Department of Information Systems and Technologies

CTIS 152 – Algorithms and Data Structure

SPRING 2024 - 2025

Lab Guide #1 – Week 1-2

OBJECTIVE : General Review of 151 Subjects

Instructors: Serpil TIN
Assistants : Berk ÖNDER, Hatice Zehra YILMAZ

Q1. Write a C program that will take positive numbers as input parameters, then use these numbers to call the swap function using the "call by reference method". Your program must check for the given input's validity and then give an error message if the numbers are not positive.

Example Run1:

```
Enter first number: -9
First number must be positive.
Enter first number: -14
First number must be positive.
Enter first number: 2

Enter second number: -3
Second number must be positive.
Enter second number: 6

Before swap: a = 2, b = 6
After swap: a = 6, b = 2
```

Example Run2:

```
Enter first number: 5

Enter second number: -12
Second number must be positive.
Enter second number: 9

Before swap: a = 5, b = 9
After swap: a = 9, b = 5
```

Project Name: LG01_Q1
File Name: Q1.cpp

Q2. Write a C program that will generate ***n*** (will be given by the user) random numbers (between 2500- 7500), then the program writes each generated number and the digits of that number in reverse order into the file **reverse.txt**.

Example Run:

```
How many numbers will you generate: 6
6 numbers and their digits in reverse order were written to reverse.txt file
```

reverse.txt

```
Generated number   Digits in reverse order
****************   ************************
            7139    9    3    1    7
            2861    1    6    8    2
            4447    7    4    4    4
            4030    0    3    0    4
            4743    3    4    7    4
            5400    0    0    4    5
```

Project Name: LG01_Q2
File Name: Q2.cpp

GENERATION OF RANDOM NUMBERS:

1. Use stdlib.h (for srand function)
2. Use time.h (for time function).
3. srand(time(0)); for getting different number every time you run the program.
4. For getting a random number between 0 – 50: num = rand() % 51;
5. Apply debug process to check the random number.

Example program:

```
#include <stdio.h>
#include <stdlib.h> //for srand funtion
#include <time.h>  //for time function

int main(void)
{
	int num;

 /* we use srand function to be able to get a random number but we cannot use the srand function
on its own, so we also use time function in it to give a start point to the srand function;
because time is different every time you run the program, the random number will be different
also */

	srand(time(NULL));

 /* because time returns a very big number it returns the millisecond value of the hour, so we
want to get a random number between 0 and 99, we get the modulus 100 of the rand  function */
	num = rand() % 100;

   /* to create a number between a range*/
   //num = rand() % ((Max+1)-Min) + Min
	printf("The random number is: %d", num);
	return 0;
}
```

Example Run #1:
The random number is: 99

Example Run #2:
The random number is: 26

Q3. Write a C program that reads positive numbers, stores them in an integer array, and finds the count of prime and perfect numbers using the functions below.

Write the following function;

- **`fillArray:`** takes an array as a parameter, reads the positive numbers from the user, stores them in the array, and returns the actual number of elements.
- **`isPrime:`** takes a number as an input parameter and finds and returns whether the number is prime.
- **`isPerfect:`** takes a number as an input parameter and finds and returns whether the number is perfect.
- **`findPrimePerfect:`** takes an array of integers and the array's size as parameters, finds and returns the number of prime and perfect numbers in the given array.

Hints:

- A **Prime number** is a positive integer that is divisible by only 1 and the number itself.
- A **Perfect Number** is defined as a positive integer that is equal to the sum of its positive divisors, excluding the number itself.

Project Name: LG01_Q3
File Name: Q3.cpp

Example Run1:

```
Enter a positive number:  6
Enter a positive number:  5
Enter a positive number:  2
Enter a positive number:  65
Enter a positive number:  23
Enter a positive number:  22
Enter a positive number:  11
Enter a positive number:  -6

There are 4 prime numbers
There are 1 perfect numbers
```

Example Run2:

```
Enter a positive number:  44
Enter a positive number:  55
Enter a positive number:  1
Enter a positive number:  5
Enter a positive number:  7
Enter a positive number:  9
Enter a positive number:  21
Enter a positive number:  56
Enter a positive number:  -1

There are 2 prime numbers
There are 0 perfect numbers
```

Q4. Write a C program that reads IDs and game scores of several dart teams from the file "dart.txt"; finds and displays the average of each game and the average of each team using the functions below.

Write the following functions;

- **`readFromFile:`** takes a file pointer, a one-dim array to keep the team IDs, and a two-dimensional array to keep the game scores as a parameter. The function reads the team IDs into the one-dim array and the 4 game scores of several dart teams into the two-dim array from the specified file. The function also returns the number of teams.
- **`findTeamAvg:`** takes the two-dim scores array and the number of teams as input parameters finds the average of each team, and stores the averages into a one-dim array.
- **`findGameAvg:`** takes the two-dim scores array and the number of teams as input parameters finds the average of each game, and stores the averages into a one-dim array.
- **`displayGameAvg:`** takes the one-dim array which keeps the game averages as an input parameter and displays the averages of all games on the screen.

