DS Assignment

Q. Create a class Set and include functions to perform following Set operations on Sets: Subset , Union , Intersection , Complement , Set difference , Symmetric difference and Cartesian product . WAP which takes sets from user and use this class.

Code:-

```
//Program to create a set class and include different set functions
//Harsh Bamotra AC-1216

#include <iostream>
#include <math.h>
using namespace std;
class set
  {
private:                    //declaring private members
int m,n;
int a[100],b[100];

public:                             //declaring public members
        void input()                //defining function to input a set
        {
                cout<<"Enter the number of elements of first set::";
                        cin>>m;
                cout<<"Enter the set in sorted form ::" << endl;
                for(int i=0;i<m;i++)
                        {
                                cin>>a[i];
                        }
                        cout<<"Enter the number of elements of second set::";
                        cin>>n;
                        cout<<"Enter the set in sorted form::" << endl;
                        for(int i=0;i<n;i++)
                                {
                                        cin>>b[i];
                                }
                }
        void set_union();                       //declaring different function to perform set operations
        void intersection();
        void symmetric_diff();
        void set_difference();
        void cartesian_prod();
        void compliment();
```

```
        void subset();
    };

void set::set_union()                   //defining function for union
    {
        int i = 0, j = 0;
        while (i < m && j < n)
            {
                if (a[i] < b[j])
                    {
                        cout << a[i++] << " ";
                    }
                else if (b[j] < a[i])
                    {
                        cout << b[j++] << " ";
                    }
                else
                    {
                        cout << b[j++] << " ";
                        i++;
                    }
            }
        while (i < m)                   //Print remaining elements of the larger array
        cout << a[i++] << " ";
        while (j < n)
        cout << b[j++] << " ";
    }

void set:: intersection()                       //defining function for intersection
    {
        int i = 0, j = 0;
        while (i < m && j < n)
        {
                if (a[i] < b[j])
                    {
                        i++;
                    }
                else if (b[j] < a[i])
                    {
                        j++;
                    }
                else                    //if arr1[i] == arr2[j]
                    {
                        cout << b[j] << " ";
                        i++;
                        j++;
                    }
            }
    }
```

```
void set:: symmetric_diff()                    //defining function for symmetric difference
  {
        int i = 0, j = 0;
        while (i < m && j < n)
        {
                        if (a[i] < b[j])
                        {
                                cout << a[i] << " ";
                                i++;
                        }
                 else if (b[j] < a[i])
                        {
                                cout << b[j] << " ";
                                j++;
                        }
                else
                        {
                                i++;
                                j++;
                        }
        }
        while (i < m)
                {
                        cout << a[i] << " ";
                        i++;
                }
        while (j < n)
                {
                        cout << b[j] << " ";
                        j++;
                }
  }

void set:: set_difference()                    //defining function for set difference
  {
        int c=0;
        for(int i=0;i<m;i++)
                {
                        c=0;
                        for(int j=0;j<n;j++)
                                {
                                        if(a[i]==b[j])
                                        c++;
                                }
                        if(c==0)
                                {
                                        cout<<a[i]<<" ";
                                }
                }
  }
```

```
void set:: cartesian_prod()                    //defining function for Cartesian product
  {
        for (int i = 0; i < m; i++)
        for (int j = 0; j < n; j++)
        cout<< "("<<a[i]<<","<<b[j]<<")"<<endl;
  }

void set:: compliment()                 //defining function for complement
  {
        int u,c=0;
        cout<<"Enter the size of universal set::";
        cin>>u;
        int uni[u];
        cout << "Enter the elements of universal set ::" << endl;
        for(int i=0;i<u;i++)
                {
                        cin>>uni[i];
                }
        cout<<"The complement of set A ::" << endl;
        for(int i =0;i<u;i++)
                {
                        c=0;
                        for(int j=0;j<m;j++)
                                {
                                        if (uni[i]==a[j])
                                        c++;
                                }
                        if(c==0)
                                {
                                        cout<<uni[i]<<" ";
                                }
                }
        cout<<endl << "The complement of set B ::" << endl;
        for(int i =0;i<u;i++)
                {
                        c=0;
                        for(int j=0;j<n;j++)
                                {
                                        if (uni[i]==b[j])
                                        c++;
                                }
                        if(c==0)
                                {
                                        cout<<uni[i]<<" ";
                                }
                }
  }

void set::subset()                      //function for subset
```

```
{
        int count = pow(2, m);
        cout<<"Subset of set A" << endl;
        for (int i = 0; i < count; i++)
                {
                                for (int j = 0; j < m; j++)
                                        {
                                                if ((i & (1 << j)) != 0)
                                                cout << a[j] << " ";
                                        }
                                cout << "\n";
                }
        count = pow(2, m);
        cout<< endl << "Subset of set B ::" << endl;
        for (int i = 0; i < count; i++)
                {
                        for (int j = 0; j < n; j++)
                        {
                                        if ((i & (1 << j)) != 0)
                                        cout << b[j] << " ";
                        }
                                cout << "\n";
                }
}

int main()
{
        set s1;
        s1.input();
        cout<<"Union ::";
        s1.set_union();
        cout<<endl<<endl;
        cout<<"Intersection ::";
        s1.intersection();
        cout<<endl<<endl<<"Symetric difference ::";
        s1.symmetric_diff();
        cout<<endl<<endl<<"Set difference ::";
        s1.set_difference();
        cout<<endl<<endl<<"Cartesian product ::";
        s1.cartesian_prod();
        cout<<endl<<endl;
        s1.compliment();
        cout<<endl<<endl;
        cout<<"Subsets ::" << endl;
        s1.subset();
        return 0;
}
```

Output:

Command Prompt - set.exe

```
C:\Users\harsh\Desktop>g++ setss.cpp -o set.exe

C:\Users\harsh\Desktop>set.exe
Enter the number of elements of first set::3
Enter the set in sorted form ::
1
3
5
Enter the number of elements of second set::5
Enter the set in sorted form::
1
3
5
6
7
Union ::1 3 5 6 7

Intersection ::1 3 5

Symetric difference ::6 7

Set difference ::

Cartesian product ::(1,1)
(1,3)
(1,5)
(1,6)
(1,7)
(3,1)
(3,3)
(3,5)
(3,6)
(3,7)
(5,1)
(5,3)
(5,5)
(5,6)
(5,7)
```

```
Enter the size of universal set::10
Enter the elements of universal set ::
1
2
3
4
5
6
7
8
9
10
The complement of set A ::
2 4 6 7 8 9 10
The complement of set B ::
2 4 8 9 10

Subsets ::
Subset of set A

1
3
1 3
5
1 5
3 5
1 3 5

Subset of set B ::

1
3
1 3
5
1 5
3 5
1 3 5

C:\Users\harsh\Desktop>
```

Harsh Bamotra

AC-1216