

# **ASSIGNMENT 1**

## **AIM:-**

To create ADT that implement the "set" concept.

- a) Add (newElement) -Place a value into the set
- b) Remove (element)
- c) Contains (element) Return true if element is in collection
- d) Size () Return number of values in collection
- e) Intersection of two sets
- f) Union of two sets
- g) Difference between two sets
- h) Subset

## **OBJECTIVE:**

To get the thorough understanding of the concepts of sets and the various operations performed on it

## **THEORY:-**

- **SET:-**

A **Set** is an unordered collection of objects, known as elements or members of the set.

An element 'a' belong to a set A can be written as ' $a \in A$ ', ' $a \notin A$ ' denotes that a is not an element of the set A.

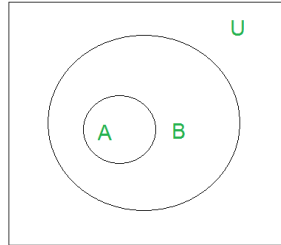
- **EQUAL SETS:-**

Two sets are said to be equal if both have same elements. For example  $A = \{1, 3, 9, 7\}$  and  $B = \{3, 1, 7, 9\}$  are equal sets.

- **SUBSET:-**

A set A is said to be **subset** of another set B if and only if every element of set A is also a part of other set B.

Denoted by ' $\subseteq$ '. ' $A \subseteq B$ ' denotes A is a subset of B.



- **SIZE OF A SET:-**

Size of a set can be finite or infinite.

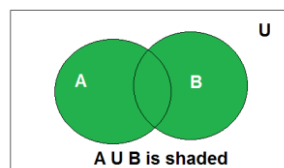
Size of the set S is known as Cardinality number, denoted as  $|S|$ .

Example: Let A be a set of odd positive integers less than 10.

Solution :  $A = \{1,3,5,7,9\}$ , Cardinality of the set is 5, i.e.,  $|A| = 5$ .

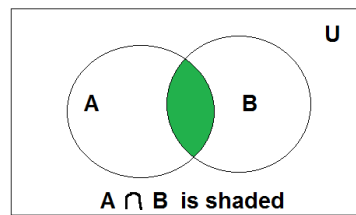
- **UNION :**

Union of the sets A and B, denoted by  $A \cup B$ , is the set of distinct element belongs to set A or set B, or both.



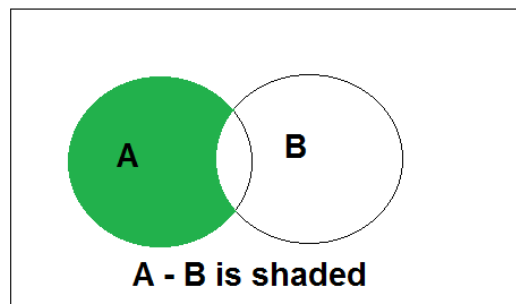
- **INTERSECTION:**

The intersection of the sets A and B, denoted by  $A \cap B$ , is the set of elements belongs to both A and B i.e. set of the common element in A and B



## • SET DIFFERENCE:-

Difference between sets is denoted by ' $A - B$ ', is the set containing elements of set A but not in B. i.e all elements of A except the element of B.



## ALGORITHM:

### • FOR INTERSECTION:

*Step 1: Take an empty set (intersection set)*

*Step 2: pass each element of set-2 and the entire set-1 to the function member()*

*Step 3: if it returns true,*

*Add that element to the intersection set*

### • FOR UNION:

*Step 1: Take an empty set (union set)*

*Step 2: copy all the elements of set1 to this new set*

*Step 3: traverse through the set2 and pass each element of set-2 along with the entire set-1 to the function member(),*

*and if it returns false then add that specified element to the union set*

- **FOR CONTAINS:**

*Step1 : take the number as input which you want to search*

*Step 2 : enter 1 for searching in set-1 or 2 for searching in set-2*

*Step 3 : initialise i=0*

*Step 4: traverse the set-1 or set-2 till the end depending on whether the input was 1 or 2 after passing the element and that set to the function member()*

*Step 5: if element found then display element is present*

- **FOR SUBSET:**

*Step 1: Enter 1 if you want to check if set 2 is subset of 1, or enter 2 if you want to check if set-1 is subset of set-2.*

*Step 2: Depending on input we will traverse the set(which has to be the subset) until its end, by passing each element of this set and other set to the function member().*

