

EE103: Introduction to Programming

Laboratory-1

Problem 1 (70 pts)

Write a program that calculates the Celsius equivalent of the input Fahrenheit temperature. Ask the user the input temperature in Fahrenheit, then print the output in Celsius temperature. Use the given Formula to calculate the output temperature. (Be careful about using int, float, double type).

$$^{\circ}\text{C} = \frac{5}{9}(^{\circ}\text{F} - 32) \quad (0.1)$$

Problem 2 (50 pts)

The value for π can be determined by the series equation

$$\pi = 4 * (1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \frac{1}{11} + \frac{1}{13} - \dots) \quad (0.2)$$

Write a program to approximate the value of π using the Formula given including terms up to $\frac{1}{99}$ and $\frac{1}{11199}$. Compare and comment these two results. (Be careful about using int, float, double type).

EE103 – INTRODUCTION TO PROGRAMMING – LAB 3

ARRAYS

Hint: It can be useful to use "Debug" option of "Dev-C++" to control the array elements during the lab assignment. By using "Debug" option, you can see, if your program works properly in every step. If you need, you can find an explanation about debugging in the next page.

1. Write a C code which creates an array having 100 elements. The array has the consecutive integers from 0 to 99 as given below. Do not create the array by manually writing each of the elements.

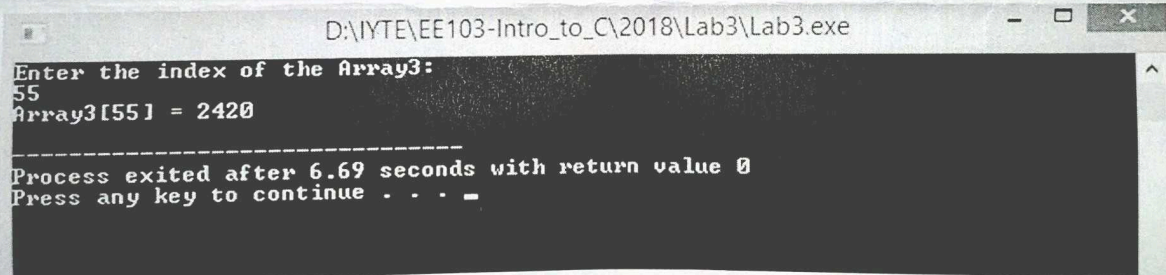
Array1 = {0, 1, 2, 3,, 97, 98, 99}.

2. Create another array by reversing the array you created above, which is given below. Do not create the array by manually writing each of the elements.

Array2 = {99, 98, 97,, 3, 2, 1, 0}.

3. Create a third array by multiplying each of the elements with the same index of Array1 and Array2. This will be your "Array3".

4. Ask the user to enter the index of the Array3. Then, print this array element as given below.



```
D:\IYTE\EE103-Intro_to_C\2018\Lab3\Lab3.exe
Enter the index of the Array3:
55
Array3[55] = 2420

-----
Process exited after 6.69 seconds with return value 0
Press any key to continue . . .
```

Lab Work- 4

Submission address: eelab204@gmail.com

A positive integer which is only divisible by 1 and itself is known as **prime number**. In this lab work,

1. First, our object is to check whether an integer (entered by the user) is a prime number or not. Write a **function** that determines if a number is prime. *Using function is obligatory.*

- (a) Your main program should ask the user for an integer input n.
- (b) The running program in the bash shell should have the following input & output:

```
$ Enter a positive integer number n:
$ 2
$ This is a prime number.
$ Enter a positive integer number n:
$ 21
$ This is not a prime number.
```

2. Use this function in a program that determines and prints all the prime numbers between 1 and 100.

- (a) Your main program should ask the user for integer input values 1 and 100.
- (b) The running program in the bash shell should have the following input & output:

```
$ Enter two defined integer numbers:
$ 1 100
$ Prime numbers between 1 and 100 are: 2 3 5 ..... 97
```

Lab Work - 5

The Simple Calculator

Submission Address: eelab204@gmail.com

L05...Student Number...Y18

14.11.2018

Purpose: You need to design a simple calculator with operation menu of 6 operations. In order to do that, you have to use the 'switch-case' structure. For each operation you also have to write functions. In the end there will be 5 functions. You **can't** use pow() function for the power part.

1. At first, design the menu of 6 operations as shown in the example output.
2. Ask the user to choose an operation number. Then you should ask the user to enter two numbers. According to the operation number realize the operation by calling functions using **switch-case** structure. Functions are:

- Addition, Subtraction, Multiplication, Division and Power (x^y).

Note that you need to ask user to choose operation until the 0 is entered. When it is entered, the process will be exited.

