#include <iostream>

using namespace std;

class Set

{

int \*arr;

int size;

public:

Set(int n)

{

size = n;

arr = new int[n];

}

void arrayInput()

{

cout << "Enter elements of array: " << endl;

for (int i = 0; i < size; i++)

cin >> arr[i];

}

void setFromArray(int Arr[], int n)

{

for (int i = 0; i < n; i++)

arr[i] = Arr[i];

}

void display()

{

for (int i = 0; i < size; i++)

cout << arr[i] << " ";

cout << endl;

}

void removeDuplicates()

{

for (int i = 0; i < size - 1; i++)

{

for (int j = i + 1; j < size; j++)

{

if (arr[i] == arr[j])

{

for (int k = j; k < size - 1; k++)

arr[k] = arr[k + 1];

size--;

j--;

}

}

}

}

int getSetSize()

{

return size;

}

bool isExist(int a)

{

for (int i = 0; i < size; i++)

{

if (arr[i] == a)

return true;

}

return false;

}

bool isSubset(Set s)

{

for (int i = 0; i < s.size; i++)

{

if (!isExist(s.arr[i]))

return false;

}

return true;

}

void setIntersection(Set s)

{

for (int i = 0; i < s.size; i++)

{

if (isExist(s.arr[i]))

cout << s.arr[i] << " ";

}

cout << endl;

}

void setUnion(Set s)

{

Set myUnion(s.size + size);

for (int i = 0; i < size; i++)

myUnion.arr[i] = arr[i];

for (int i = 0; i < s.size; i++)

myUnion.arr[i + size] = s.arr[i];

myUnion.removeDuplicates();

myUnion.display();

}

void complement(Set universalSet)

{

for (int i = 0; i < universalSet.size; i++)

{

if (!isExist(universalSet.arr[i]))

cout << universalSet.arr[i] << " ";

}

cout << endl;

}

void difference(Set B)

{

// A(\*this) - B

for (int i = 0; i < size; i++)

{

if (!B.isExist(arr[i]))

cout << arr[i] << " ";

}

cout << endl;

}

void symmetricDifference(Set B)

{

// A(\*this) - B

for (int i = 0; i < size; i++)

{

if (!B.isExist(arr[i]))

cout << arr[i] << " ";

}

// B - A(\*this)

for (int i = 0; i < B.size; i++)

{

if (!isExist(B.arr[i]))

cout << B.arr[i] << " ";

}

cout << endl;

}

void cartesianProduct(Set B)

{

// A(\*this) x B

for (int i = 0; i < size; i++)

{

for (int j = 0; j < B.size; j++)

cout << "(" << arr[i] << "," << B.arr[j] << ")" << endl;

}

}

};

int main()

{

int arr1[] = {1, 2, 3, 4, 5};

int arr2[] = {5, 7, 8};

Set A(5), B(3);

A.setFromArray(arr1, 5);

B.setFromArray(arr2, 3);

cout << "A -> ";

A.display();

cout << "B -> ";

B.display();

cout << "A subset of B: " << A.isSubset(B) << endl;

cout << "A union B : ";

A.setUnion(B);

cout << "A intersection B : ";

A.setIntersection(B);

cout << "Difference : ";

A.difference(B);

cout << "Symmetric Difference : ";

A.symmetricDifference(B);

cout << "Cartesian Product" << endl;

A.cartesianProduct(B);

return 0;

}

