**OBJECTIVE** : String Operations, Usage of Sorting and Binary Search Algorithms, Usage of Binary Search and Merge Algorithms

**Instructor** : Yusuf Evren AYKAÇ
**Assistants** : Elif GÜL, Yusuf Şevki GÜNAYDIN, Hatice ÇATALOLUK

**1.**

The phone numbers of the companies are stored in the **phones.txt**. The area codes are missing in some of the phone numbers. (A phone number including area code is 10 characters). You will write a program to format phone numbers and add area code if necessary. At the end all the phone numbers will look like the following; `(312) 4250204`.

Write the following functions;

- **readFromFile** that reads all the phone numbers from the file **phones.txt** into a string array. The function should return the number of phones.

- **updateDirectory** that takes the string array which keeps the phone numbers and the number of phones in the array as parameters, and updates the phone numbers adding the parentesis and area code (312) where necessary (i.e `(312) 4250204`).

- **printDirectory** that takes the string array which keeps the phone numbers and the number of phones in the array as parameters, and displays the number of phones and the phone list on the screen.

**Project Name:** LabGuide4_1
**File Name:** Question_1.cpp

**Example Run:**
```
There are 8 phone numbers in the directory
(312) 4250204
(312) 2660408
(312) 2748540
(312) 3659874
(312) 3547862
(312) 2660808
(312) 2905128
(312) 4421578
```

**phones.txt**
```
3124250204
2660408
3122748540
3123659874
3123547862
2660808
2905128
3124421578
```

**2.**

Write a function that replaces a given string with '*' within a given text if that string is a full word, or the beginning or end of a word. Test your function with a suitable main.

**Example Run#1:**
```
Enter a text:IS THIS LIST STARTING WITH ISTANBUL
Enter a string:IS
Old Text: IS THIS LIST STARTING WITH ISTANBUL
New Text: * TH* LIST STARTING WITH *TANBUL
```

**Example Run#2:**
```
Enter a text:HIS NAME IS ALIS
Enter a string:IS
Old Text: HIS NAME IS ALIS
New Text: H* NAME * AL*
```

**Project Name:** LabGuide4_2
**Source Name:** Question_2.cpp

```
BUBBLE SORT ALGORITHM:
1. Repeat
2. Initialize sorted 1
   3. Repeat for each pair of adjacent array elements
   4. If the values in a pair are out of order
      4.1. Exchange the values
      4.2. Set sorted to 0
 as long as the array is not sorted

SELECTION SORT ALGORITHM:
1. For each value of fill from 0 to n – 1
2. Find index_of_min, the index of the smallest element in the unsorted sub array list[fill] through list[n-1].
3. If fill is not the position of the smallest element (index_of_min)
4. Exchange the smallest element with the one at position fill.
```

3. Write a C program that will get the points for several football teams from a file "**points.txt**" and:

   **a)** Sort the points with <u>selection sort</u> algorithm in **ascending order** and display them on the screen.

**Example Run:**
```
Points
------
16
17
20
25
26
26
27
29
30
30
31
32
35
35
40
41
49
53
```

| points.txt |
|------------|
| 41 |
| 40 |
| 35 |
| 26 |
| 30 |
| 17 |
| 16 |
| 29 |
| 53 |
| 49 |
| 27 |
| 26 |
| 35 |
| 32 |
| 31 |
| 30 |
| 25 |
| 20 |

**Project_name:** Labguide4_3a
**File_name:** Question_3a.cpp

   **b)** Sort the points with <u>bubble sort</u> algorithm in **descending order** and display them on the screen.

**Example Run:**
```
Points
------
53
49
41
40
35
35
32
31
30
30
29
27
26
26
25
20
17
16
```

**Project_name:** Labguide4_3b
**File_name:** Question_3b.cpp

**4. a)** Write a C program that forms a Disco Song list by getting song names from the user until "end" is entered. Each song name will be added to the list and the list should be sorted in <u>ascending order</u> by using <u>bubble sort algorithm</u>. After sorting, the new form of the list will be displayed on the screen. Assume that NO duplicate value is given.

```
Example Run:                                     Disco Songs
Enter a song name (end to stop): Stanga          --------------------
Enter a song name (end to stop): Bonbon          1) Age of Emotions
Enter a song name (end to stop): OMG             2) Bonbon
Enter a song name (end to stop): Rockabye        3) But A Lie
Enter a song name (end to stop): Habib Galbi     4) Habib Galbi
Enter a song name (end to stop): Just Say        5) Just Say
Enter a song name (end to stop): But A Lie       6) OMG
Enter a song name (end to stop): Age of Emotions 7) Rockabye
Enter a song name (end to stop): end             8) Stanga
```

**b)** modify the program **Question_4a.cpp,** so the list will be sorted in descending order by using bubble sort algorithm.

```
Example Run:                                     Disco Songs
Enter a song name (end to stop): Stanga          --------------------
Enter a song name (end to stop): Bonbon          1) Stanga
Enter a song name (end to stop): OMG             2) Rockabye
Enter a song name (end to stop): Rockabye        3) OMG
Enter a song name (end to stop): Habib Galbi     4) Just Say
Enter a song name (end to stop): Just Say        5) Habib Galbi
Enter a song name (end to stop): But A Lie       6) But A Lie
Enter a song name (end to stop): Age of Emotions 7) Bonbon
Enter a song name (end to stop): end             8) Age of Emotions
```

**5. a)** Write a C Program which search a record of a car plate from the sorted data in **plates.txt** produced in. Use sequential search, and prints how many comparisons made to find the record.

