#include <iostream>

using namespace std;

typedef struct {

int\* arr;

int length;

} AwesomeArray;

void addElements(AwesomeArray & arr, int \* valuesToAdd, int numberOfValuesToAdd) {

// Allocate new memory

int \* newarr = new int[arr.length + numberOfValuesToAdd];

// Copy previous array values to new array

for(int i = 0; i < arr.length; i++){

newarr[i] = arr.arr[i];

}

// add new elements to new array

for(int i = 0; i < numberOfValuesToAdd; i++){

newarr[arr.length + i] = valuesToAdd[i];

}

//delete previous array

delete arr.arr;

arr.arr = newarr;

arr.length += numberOfValuesToAdd;

}

//add 1 element to function that add more elements

//but its not neccesery because function AddElements do same with one element

void addElement(AwesomeArray & arr, int valueToAdd) {

int newarr[1] = {valueToAdd};

addElements(arr, newarr , 1);

}

//function that delete element

//she do this, in so doing that she take place element that user want delete in another element that available one place forward

void deleteElement(AwesomeArray & arr, int indexToDelete) {

for(int i = indexToDelete; i < arr.length; i++){

arr.arr[i] = arr.arr[i + 1];

}

arr.length--;

}

//function that find value

//first line code : loop that pass on all array

//second line : check if in arrat have the value that selected . if have function return index

//last line : another result function return -1 . -1 Signifies that not find value index

int Find(AwesomeArray & arr , int FindValue ){

for(int i = 0; i < arr.length ; i++) {

if (arr.arr[i] == FindValue ) return i;

}

return -1;

}

// function that delete selected value

//first line of code : find index that i want delete

//second and third line: As long as i have index find in array so delete selected value

//last line: back to line 2

void deleteByValue(AwesomeArray & arr, int valueToDelete) {

int indexTodelete = Find(arr, valueToDelete);

while(indexTodelete != -1) {

deleteElement(arr, indexTodelete);

indexTodelete = Find(arr, valueToDelete);

}

}

int main()

{

//set data to array

AwesomeArray arr;

cout << "Enter the size of array: ";

cin >> arr.length;

arr.arr = new int[arr.length];// allocate new memory place

//loop to enter data to array

for(int i = 0; i < arr.length; i++){

cout << "Enter number" << i + 1 << ": ";

cin >> arr.arr[i];

}

cout << "thats all the numbers in the array: " << endl;

for(int i = 0; i < arr.length; i++){

cout << arr.arr[i] ;

if(i == arr.length - 1) {

cout << "." << endl;

}

else cout << ",";

}

while(true){

int command;// new variable

cout << "enter command that you want: " << endl;

cout << "(0 - add values, 1 - add one value , 2 - delete element, 3 - find index of value , 4 - print, 5 - exit): " << endl;

cin >> command;

// exit from loop

if (command == 5) break;

// if have a mistake in command so user will receive massage what to do now

if(command > 5){

cout << "worng command . try again: " << endl;

continue;

}

//command to add values

if(command == 0) {

int numberOfValues = 0;

cout << "Enter number of values to add: ";

cin >> numberOfValues;

int valuesToAdd[numberOfValues]; //amount the new numbers that will be added to original array

for(int i = 0; i < numberOfValues; i++) {

cout << "Enter number: ";

cin >> valuesToAdd[i];

}

addElements(arr, valuesToAdd,numberOfValues); // all numbers of array new and old enter to function

}

//command to add 1 value

else if(command == 1) {

int valueToAdd;

cout << "Enter value: ";

cin >> valueToAdd;

addElement(arr, valueToAdd);

}

// command to delete one element

else if(command == 2) {

int index;

cout << "Enter index to delete: ";

cin >> index;

deleteElement(arr, index);

}

// command to find index of value in array

else if(command == 3){

int toFind;

cout << "Enter number to find: ";

cin >> toFind;

for (int i = 0; i < arr.length; i ++){

if (arr.arr[i] == toFind) {

cout << "Value at index " << Find(arr, toFind) << endl;

}

}

}

// print result for display

else if(command == 4){

for( int i = 0; i < arr.length; i++){

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

}

cout << endl;

}

}

}