#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 display()

{

cout << "ArraySet -> ";

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--;

}

}

}

}

static bool checkBit(int n, int i)

{

int mask = 1 << i;

return (n & mask) != 0;

}

int getSetSize(){

return size;

}

bool isExist(int a)

{

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

{

if (arr[i] == a)

return true;

}

return false;

}

void getPowerSet()

{

int n = 1 << size; // 2\*\*n subsets

cout << "Power Set" << endl;

for (int i = 0; i < n; i++) // binary numbers of size bits

{

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

{

if (checkBit(i, j))

cout << arr[j] << " ";

}

cout << endl;

}

}

};

int main()

{

Set s(8);

s.arrayInput();

s.display();

s.removeDuplicates();

s.display();

s.getPowerSet();

// cout << Set::checkBit(2,1);

return 0;

}