Example Run:

```
Team Number  Average
***********  *******
 12          483.50
 24          436.25
 33          505.25
 45          470.00
 57          517.50
 68          449.00
 79          444.25
 89          500.00
 96          484.00
 98          455.50

Game Number  Average
***********  *******
1            475.7
2            482.1
3            496.0
4            444.3
```

dart.txt

```
12 482 570 500 382
24 350 395 575 425
33 475 482 552 512
45 552 545 418 365
57 660 385 475 550
68 446 520 345 485
79 273 582 498 424
89 445 510 570 475
96 624 347 465 500
98 450 485 562 325
```

Project Name: LG01_Q4
File Name: Q4.cpp

ADDITIONAL QUESTIONS

AQ1. Write a program that reads numbers from a text file into a two-dimensional integer array. There are 5 columns in each row, while the number of rows is unknown. If the matrix is square, the program will calculate the sum of the elements on the minor diagonal, otherwise, it will calculate the product of elements in the given row.

Write the following functions;

- **`display`:** displays all elements in the matrix.
- **`sumOfRow:`** finds the sum of the elements in the specified row of the matrix.
- **`productOfMinor:`** finds the product of the elements on the minor diagonal of the matrix.

Project Name: LG01_AQ1
File Name: AQ1.cpp

input1.txt

```
2 52 4 7 8
3 36 95 47 48
26 12 25 41 85
15 36 45 73 5
48 7 11 98 12
79 35 64 72 19
36 44 21 36 15
45 77 5 25 4
14 4  6 78 23
78 36 12 3 98
25 89 7 12 54
95 26 36 85 74
42 78 69 42 1
```

input2.txt:

```
2 52 4 7 8
3 36 95 47 48
26 12 25 41 85
15 36 45 73 5
48 7 11 98 12
```

Example Run with input1.txt:

```
  2  52   4   7   8
  3  36  95  47  48
 26  12  25  41  85
 15  36  45  73   5
 48   7  11  98  12
 79  35  64  72  19
 36  44  21  36  15
 45  77   5  25   4
 14   4   6  78  23
 78  36  12   3  98
 25  89   7  12  54
 95  26  36  85  74
 42  78  69  42   1

The matrix is not a SQUARE matrix
Enter the ROW number: 1
The sum of the elements in the given ROW: 73
```

Example Run with input2.txt:

```
  2  52  4   7   8
  3  36 95  47  48
 26  12 25  41  85
 15  36 45  73   5
 48   7 11  98  12

The matrix is a SQUARE matrix
The product of the elements on the Minor Diagonal is:
    16243200
```

AQ2. Write a modular C program that reads the text file named "numbers.txt" and stores them in an array using a function named readFile(...) that takes in the file stream and the array as parameters to read the data from the text file into the array. Then, write a function named menu() that displays the menu as shown in the example run below to read, validate, and return the user's choice. Depending on this choice, perform the operation mentioned in the menu item selected, and write each menu item as a separate function.

numbers.txt

```
15 24  65  2  33  78  5  61  4  42  23  1  12  18  32  68  123  111  75
```

Project Name: LG01_AQ2
File Name: AQ2.cpp

Example Run:

```
DISPLAY
1. All numbers
2. Even numbers
3. Subscripts of odd numbers
4. The numbers with even subscripts
5. Minimum number
6. Subscript of maximum number
7. Exit
Enter your choice: 1

All numbers
************
15 24 65 2 33 78 5 61 4 42 23 1 12 18 32 68 123 111 75

DISPLAY
1. All numbers
2. Even numbers
3. Subscripts of odd numbers
4. The numbers with even subscripts
5. Minimum number
6. Subscript of maximum number
7. Exit
Enter your choice: 2
Even numbers
************
24 2 78 4 42 12 18 32 68

DISPLAY
1. All numbers
2. Even numbers
3. Subscripts of odd numbers
4. The numbers with even subscripts
5. Minimum number
6. Subscript of maximum number
7. Exit
Enter your choice: 3

Subscripts of odd numbers
************************
0 2 4 6 7 10 11 16 17 18

DISPLAY
1. All numbers
2. Even numbers
3. Subscripts of odd numbers
4. The numbers with even subscripts
5. Minimum number
6. Subscript of maximum number
7. Exit
Enter your choice: 4

The numbers with even subscripts
********************************
     15 65 33 5 4 23 12 32 123 75

DISPLAY
1. All numbers
2. Even numbers
3. Subscripts of odd numbers
4. The numbers with even subscripts
5. Minimum number
6. Subscript of maximum number
7. Exit
Enter your choice: 5

Minimum number
********************************
     1

DISPLAY
1. All numbers
2. Even numbers
3. Subscripts of odd numbers
4. The numbers with even subscripts
5. Minimum number
6. Subscript of maximum number
7. Exit
Enter your choice: 6

Minimum number
********************************
     16

DISPLAY
1. All numbers
2. Even numbers
3. Subscripts of odd numbers
4. The numbers with even subscripts
5. Minimum number
6. Subscript of maximum number
7. Exit
Enter your choice: 7
```