**Project_name:** LabGuide4_5a
**File_name:** Question_5a.cpp

**Example Run:**
```
Enter plate of a car (END for exit): 06HTC452
EFE KOROGLU 06HTC452
14 comparisons.

Enter plate of a car (END for exit): 01KL5641
BARIS COLAK 01KL5641
1 comparisons.

Enter plate of a car (END for exit): 53EF4587
DEMIRCAN COSKUN 53EF4587
37 comparisons.

Enter plate of a car (END for exit): 06GHK567
NOT FOUND
37 comparisons.

Enter plate of a car (END for exit): END
```

| plates.txt | | |
|---|---|---|
| BARIS | COLAK | 01KL5641 |
| DENIZ | CORBACI | 01RR5678 |
| DENIZ | AGAH | 06ABC453 |
| OZAN | CERGEL | 06BA1246 |
| BARIS | ARSLAN | 06BB9987 |
| ERGIN | ERANT | 06BLN027 |
| ABDULLAH | ARSLAN | 06CAN063 |
| EMRE | DINCEL | 06CRN027 |
| MUSTAFA | KAPLAN | 06EMN652 |
| FATMA | KAYA | 06ERD063 |
| BERAT | KINA | 06ERK751 |
| OZGUR | AKBABA | 06ES4587 |
| HUSEYIN | KINIKLI | 06GNS697 |
| EFE | KOROGLU | 06HTC452 |
| CELAL | ERBAY | 06JAL254 |
| ABDULLAH | AKTAS | 06KM3657 |
| ONUR | BAGDADIOGLU | 06KRM323 |
| AYLIN | GURTUNA | 06KRM442 |
| BATURAY | HASER | 06MNV660 |
| MERTHAN | ILVAN | 06OIL633 |
| BARIS | INALOZ | 06PRL471 |
| ESIN | INANC | 06REN694 |
| DENIZ | INTEPE | 06RMZ458 |
| CAN | KANBAY | 06VYS452 |
| MUSTAFA | BOLAT | 07ABC487 |
| BILGEN | BILGIN | 07TRK658 |
| EDIZ | CITAK | 19HJ1245 |
| ALI | CICEK | 19HT6547 |
| DERYA | DEDEOGLU | 27CNR006 |
| AYSE | BAYHAN | 34KLM365 |
| ZEYNEP | AKANDIR | 34ZZ1785 |
| MERT | AKDEMIR | 35ED5678 |
| TAMER | CAN | 35TT2369 |
| CUNEYT | EKINCI | 35ZRF006 |
| GAMZE | BASIBUYUK | 42HKN451 |
| ABDULLAH | BATTAL | 42MN4178 |
| DEMIRCAN | COSKUN | 53EF4587 |

**b)** Modify the program **Question_1a.cpp** using <u>Binary Search algorithm</u>.

**Example Run:**
```
Enter plate of a car (END for exit): 06HTC452
EFE KOROGLU 06HTC452
3 comparisons.

Enter plate of a car (END for exit): 01KL5641
BARIS COLAK 01KL5641
5 comparisons.

Enter plate of a car (END for exit): 53EF4587
DEMIRCAN COSKUN 53EF4587
6 comparisons.

Enter plate of a car (END for exit): 06GHK567
NOT FOUND
6 comparisons.

Enter plate of a car (END for exit): END
```

**Project_name:** LabGuide4_5b
**File_name:** Question_5b.cpp

**6.** Write a C program that gets the names of bicycles from two different sorted files named **bike1.txt** and **bike2.txt** into two arrays, and merges them and writes in a new file named **bike_list.txt.**

Write the following functions;
- **readBikeList** that takes the file pointer and the array which will keep the bike list as parameters, reads the bike names from the file into the specified array. The function should also return the number of bikes in the array.
- **shiftDown** that takes the bike list array, number of bikes and the position of the bike which will be shifted down as parameters. The function moves down all of the elements starting from specified position in the array.
- **merge** that takes two bike lists and their sizes as parameters, merges two array putting the bike names in the second array into the first array in a sorted form.

In main call the necessary functions.

**Project_name:** LabGuide4_6
**File_name:** Question_6.cpp

`Example Run:`

| **bike1.txt** |
|---|
| bianchi |
| bisan |
| colnago |
| gazelle |
| look |
| raleigh |
| trek |

| **bike2.txt** |
|---|
| beldesan |
| beldeyama |
| bisan |
| bmx |
| giant |
| kona |
| marin |
| pinokyo |
| polo |
| salcano |
| scott |

| **bike_list.txt** |
|---|
| beldesan |
| beldeyama |
| bianchi |
| bisan |
| bisan |
| bmx |
| colnago |
| gazelle |
| giant |
| kona |
| look |
| marin |
| pinokyo |
| polo |
| raleigh |
| salcano |
| scott |
| trek |

**Note that:** If we want to eliminate the **duplicate** bike names, then remember the lecture notes..