CS 201: Problem Solving & Programming II

Lab # 7

Exercise 1

Dynamic array are useful when we want set variable size of an array. Let’s assume that, we have a big file of integer values from which we have to find a specific value. The count of values is given at the beginning of the file, followed by the values themselves.

**Input**

1. The input of this program will be a file containing the integer values. You have to load those values in an array. The first line of the input file (values.txt) will be the count value followed by the actual integer values starting from the second line. A sample file:

10

1 1 2 3 5 8 13 21 34 55

If a value appears more than once (like the one in sample file) ignore the duplicates.

|  |
| --- |
| **Note:** We can define a dynamic array in the following manner-  cout << “specify the size: ”;  int size = 0;  cin >> size;  int\* d\_array = new int[size];  d\_array[0] = 10;  delete []d\_array; // free the dynamic array  Look, we are taking the value of array size from user input! Also notice that although the array is a pointer variable we can access its elements like we do with regular array. |

1. Print the array in console to make sure that the array is identical to the input values.
2. Have the user input a number, then do a search for that number within the array.

**Output**

If the input value is found in the array print its index otherwise print “value doesn’t exists”.

Search a value: 13

Index: 7

Search a value: 10

Value doesn’t exist.

Don’t forget to free the dynamic array, at the end of the program.

Exercise 2

Remember what you did in the mid-term exam section 3. There was an extra credit task where you had to print the list of items the user has bought. Now implement that part using dynamic array.

**How can we I that? How can I increase the size of an already declared dynamic array?**

Here’s one way to increase the size of a dynamic array:

1. Add a variable(counter) to keep track how many items user have selected.
2. Every time user adds an item, increase the size of the counter.
3. Create a new array whose size is equal to the counter value.
4. Then copy the contents of the existing array to the new array.
5. Use the remaining spaces to hold the new data.

For example in the following code, the size of the new array is increased by one:

int size = 1;

int\* d\_array = new int[size];

// put the user’s first choice in d\_array

// when user adds another item...

int\* new\_array = new int[size + 1];

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

new\_array[i] = d\_array[i]; // add the old items

}

// add the new item ...

Use this technique to increase the size of the array that holds user’s choice. Whenever the user adds a new item in his cart, increase the array size by one and hold the newly added item there. At the end print the array information to console.