Example Output

```
$[0] Exit
$[1] Addition
$[2] Subtraction
$[3] Multiplication
$[4] Division
$[5] Power
$Enter Your Choice:3
$Enter the 1st number:
$40
$Enter the 2nd number:
$30
$1200
$[0] Exit
$[1] Addition
$[2] Subtraction
$[3] Multiplication
$[4] Division
$[5] Power
$Enter Your Choice:5
$Enter the 1st number:
$4
$Enter the 2nd number:
$5
$1024
$[0] Exit
$[1] Addition
$[2] Subtraction
$[3] Multiplication
$[4] Division
$[5] Power
$Enter Your Choice:0
$-----
```

Experiment-6

How Random is Your Random Sequence

(Duration: 90 mins)

In this experiment, you are going to examine the effect of changing the seed on the produced random numbers. To be able to do that, you need to compare histograms of random sequences that are generated by using different seeds.

Use the example program provided in your preliminary work by modifying the main function. Before you start, change N_C to 50 and N_R to 20. Press alt+enter to switch to fullscreen mode when running your program.

You need to generate 500 random integers between 0 and 49, count how many times an integer appeared and then save the number of counts to an array. For example if the number 3 was generated 10 times, the 3rd element of the array must be 10. Follow the instructions below to create your histograms;

1. Set the seed according to one of the cases given below
2. Generate 500 random integers between 0 and 49
3. Count the number of times each integer appears
4. Store the count in the corresponding element of an array
5. Draw the histogram of the array using the draw_histogram function provided in your preliminary work

Repeat the same process for the cases given below. Each case should show 2 graphs one under the other to compare the histograms for the given seeds. You should set the seed before generating each sequence. For example for the Case 1, you should set the seed to 0 and draw the first histogram, then set the seed to 0 again before drawing the second histogram. Use a switch-case statement and an infinite loop(i.e. while(1)) for building a user interface. The user should be able to select which case will be executed.

	Case 1	Case 2	Case 3	Case 4*
First Seed	0	0	time	time
Second Seed	0	15	time	time

*Case 4 is the same as Case 3 in terms of seed settings but the program needs to wait for a second to set the second time seed. Use the "sleep" function from unistd library before generating the second sequence.

```
//sleep function usage example
#include<unistd.h>
//your libraries

int main(){
    //your code
    sleep(1); //suspend this program for 1 second
    //your code
}
```

P.S. Keep in mind that a counter needs to be reset. Reset your counters to 0 before any time you want to count something.

Questions for You to Think About

1. Do you think you have finally achieved a real random sequence after setting your seed to time?
2. Your computers are very deterministic devices. How do you think that a deterministic device can actually generate random numbers? What do you call random? Do you think true randomness is achievable?

Experiment - 7

String Operations

Submission Address: eelab204@gmail.com
L07...Student Number...Y18
28.11.2018

Use the code below and write the functions whose prototypes are given.

```
#define SIZE 40
void printWord(char sentence[], int wordNum);
int findLength(char word[]);
void findWord(char sentence[], char word[], int wordLength);

int main() {
    char sentence[] = "EE103 Introduction to Programming";
    int wordNum;
    char word[SIZE];
    printf("Enter the word number:\n");
    scanf("%d",&wordNum);
    fflush(stdin); // Just to make gets() work properly
    printWord(sentence, wordNum);
    printf("\nEnter the word:\n");
    gets(word);
    printf("Word length is %d\n",findLength(word));
    findWord(sentence,word,findLength(word));
    system("pause");
    return 0;
}
```

- I. printWord() function should print the word of the sentence which is specified in the second argument(word number).(60 pt)
 - II. findLength() function should return the length of the word. (30 pt)
- Hint:** To find the length of the word, you may search for the string terminator location.
- III. findWord() function should print whether the word received from the user is a substring of the sentence or not. (40 pt)

Note: Do not use any <string.h> function

Example Outputs:

Enter the word number:
>>1
EE103
Enter the word:
>>Introduct
Word length is 9
Word is a part of sentence

Enter the word number:
>>3
to
Enter the word:
>>the
Word length is 3
Word is not a part of sentence

