ASSIGNMENT 1

AIM: TO CREATE ADT TO PERFORM THE FOLLOWING SET OPERATIONS:

1. ADD (NEW ELEMENT) PLACE A VALUE IN A SET.
2. REMOVE(ELEMENT).
3. RETURNS TRUE IF ELEMENT IS IN COLLECTION.
4. SIZE() RETURNS NUMBER OF VALUES IN A COLLECTION.
5. INTERSECTION OF TWO SETS.
6. UNION OF TWO SETS.
7. DIFFERENCE BETWEEN TWO SETS
8. SUBSET.

OBJECTIVE: TO IMPLEMENT THE " SET " CONCEPT.

THEORY : A **set** is an abstract data type that can store unique values, without any particular order. It is a computer implementation of the mathematical concept of a finite set. Unlike most other collection types, rather than retrieving a specific element from a set, one typically tests a value for membership in a set. One may define the operations of the algebra of sets:

- union(*S*,*T*) : returns the union of sets *S* and *T*.
- intersection(*S*,*T*) : returns the intersection of sets *S* and *T*.
- difference(*S*,*T*) : returns the difference of sets *S* and *T*.
- subset(*S*,*T*) : a predicate that tests whether the set *S* is a subset of set *T*.

ALGORITHM:

Union*:*

```
1) Initialize union U as empty.
2) Copy all elements of first array to U.
3) Do following for every element x of second array:
.....a) If x is not present in first array, then copy x to U.
4) Return U.
```

Intersection:

1) Initialize intersection I as empty.

2) Do following for every element x of first array

…..a) If x is present in second array, then copy x to I.

4) Return I.

CODE:

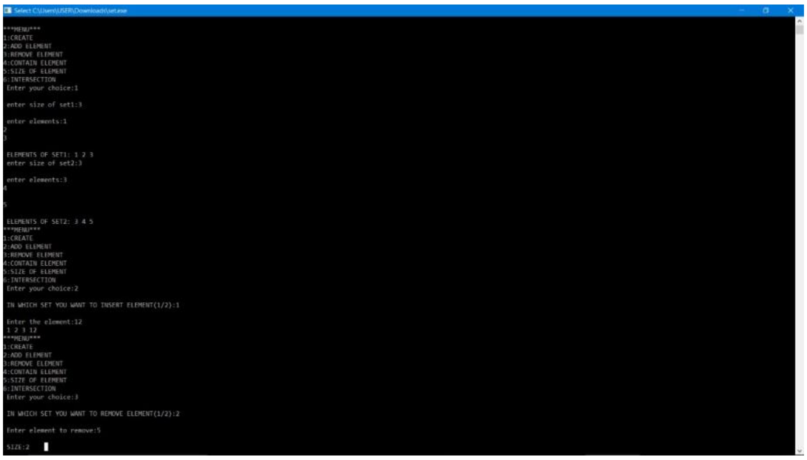
```
#include<iostream>
using namespace std;


void create(int *s1,int *s2);
void display(int *s);
void intersection(int *s1,int *s2);
void insert(int *s);
void remove(int *s);
void contain(int *s);
void set_size(int *s);
void intersection(int *s1,int *s2);
int linear(int *s,int e);
#define SIZE 20


int main()
{
        int s1[SIZE], s2[SIZE];
        int element,ch,c,i,r;

        do{
                cout<<"\n***MENU***";
                cout<<"\n1:CREATE \n2:ADD ELEMENT \n3:REMOVE ELEMENT
\n4:CONTAIN ELEMENT \n5:SIZE OF ELEMENT \n6:INTERSECTION";
                cout<<"\n Enter your choice:";
                cin>>ch;
                switch(ch)
                {
                        case 1:create(s1,s2);
                                        break;
                        case 2: cout<<"\n IN WHICH SET YOU WANT TO INSERT
ELEMENT(1/2):";
                                        cin>>c;
                                        if(c==1)
                                            insert(s1);
                                        else
                                            insert(s2);
                                        break;
                        case 3:cout<<"\n IN WHICH SET YOU WANT TO REMOVE
ELEMENT(1/2):";
                                        cin>>c;
                                        if(c==1)
                                            remove(s1);
                                        else
                                            remove(s2);

                                        break;
                        case 4:cout<<"\n IN WHICH SET YOU WANT TO CHECK THE
ELEMENT(1/2):";
                                        cin>>c;
                                        if(c==1)
                                            contain(s1);
                                        else
                                            contain(s2);
                                        break;
                        case 5:cout<<"\n IN WHICH SET YOU WANT TO CHECK THE
SIZE(1/2):";
                                        cin>>c;
                                        if(c==1)
                                            set_size(s1);
                                        else
                                            set_size(s2);

                                        break;
                        case 6:intersection(s1,s2);
                        default: cout<<"\n WRONG CHOICE!!!";
                }

        }while(ch<6);
        return 0;
}
int linear(int *s, int e)
{
        int f;
        for(int i=1;i<=s[0];i++)
        {
                if(s[i]==e)
```

```
                {
                        f=1;
                        return f;
                }
        }
        if(f==0)
                return f;

}
void intersection(int *s1,int *s2)
{
        int s3[SIZE],i,j=1;
        for( i=1;i<=s1[0];i++)
        {
        if(linear(s2,s1[i])==1)
        {
                s3[j]=s1[i];
                }
        }
}
void set_size(int *s)
{
        cout<<"\n SIZE OF SET:"<<s[0];
}
void contain(int *s)
{
        int element;
        cout<<"\n Enter element to check:";
        cin>>element;
        if(linear(s,element)==1)
                cout<<"\n ELEMENT PRESENT!";
        else
                cout<<"\n ELEMENT NOT PRESENT!!!";


}
void remove(int *s)
{
        int element,i,j;
        cout<<"\n Enter element to remove:";
        cin>>element;
        for(i=1;i<=s[0];i++)
        {
```