*Step 3: If member() returns true, then continue  
else return false*

*Step 4: If false, then display "it is a subset"  
else display "it is not"*

- **FOR DIFFERENCE :**

*Step 1: Initialise the difference set to 0, difference set contains all the element. which are in set-1 but not in set-2*

*Step 2: Traverse the entire set-1 and, pass each element of this set and the set-2 to the function member()*

*Step 3: if it returns false then add this element to the difference set*

- **FOR REMOVE:**

*Step 1: enter 1 or 2 for removing element from set-1 or set-2 respectively*

*Step 2: enter the index from which you want to remove the element*

*Step 3: if entered position is less than the size of the set then move all the elements to their left from the position at which you want to remove the element and just decrease the size of the set else, entered position is equal to the size of the set then just decrease the size*

- **FOR SIZE:**

*step 1: show the  $0^{th}$  index of the set which contains the size of our set*

## **Program:**

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

void create(int set[])
{
    int n;
    cout<<"\n enter the size of set : ";
    cin>>n;
    cout<<"\n enter the elements in the set : ";
    for(int i=1;i<=n;i++)
        cin>>set[i];

    set[0]=n;
}

bool member(int set[],int num)
{
    for(int i=1;i<=set[0];i++)
```

```

        if(set[i]==num)
            return true;

    return false;
}

void intersection(int set1[],int set2[],int set3[])
{
    for(int i=1;i<=set2[0];i++)
    {
        if(member(set1,set2[i])== true)
        {
            set3[0]++;
            set3[set3[0]]=set2[i];
        }
    }
}

void union1(int set1[],int set2[],int set4[])
{
    for(int i=0;i<=set1[0];i++)
        set4[i]=set1[i];

    for(int i=1;i<=set2[0];i++)
    {
        if(member(set1,set2[i])== false)
        {
            set4[0]++;
            set4[set4[0]]=set2[i];
        }
    }
}

void difference1(int set1[],int set2[],int set5[])
{
    for(int i=1;i<=set1[0];i++)
    {

```

```

        if((member(set2,set1[i]) == false))
        {
            set5[0]++;
            set5[set5[0]]=set1[i];
        }
    }
; }
}

```

```

void contains(int set[])
{
    int num;
    cout<<"\n enter the element to be searched ";
    cin>>num;
    if((member(set,num))== true)
        cout<<"\n element is present ";
    else
        cout<<"\n element is not present ";
}

```

```

bool subset(int seta[],int setb[])
{
    for(int i=1;i<=setb[0];i++)
    {
        if((member(seta,setb[i]))==true)
            continue;
        else
            return false;
    }
    return true;
}

```

```

void remove(int set[])
{
    int pos;
    cout<<"\n enter the position from which you want to remove the
element : ";
    cin>>pos;
    if(pos<=set[0])
    {
        if(pos<set[0])
        {

```

```

        for(int i=pos;i<=set[o];i++)
        {
            set[i]=set[i+1];
        }
        set[o]--;
    }
    else if(pos==set[o])
    {
        set[o]--;
    }
}
else
{
    cout<<"\n entered position exceeds the size of the set " ;
}
}

```

```

void size(int set[])
{
    cout<<set[o];
}

```

```

void display(int set[])
{
    cout<<"\n size : "<<set[o]<<"\t";
    for(int i=1;i<=set[o];i++)
    {
        cout<<set[i]<<" ";
    }
}

```

```

int main()
{
    int set1[10];
    cout<<"\n FOR SET 1 ";
    create(set1);
}

```



```

int set2[10];
cout<<"\n FOR SET 2 ";
create(set2);

int ch,c;
char choice;
do{
cout<<"\n\n ----- OPERATION MENU ----- ";
cout<<"\n 1 for INTERSECTION ";
cout<<"\n 2 for UNION ";
cout<<"\n 3 for DIFFERENCE ";
cout<<"\n 4 for CONTAINS( if element is present in set or not)";
cout<<"\n 5 for SUBSET";
cout<<"\n 6 for REMOVE";
cout<<"\n 7 for SIZE";
cout<<"\n 8 for DISPLAY";
cout<<"\n 9 for EXIT";
cout<<"\n\n Enter your choice : ";
cin>>ch;
switch(ch)
{
    case 1:
        {
            int set3[1];
            set3[0]=0;
            cout<<"\n the intersection of two sets : \t";
            intersection(set1,set2,set3);
            display(set3);
            break;
        }
    case 2:
        {
            int set4[set1[0]+1];
            set4[0]=0;
            cout<<"\n the union of two sets \t";
            union1(set1,set2,set4);
            display(set4);
            break;
        }
    case 3:
        {

```