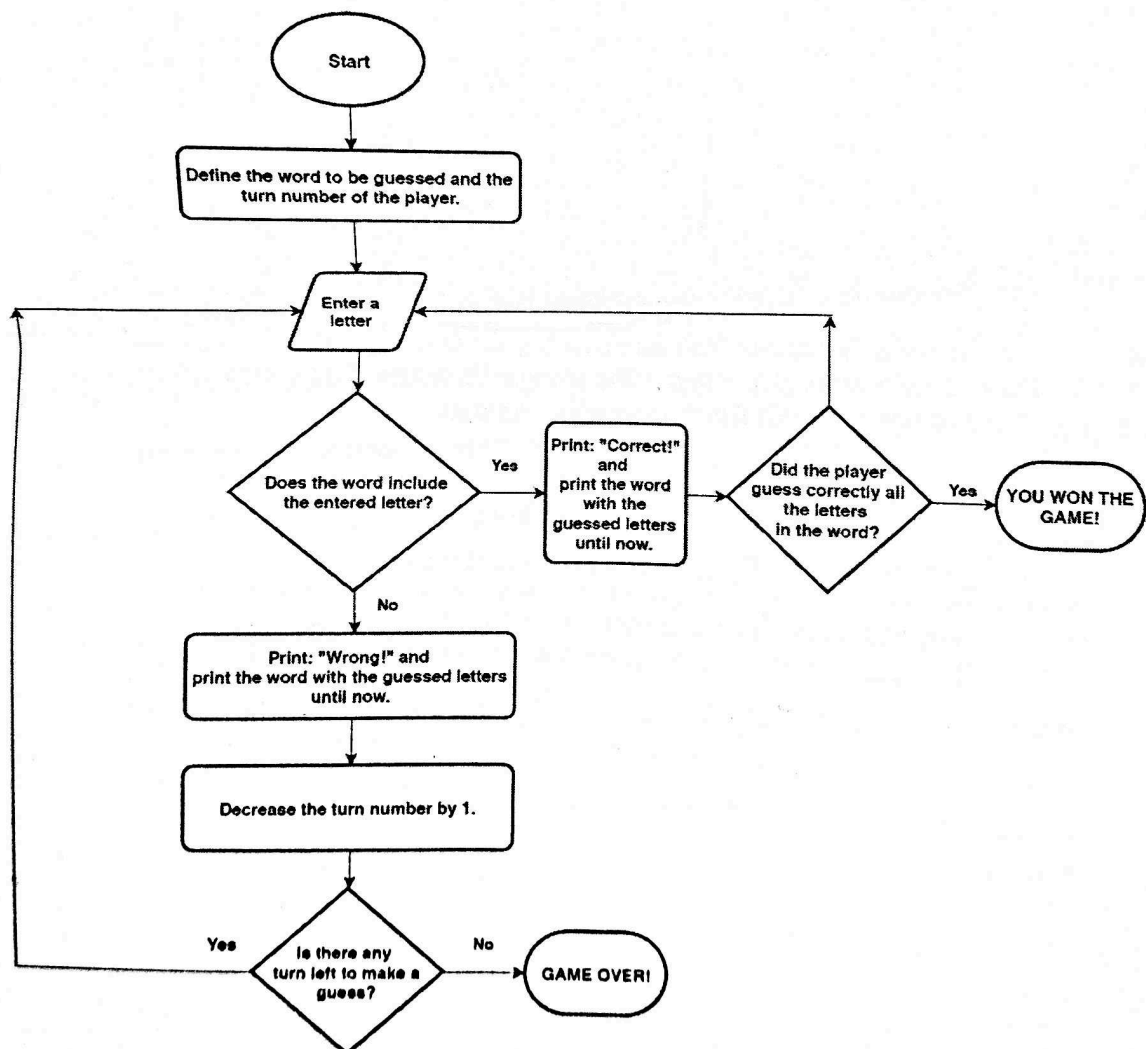
Enter the word number:
>>4
Programming
Enter the word:
>>EE103
Word length is 5
Word is a part of sentence

EE103 – INTRODUCTION TO PROGRAMMING – EXPERIMENT 8

YOUR FIRST GAME: HANGMAN

This week's experiment is for gamers 😊 You can use C programming language to develop games. In this experiment, you will write a C program that executes the "Hangman". In this game, the player tries to guess the word which is asked by the computer. As the game developer, you will define the word to be guessed in the main() part. Then, the player makes guesses by entering one letter in every step. At the beginning, you will define the player's turn number which shows how many WRONG guesses the player can make. If the player cannot guess the word before the turn number decreases to zero, then the game is over. If the player guesses the correct letters before the turn number decreases to zero, then the player wins the game. Assume that the player enters a different letter for every guess.

Implement the algorithm given below to develop the "Hangman" game.



TURN THE PAGE! →

The output of your game will be as below for winners and losers.

