Department of Information Systems and Technologies

**CTIS151 – Introduction to Programming**

Fall 2024 - 2025

**Lab Guide #14 – Week 10 – 2**

**OBJECTIVES :** Parallel Arrays and operations, One-dimensional Arrays as input parameters

**Instructors :** Serpil TIN
**Assistants :** Berk ÖNDER, Efe Mert ŞAHİNKOÇ, Hatice Zehra YILMAZ

**Q1.** In a bag, there are several balls with 10 different colors. The colors of the balls are coded with integer numbers as follows; 0:White, 1:Black, 2:Red, 3:Blue, 4:Green, 5:Yellow, 6:Purple, 7:Orange, 8:Pink, 9:Gray.

Write a C program that gets the color codes of the balls in a bag from a file named **bag.txt**, and finds the total number of balls, the number of balls in each color, and the percentage of each color of balls to the total number of balls.

Write the folowing function;
- **display:** that gets the color code and displays the color name according to the color code as in the example run.

**Project Name:** LG14_Q1
**File Name:** Q1.cpp

**bag.txt**

```
1 0 2 3 6 9 7 4 9 2 3 9 5 0 9 4 6 9 7 1 6 9 2 9 8 3 8 4 9 5 6 0 2 6 3 6 9 3 6
```

**Example Run:**

```
There are 39 balls

COLOUR CODE NUMBER  PERCENTAGE TO TOTAL
White    0    3            7.69
Black    1    2            5.13
Red      2    4           10.26
Blue     3    5           12.82
Green    4    3            7.69
Yellow   5    2            5.13
Purple   6    7           17.95
Orange   7    2            5.13
Pink     8    2            5.13
Gray     9    9           23.08
```

**Q2.** Write a C program that reads the weekly weather forecast (day and night) from the file named **weather.txt** into two parallel arrays. Then the program will display the weekly forecast and the coldest day with the degree.

Write the following functions;
- **display:** that takes the arrays that keep the weekly day and night temperatures as parameters and displays the information on the screen.

- **findColdest:** that takes the day temperatures array as an input parameter, finds and returns the index of the coldest day.

**Example Run:**

```
DAY NIGHT
--- -----
 18    10
 22    15
 19    15
 11     8
 21    12
 19    16
 22    17

The coldest day is THURSDAY. The degree is: 11
```

**weather.txt**

```
18 10
22 15
19 15
11 8
21 12
19 16
22 17
```

**Project Name:** LG14_Q2
**File Name:** Q2.cpp

**Q3.**

**a)** A quiz is graded over 10, thus the quiz grades are numbers in the range of **1** to **10**. Given the quiz grades of some students within a file named **"grades.txt"**, count the number of students in each of the given grades and display a table that shows the frequency distribution of each grade, as shown in the example run below. Also, find and display the grade which has the maximum number of students. **HINT: Use an array to count the grades.**

Write the following functions;

- **`findMax`**: gets the array which keeps the frequency distribution as input parameter, computes and returns the index of grade which has the maximum number of student.
- **`display`**: gets the array which keeps the frequency distribution as input parameter and displays the frequency distribution as in the example run.

**Project Name:** LG14_Q3a
**File Name:** Q3a.cpp

**Example Run:**

```
 GRADE      FREQUENCY
 -----      ---------
     1              2
     2              5
     3              1
     4              5
     5              9
     6              4
     7              2
     8              3
     9              2
    10              2

 Most of the students(9 students) have got the grade 5
```

**grades.txt**

```
5 2 3 1 7 8 2 6 5 8 4 10 2 5 6 5 4 2 5 1 6 9 5 10 5 4 4 6 7 8 2 5 4 9 5
```

**b)** Modify the program **Q3a.cpp**, so the program displays the histogram of the frequency distribution of the grades.

A **Histogram** is a pictorial representation of a frequency array. Instead of printing the values of the elements to show the frequency of each number, we print a histogram in the form of a bar chart. For example, we may use asterisk (*) to build the bar. Each asterisk represents one data value occurrence (See the example run).

**Project Name:** LG14_Q3b
**File Name:** Q3b.cpp

**grades.txt**

```
5 2 3 1 7 8 2 6 5 8 4 10 2 5 6 5 4 2 5 1 6 9 5 10 5 4 4 6 7 8 2 5 4 9 5
```

**Example Run:**

```
GRADE      FREQUENCY HISTOGRAM
-----     ---------  ---------
    1             2  * *
    2             5  * * * * *
    3             1  *
    4             5  * * * * *
    5             9  * * * * * * * * *
    6             4  * * * *
    7             2  * *
    8             3  * * *
    9             2  * *
   10             2  * *
```

**Q4.**

**a)** W1rite a C program that reads several numbers from the **numbers.txt** file into an integer array with a maximum size of **20**. Then, it reads an index of a number, validates the index, and deletes the element from the array. The program displays the content of the array before and after the delete operation.

Write the following function;

- **`display`**: gets the integer array, and the number of elements as input parameters and displays the number of elements and the array content on the screen.

