB(int key)
{
    int index;

    index = key % size;  //....Hash Function: Division Method

    HT2[index] = key;
}

void search_SetA(int key)
{
    int index;

    cout<<"\n\n Searching "<<key<<" in Set A";
    index = key % size;  //....Hash Function: Division Method

    if(HT1[index] == key)
        cout<<"\n\t Key"<<key<<" Found at "<<index;
    else
        cout<<"\n\t Key"<<key<<" Not Found....!!!";
}

void search_SetB(int key)
{
    int index;

    cout<<"\n\n Searching "<<key<<" in Set B";
    index = key % size;  //....Hash Function: Division Method

    if(HT2[index] == key)
        cout<<"\n\t Key"<<key<<" Found at "<<index;
    else
        cout<<"\n\t Key"<<key<<" Not Found....!!!";
}
```

```
void delete_SetA(int key)
{
   int index;

   cout<<"\n\n Deleting "<<key<<" from Set A";
   index = key % size;  //....Hash Function: Division Method

   if(HT1[index] == key)
   {
      HT1[index] = 0;
      cout<<"\n\t Key"<<key<<" Found at "<<index<<" and Deleted.";
   }

   else
      cout<<"\n\t Key"<<key<<" Not Found....!!!";
}



void delete_SetB(int key)
{
   int index;

   cout<<"\n\n Deleting "<<key<<" from Set B";
   index = key % size;  //....Hash Function: Division Method

   if(HT2[index] == key)
   {
      HT2[index] = 0;
      cout<<"\n\t Key"<<key<<" Found at "<<index<<" and Deleted.";
   }

   else
      cout<<"\n\t Key"<<key<<" Not Found....!!!";
}



int dup(int val)
{
   int i, dupl = 0;

   for(i = 0; i < size; i++)
   {
      if(val == HT1[i])
        dupl = 1;
   }
   return dupl;
}
```

```
void unionAB()
{
   int i, j;
   int C[10];

   j = 0;

   for(i = 0; i < size; i++)
   {
      if(HT1[i] != 0)     //....Copy Set-A in Set C
      {
        C[j] = HT1[i];
        j++;
      }
   }
   for(i = 0; i < size; i++)
   {
      if(HT2[i] != 0)     //....Copy Set-A in Set C
      {
        if(dup(HT2[i]) == 0)
        {
          C[j] = HT2[i];
          j++;
        }
      }
   }

   cout<<"\n\t Union of SET-A,B = ";
   for(i = 0; i < j; i++)
     cout<<C[i]<<", ";

}


int main ()
{
  cout << "--- A C++ Program to implement Set an ADT using Hash Table --- ";

  init ();

  display ();

  insert_SetA(11);
  insert_SetA(13);
  insert_SetA(15);
  insert_SetA(17);
  insert_SetA(19);

  insert_SetB(12);
  insert_SetB(14);
  insert_SetB(16);
  insert_SetB(18);
```

```
  insert_SetB(20);

  display ();

  search_SetA(13);
  search_SetA(20);
  search_SetB(14);
  search_SetB(17);

  delete_SetA(11);
  delete_SetA(20);

  delete_SetB(14);
  delete_SetB(19);

  display ();

  insert_SetA(20);
  display ();
  unionAB();

  return 0;
}

/*
--- A C++ Program to implement Set an ADT using Hash Table ---
        ------Hash Table - 01 ------
        Index - Key
        0       0
        1       0
        2       0
        3       0
        4       0
        5       0
        6       0
        7       0
        8       0
        9       0
        ------Hash Table - 02 ------
        Index - Key
        0       0
        1       0
        2       0
        3       0
        4       0
        5       0
        6       0
        7       0
        8       0
        9       0
        ------Hash Table - 01 ------
        Index - Key
```

```
0	0
1	11
2	0
3	13
4	0
5	15
6	0
7	17
8	0
9	19
------Hash Table - 02 ------
Index - Key
0	20
1	0
2	12
3	0
4	14
5	0
6	16
7	0
8	18
9	0

Searching 13 in Set A
	Key13 Found at 3

Searching 20 in Set A
	Key20 Not Found....!!!

Searching 14 in Set B
	Key14 Found at 4

Searching 17 in Set B
	Key17 Not Found....!!!

Deleting 11 from Set A
	Key11 Found at 1 and Deleted.

Deleting 20 from Set A
	Key20 Not Found....!!!

Deleting 14 from Set B
	Key14 Found at 4 and Deleted.

Deleting 19 from Set B
	Key19 Not Found....!!!
	------Hash Table - 01 ------
	Index - Key
	0	0
	1	0
	2	0
	3	13
```

```
4	0
5	15
6	0
7	17
8	0
9	19
------Hash Table - 02 ------
Index - Key
0	20
1	0
2	12
3	0
4	0
5	0
6	16
7	0
8	18
9	0
------Hash Table - 01 ------
Index - Key
0	20
1	0
2	0
3	13
4	0
5	15
6	0
7	17
8	0
9	19
------Hash Table - 02 ------
Index - Key
0	20
1	0
2	12
3	0
4	0
5	0
6	16
7	0
8	18
9	0
Union of SET-A,B = 20, 13, 15, 17, 19, 12, 16, 18,

...Program finished with exit code 0
Press ENTER to exit console.
*/
```