WINNER

```
You have 6 turns left. Enter a letter
E
_ _ E _ _
Correct!
You have 6 turns left. Enter a letter
D
_ _ E E D _ _
Correct!
You have 6 turns left. Enter a letter
P
_ _ E E D _ _
Wrong!
You have 5 turns left. Enter a letter
F
_ _ E E D _ _
Correct!
You have 5 turns left. Enter a letter
R
F R E E D _ _
Correct!
You have 5 turns left. Enter a letter
M
F R E E D _ M
Correct!
You have 5 turns left. Enter a letter
O
F R E E D O M
Correct!
You Won The Game :)

-----
Process exited after 22.06 seconds with return value 0
Press any key to continue . . .
```

LOSER

```
E
_ _ E E _ _
Correct!
You have 6 turns left. Enter a letter
P
_ _ E E _ _
Wrong!
You have 5 turns left. Enter a letter
T
_ _ E E _ _
Wrong!
You have 4 turns left. Enter a letter
W
_ _ E E _ _
Wrong!
You have 3 turns left. Enter a letter
Q
_ _ E E _ _
Wrong!
You have 2 turns left. Enter a letter
D
_ _ E E D _ _
Correct!
You have 2 turns left. Enter a letter
R
_ R E E D _ _
Correct!
You have 2 turns left. Enter a letter
Y
_ R E E D _ _
Wrong!
You have 1 turns left. Enter a letter
V
_ R E E D _ _
Wrong!
Game Over!!!

-----
Process exited after 196.7 seconds with return value 0
Press any key to continue . . .
```

Hint: Think as simple as possible. You can use any function in the compiler's libraries such as `<string.h>`. For example, `strlen()` function in the `string.h` library can be used to determine the length of a string. You can use the main function as given below.

```
int main()
{
    int i, size;
    char word[] = "FREEDOM"; // The word to be guessed by the player
    char g; // String that holds the letters guessed by the player
    size = strlen(word); // Determine the size of the string "word"
    char guess[] = "_____"; // Initialize the guessed string which will change for every guess

    int turn = 6; // The number that shows how many times the player can make a guess (turns)

    //****

    getch();
    return 0;
}
```

Experiment - 10

List Operations

Submission Address: eelab204@gmail.com
L10...Student Number...Y18
19.12.2018

Use the Python code below and write the functions which are used in the main.

```
from random import randint

'''
Write your functions here
'''

def main():
    random_list = create_random_list()
    print("Random list =\n" + str(random_list))
    print("Max element = " + str(find_max(random_list)))
    print("Sorted list =\n" + str(sort_array(random_list)))

if __name__ == '__main__':
    main()
```

- I. `create_random_list()` function should create and return a list, whose length is a random integer number between [4,9] and whose elements are also random integer numbers between [0,9] (40 pt)

Hint: Use `randint(min,max)` to produce random number and use built-in `append()` function to append elements to the list. (`a.append(b)` function adds element b to list a)

- II. `find_max(random_list)` function should return the maximum element in the list. (40 pt)

Hint: You may use `len(a)` function to get the length of the list a

- III. `sort_array(random_list)` function should return another list which is the sorted version of `random_list` in descending order (40 pt)

Note: In order to sort the array, use built-in `append()` and `remove()` functions and `find_max(random_list)` function that you created in the previous part. Find and remove the maximum element of the list iteratively and append to a new list (`a.remove(b)` removes the first element which is equal to b from the list a). Using another method for sorting will reduce the score you can get from the last part by 20 points.

Note: Do not use any built-in max or sort functions.

Example Outputs:

Random list =
[5, 2, 9, 2, 4, 5]
Max element = 9
Sorted list =
[9, 5, 5, 4, 2, 2]

Random list =
[8, 7, 9, 2, 0, 7, 8, 0, 7]
Max element = 9
Sorted list =
[9, 8, 8, 7, 7, 7, 2, 0, 0]

Random list =
[1, 1, 3, 2, 1]
Max element = 3
Sorted list =
[3, 2, 1, 1, 1]

—Experiment 11—

Create a code that asks user to enter a CHAR type array with 8 elements that contains numbers, letters and characters such as '&', ':', '+'. Then, your code has to extract the digits from the array and write these digits into a different array defined by INT type. You will create two-digit numbers by using the successive digits. After that, it sums all two-digit numbers and display the answer.

```
Please Enter the Characters of Array
k56l=4o4
You Entered : k56l=4o4
Total is = 100
```

Figure 1: $56+44=100$

However there are also some restrictive conditions:

- If the entered array contains capital letter, your program sums all two-digit numbers before the capital letter.

```
Please Enter the Characters of Array
k89K9817
You Entered : k89K9817
Total is = 89
```

Figure 2: Capital letter 'K'

- If the entered array contains zero ('0'), your program must ignore the zero and sums all other two-digit numbers.

```
Please Enter the Characters of Array
k9018098
You Entered : k9018098
Total is = 196
```

Figure 3: $98+98=196$