**Project Name:** LG14_Q4a
**File Name:** Q4a.cpp

**Example Run:**

```
Array Content BEFORE Delete operation:
There are 12 numbers in the array!

[ 0]   23
[ 1]   58
[ 2]   69
[ 3]    4
[ 4]   21
[ 5]    7
[ 6]   56
[ 7]   82
[ 8]   96
[ 9]   12
[10]   47
[11]   52
Enter the index of the element to be deleted(0-11):15
Enter the index of the element to be deleted(0-11):-5
Enter the index of the element to be deleted(0-11):20
Enter the index of the element to be deleted(0-11):6

The value 56 which is on the index 6 will be deleted!

Array Content AFTER Delete operation:
There are 11 numbers in the array!

[ 0]   23
[ 1]   58
[ 2]   69
[ 3]    4
[ 4]   21
[ 5]    7
[ 6]   82
[ 7]   96
[ 8]   12
[ 9]   47
[10]   52
```

**numbers.txt**

```
23 58 69 4 21 7 56 82 96 12 47 52
```

**b)** Modify the program **Q4a.cpp**, so the program will read the index and delete the element on the given index until an invalid index is given. The program displays the content of the array after each delete operation.

**Project Name:** LG14_Q4b
**File Name:** Q4b.cpp

**Example Run**:

```
Array Content BEFORE Delete operation:
There are 12 numbers in the array!

[ 0]   23
[ 1]   58
[ 2]   69
[ 3]    4
[ 4]   21
[ 5]    7
[ 6]   56
[ 7]   82
[ 8]   96
[ 9]   12
[10]   47
[11]   52
Enter the index of the element to be deleted(0-11):4

The value 21 which is on the index 4 will be deleted!

Array Content AFTER Delete operation:
There are 11 numbers in the array!

[ 0]   23
[ 1]   58
[ 2]   69
[ 3]    4
[ 4]    7
[ 5]   56
[ 6]   82
[ 7]   96
[ 8]   12
[ 9]   47
[10]   52
Enter the index of the element to be deleted(0-10):10

The value 52 which is on the index 10 will be deleted!

Array Content AFTER Delete operation:
There are 10 numbers in the array!

[ 0]   23
[ 1]   58
[ 2]   69
[ 3]    4
[ 4]    7
[ 5]   56
[ 6]   82
[ 7]   96
[ 8]   12
[ 9]   47
Enter the index of the element to be deleted(0-9):0

The value 23 which is on the index 0 will be deleted!

Array Content AFTER Delete operation:
There are 9 numbers in the array!

[ 0]   58
[ 1]   69
[ 2]    4
[ 3]    7
[ 4]   56
[ 5]   82
[ 6]   96
[ 7]   12
[ 8]   47
Enter the index of the element to be deleted(0-8):15
```

## ADDITIONAL QUESTIONS

**AQ1.**

Write a C program that reads the purchase and sale price for several products from the **sales.txt** file into two parallel arrays with a maximum size of **20**. Then displays a summary info (the purchase price, the sale price, and the profit for each product). The program should also display the total profit for the company.

**Project Name:** LG14_AQ1
**File Name:** AQ1.cpp

**Example Run:**

```
Purchase  Sale    Profit
--------  ------  ------
   19.90   24.50    4.60
  105.25  130.42   25.17
   42.89   40.25   -2.64
   54.79   45.36   -9.43
   85.90  100.90   15.00
   74.50   89.95   15.45
   15.90   22.90    7.00
   32.85   40.75    7.90
   19.95   24.25    4.30
   36.25   42.50    6.25
   12.90   15.85    2.95

Total profit is: 76.55
```

**sales.txt**

```
19.90 24.50
105.25 130.42
42.89 40. 25
54.79 45.36
85.90 100.90
74.50 89.95
15.90 22.90
32.85 40.75
19.95 24.25
36.25 42.50
12.90 15.85
```

**AQ2.**

Two positive integers are considered to be *relatively prime* if **there exists no integer greater than 1 that divides them both.** Write a C program that reads the pairs of numbers from an input file named **"nums1.txt"** or **"nums2.txt"** and stores them in two parallel arrays, checks whether the pairs are relatively prime or not, and displays the ones that are relatively prime on the screen. If there are no relatively prime numbers in the arrays display an appropriate message.

Write the following functions;

- **readFromFile:** takes the input file pointer and two integer arrays as parameters, reads the pairs of numbers from the file, and stores them in two parallel arrays. The function returns the actual number of elements in the arrays.
- **isRelPrime:** takes two numbers as its parameters, and returns 1 if the given numbers are relatively prime; otherwise returns 0.
- **findRelPrimes:** takes two parallel arrays and their size as input parameters, finds the relatively prime numbers, and stores their indexes in another integer array. The function returns the new array and the number of relatively primes.

**Project Name:** LG14_AQ2
**File Name:** AQ2.cpp

**Example Run(using nums1.txt):**

```
There are 3 relatively prime numbers in the arrays

    7    17
   13     7
   15     8
```

**nums1.txt**

```
9 3
7 17
24 15
13 7
8 18
15 8
```

**Example Run(using nums2.txt):**

```
There are no relatively prime numbers in the arrays!
```

**nums2.txt**

```
8 18
25 35
14 21
45 9
24 66
```