```

        int set5[1];
        set5[0]=0;
        cout<<"\n the difference of two sets \t";
        difference1(set1,set2,set5);
        display(set5);
        break;
    }
    case 4:
    {
        cout<<"\n enter 1 for searching in set1 and 2 for searching
in set2 ";
        cin>>c;
        switch(c)
        {
            case 1: contains(set1); break;
            case 2: contains(set2); break;
            default: cout<<"\n wrong choice entered ";
        }

        break;
    }

    case 5:
    {
        label:
        cout<<"\n enter 1 for checking if set1 is subset of set2 else
enter 2 ";
        cin>>c;
        switch(c)
        {
            case 1:
            {
                if(subset(set2,set1)==true )
                    cout<<"\n set1 is subset of set2";
                else
                    cout<<"\n set1 is not a subset of set2";
                break;
            }
            case 2:
            {
                if(subset(set1,set2)==true )

```

```

        cout<<"\n set2 is subset of set1";
    else
        cout<<"\n set2 is not a subset of set1";
        break;
    }
    default:
        cout<<"\n wrong choice entered ";
        goto label;
    }
    break;
}
case 6:
{
    label2:
        cout<<"\n enter 1 for removing element from set 1 and 2 for
removal from set 2 ";
        cin>>c;
        switch(c)
        {
            case 1:
                {
                    remove(set1);
                    break;
                }
            case 2:
                {
                    remove(set2);
                    break;
                }
            default:
                cout<<"\n wrong choice entered ";
                goto label2;
            }
        break;
    }
case 7:
{
    cout<<"\n size of set 1 : ";
    size(set1);
    cout<<"\n size of set 2 : ";
    size(set2);

```

```

        break;
    }
    case 8:
    {
        display(set1);
        display(set2);
        break;
    }

    case 9:
        exit(0);
    default:
        cout<<"\n wrong choice entered ";
}

cout<<"\n want to continue with the operation?(y/n) :";
cin>>choice;

}while((choice=='y')||(choice=='Y'));

return 0;
}

```

## OUTPUT:

```
assignment1.cpp x
C:\Users\hkp10\Desktop\assignment1.exe

FOR SET 1
enter the size of set : 4

enter the elements in the set : 1 2 3 4

FOR SET 2
enter the size of set : 4

enter the elements in the set : 4 5 6 7

----- OPERATION MENU -----
1 for INTERSECTION
2 for UNION
3 for DIFFERENCE
4 for CONTAINS( if element is present in set or not)
5 for SUBSET
6 for REMOVE
7 for SIZE
8 for DISPLAY
9 for EXIT

Enter your choice : 1

the intersection of two sets :
size : 1      4
want to continue with the operation ?(y/n) :y

& others

assignment1.cpp x
C:\Users\hkp10\Desktop\assignment1.exe

----- OPERATION MENU -----
1 for INTERSECTION
2 for UNION
3 for DIFFERENCE
4 for CONTAINS( if element is present in set or not)
5 for SUBSET
6 for REMOVE
7 for SIZE
8 for DISPLAY
9 for EXIT

Enter your choice : 2

the union of two sets
size : 7      1 2 3 4 5 6 7
want to continue with the operation ?(y/n) :y

----- OPERATION MENU -----
1 for INTERSECTION
2 for UNION
3 for DIFFERENCE
4 for CONTAINS( if element is present in set or not)
5 for SUBSET
6 for REMOVE
7 for SIZE
8 for DISPLAY
9 for EXIT

Enter your choice : 3
```

```

assignment1.cpp
C:\Users\hkp10\Desktop\assignment1.exe
3 for DIFFERENCE
4 for CONTAINS( if element is present in set or not)
5 for SUBSET
6 for REMOVE
7 for SIZE
8 for DISPLAY
9 for EXIT

Enter your choice : 3

the difference of two sets
size : 3 1 2 3
want to continue with the operation ?(y/n) :y

----- OPERATION MENU -----
1 for INTERSECTION
2 for UNION
3 for DIFFERENCE
4 for CONTAINS( if element is present in set or not)
5 for SUBSET
6 for REMOVE
7 for SIZE
8 for DISPLAY
9 for EXIT

Enter your choice :

C:\Users\hkp10\Desktop\assignment1.exe
8 for DISPLAY
9 for EXIT

Enter your choice : 4

enter 1 for searching in set1 and 2 for searching in set2 1
enter the element to be searched 3