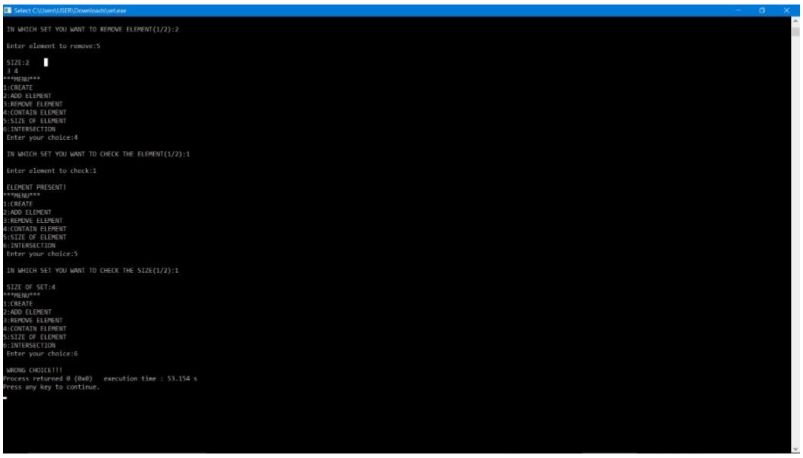
```
        if(s[i]==element)
        {
            for(int j=i;j<=s[0];j++)
                {
                    s[j]=s[j+1];
                }
                s[0]-=1;
                cout<<"\n SIZE:"<<s[0]<<"\n";
                display(s);
                return;
        }
    }
    cout<<"\n ELEMENT NOT FOUND!!!";
}

void insert(int *s)
{
    int element;
    cout<<"\n Enter the element:";
    cin>>element;
    int size=s[0];
    s[++size]=element;
    s[0]=size;
    display(s);
}

void create(int *s1,int *s2)
{
    int n,i;
    cout<<"\n enter size of set1:";
    cin>>n;
    s1[0]=n;
    cout<<"\n enter elements:";
    for(i=1;i<=n;i++)
    {
    cin>>s1[i];
    }
    cout<<"\n ELEMENTS OF SET1:";
    display(s1);
    cout<<"\n enter size of set2:";
    cin>>n;
    s2[0]=n;
    cout<<"\n enter elements:";
```

```
        for(i=1;i<=n;i++)
        {
        cin>>s2[i];
        }
        cout<<"\n ELEMENTS OF SET2:";
        display(s2);

}
void display(int *s)
{
        int i;
        for(i=1;i<=s[0];i++)
        {
                cout<<" "<<s[i];
        }
}
```

OUTPUT:



```
***MENU***
1:CREATE
2:ADD ELEMENT
3:REMOVE ELEMENT
4:CONTAIN ELEMENT
5:SIZE OF ELEMENT
6:INTERSECTION
 Enter your choice:1

 enter size of set1:3

 enter elements:1
2
3

 ELEMENTS OF SET1: 1 2 3
 enter size of set2:3

 enter elements:3
4

5

 ELEMENTS OF SET2: 3 4 5
***MENU***
1:CREATE
2:ADD ELEMENT
3:REMOVE ELEMENT
4:CONTAIN ELEMENT
5:SIZE OF ELEMENT
6:INTERSECTION
 Enter your choice:2

 IN WHICH SET YOU WANT TO INSERT ELEMENT(1/2):1

 Enter the element:12
 1 2 3 12
***MENU***
1:CREATE
2:ADD ELEMENT
3:REMOVE ELEMENT
4:CONTAIN ELEMENT
5:SIZE OF ELEMENT
6:INTERSECTION
 Enter your choice:3

 IN WHICH SET YOU WANT TO REMOVE ELEMENT(1/2):2

 Enter element to remove:5

 SIZE:2
```



```
 IN WHICH SET YOU WANT TO REMOVE ELEMENT(1/2):2

 Enter element to remove:5

 SIZE:2
 3 4
***MENU***
1:CREATE
2:ADD ELEMENT
3:REMOVE ELEMENT
4:CONTAIN ELEMENT
5:SIZE OF ELEMENT
6:INTERSECTION
 Enter your choice:4

 IN WHICH SET YOU WANT TO CHECK THE ELEMENT(1/2):1

 Enter element to check:1

 ELEMENT PRESENT!
***MENU***
1:CREATE
2:ADD ELEMENT
3:REMOVE ELEMENT
4:CONTAIN ELEMENT
5:SIZE OF ELEMENT
6:INTERSECTION
 Enter your choice:5

 IN WHICH SET YOU WANT TO CHECK THE SIZE(1/2):1

 SIZE OF SET:4
***MENU***
1:CREATE
2:ADD ELEMENT
3:REMOVE ELEMENT
4:CONTAIN ELEMENT
5:SIZE OF ELEMENT
6:INTERSECTION
 Enter your choice:6

 WRONG CHOICE!!!
Process returned 0 (0x0)   execution time : 53.154 s
Press any key to continue.
```

CONCLUSION: We saw all the algorithms the STL offers to operate on sets, that are collections of sorted elements, in the general sense.