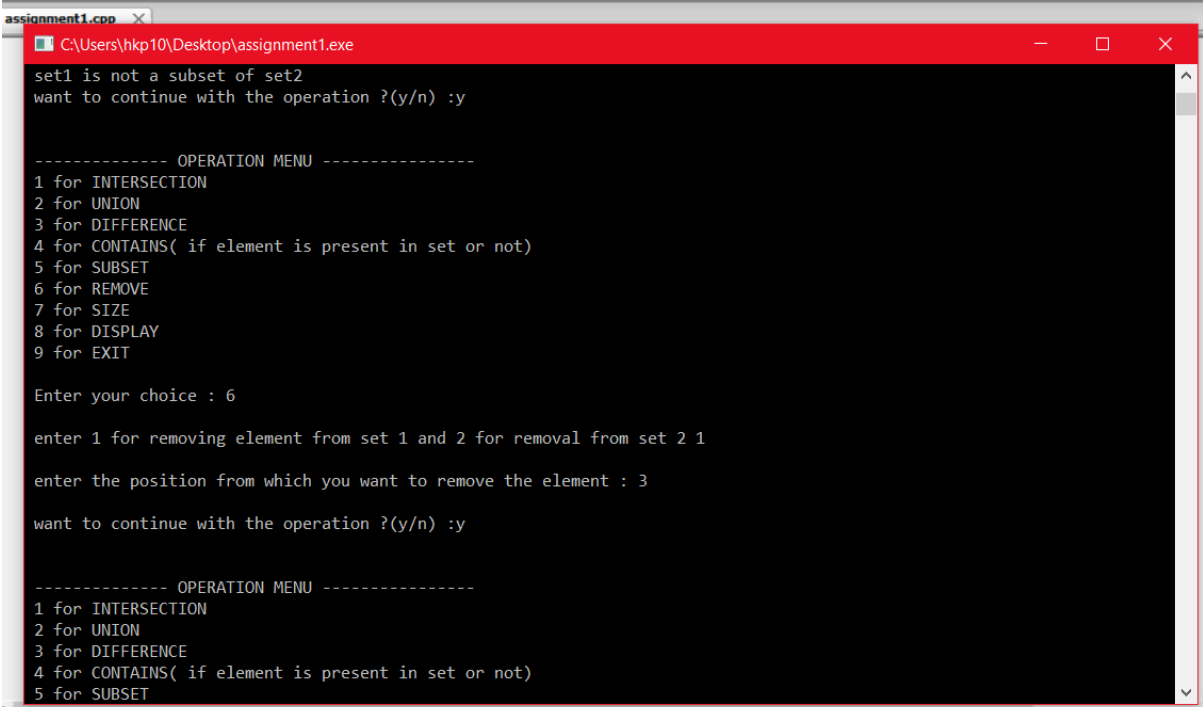
element is present
want to continue with the operation ?(y/n) :y

----- OPERATION MENU -----
1 for INTERSECTION
2 for UNION
3 for DIFFERENCE
4 for CONTAINS( if element is present in set or not)
5 for SUBSET
6 for REMOVE
7 for SIZE
8 for DISPLAY
9 for EXIT

Enter your choice : 5

enter 1 for checking if set1 is subset of set2 else enter 2 1

set1 is not a subset of set2
want to continue with the operation ?(y/n) :
  
```



```
assignment1.cpp
C:\Users\hkp10\Desktop\assignment1.exe
set1 is not a subset of set2
want to continue with the operation ?(y/n) :y

----- OPERATION MENU -----
1 for INTERSECTION
2 for UNION
3 for DIFFERENCE
4 for CONTAINS( if element is present in set or not)
5 for SUBSET
6 for REMOVE
7 for SIZE
8 for DISPLAY
9 for EXIT

Enter your choice : 6

enter 1 for removing element from set 1 and 2 for removal from set 2 1

enter the position from which you want to remove the element : 3

want to continue with the operation ?(y/n) :y

----- OPERATION MENU -----
1 for INTERSECTION
2 for UNION
3 for DIFFERENCE
4 for CONTAINS( if element is present in set or not)
5 for SUBSET
```

gs & others

Code::Blocks Search results Cpp Build log Build messages CppCheck/Ver++ CppCheck/Ver++ messages

## **CONCLUSION:**

We understood the concepts of sets and the various operations performed on them, and were able to apply those concepts through programming.