ITF22519 - Introduction to Operating Systems

Lab 3: Basic C Programming

Fall 2020

In this lab, you will do some practices with basic C programming in Linux. The practices include input, output, conditional statement, loop, function and arrays.

Before you start, remember to `commit` and `push` your previous lab to your git repository. Then, try to `pull` the new lab:

```
$ cd ITF22519/labs
$ git pull upstream master
$ cd lab3
```

Please remember these steps for the future labs.

1 Compiling a program in C into an executable file output

Suppose that you have your code in a single file *`example.c`* and that you want to run your code, type the following command in your terminal:

```
$ gcc example.c -o output
```

This command complies *`example.c`* into executable file output. Then,

```
$ ./output
```

to run the executable file output in the current directory. To get you started, we will be going over through creation, compilation, and execution of some simple examples.

2 Output

In C, function `printf` is used to produce output in the form of characters, string, float, integer that are printed on your terminal. This function is in the library *`stdio.h`*. Therefore, to use `printf`, you have to add the header file *`stdio.h`* in your code. This is done by preprocessor directive `include` as follows

```
# include <stdio.h>
```

To generate a newline, use "$\backslash n$" in printf() statement.

2.1 Task 1: Warming up with Greeting Program

This task requires you to print out *Hello, YourName* on the first line, and the string from the given input on the second line in your screen. Output format:

```
Hello, YourName!
Welcome to C programming.
```

Create the *welcome.c* file by using your preferred editors such as nano, emacs, vim, gedit, pico.

```
#include <stdio.h>
int main() {
     printf("Hello, YourName!\n"); //replace YourName with your name
     printf("Welcome to C programming.\n");
     return 0;
}
```

You can now compile your code by the following command

```
$ gcc welcome.c -o welcome
```

To run your program, type

```
$ ./welcome
```

`printf()` not only prints out the strings that a programmer writes in his/her own program but also variables. However, we need to add format specifiers to the string and the variables we want to print as arguments. Here is another example:

```
int num;
char ch;
printf("%d is an integer \n", num);
printf("%c is character \n", ch);
```

In C programming, %`d`, %`f`, %`s` and %`c` are for integer, float/double, string, and character types, respectively.

3 Input

Function `scanf` is used to get input from terminal. The function is also from *stdio.h*. The `scanf` function takes a format string followed by references to where the input should be stored. Remember & in front of variable. This means that the variable is passed as *reference* to `scanf`.

3.1 Task 1: Sum of two integers

The following is the code to calculate the summation of two integers input from terminal, however, it is not completed. Output format of the code is the following:

```
Enter one integer:
Enter another integer:
Summation of the two integers is:
```

Use any editor to make your own `.c` and complete the code

```
#include <stdio.h>
int main() {
       int A, B; // A and B to get input from terminal
       int Sum = 0; // Sum is used to store the sumation of A and B

       printf("Enter one interger:");
       scanf("%d", &A); // d is used for integer
       printf("Enter another integer:");
       scanf("%d", &B);

       // Add your own code here

       // End of your code
       printf("Sum is %d\n", Sum);
       return 0;
}
```

3.2 Task 2: Difference of two floats

Write a program which asks for two floats, prints out the input floats, and outputs their difference.

4 Conditionals

`if-else` statement is used to make a program behave differently depending on the program status or user input. The following code asks for two integers and prints out the their maximum value

```
#include <stdio.h>
#include <math.h>
int main() {
     int A, B;
     int Max;

     printf("Enter one interger:");
```

```
    scanf("%d", &A);
    printf("Enter another integer:");
    scanf("%d", &B);

    if (A < B)
        Max = B;
    else
        Max = A;
    printf("Max is %d ", Max);
}
```

4.1 Task 1

Save the above code in a `.c` file and run the code.

4.2 Task 2

Write the code that asks for two floats and prints out their minimum value.

4.3 Task 3

Write the code that asks for three integers and output whether any of them are equal. Use only one `if-else` statement.

5 Loops

Loops are used to execute a statement or a number of statements multiple times as long as the loop condition is satisfied. The following code prints out the numbers from 1 to 10 in one line and from 11 to 20 on another line using `while` loop:

```
#include<stdio.h>
#include<math.h>

int main() {
          int i = 1;
          while (i < 11) {
                printf("%d ", i);
                i++;
           }
          printf("\n");
          while (i < 21) {
                printf("%d ", i);
                i++;
          }
          printf("\n");
```

```
}
```

5.1 Task 1

Save the above code in a `.c` file and run the code.

5.2 Task 2

Do the same task with the example using for loop.

5.3 Task 3

Write a code to print all the numbers from 1 to 100 with 10 numbers on each line.

5.4 Task 4: Fibonacci series

A Fibonacci sequence is defined as follows: the first and second terms in the sequence are 0 and 1. Subsequent terms are found by adding the preceding two terms in the sequence. Write a C program to calculate the *nth* element of the Fibonacci series.

```
f(0) = 0;
f(1) = 1;
f(n) = f(n-1) + f(n-2) with n>1;
```

6 Functions

Functions are a great way to make your code reusable, improve the structure of the code, and isolate error. The following is a piece of code to calculate the summation of two integer using functions

```
#include <stdio.h>
#include <math.h>
int Calculate_Sum(int A, int B) {
       return (A + B);
}
int main() {
       int A, B;
       printf("Enter one interger:");
       scanf("%d", &A);
       printf("Enter another integer:");
       scanf("%d", &B);
       printf("Sum is %d\n", Calculate_Sum(A,B));
}
```

6.1 Task 1

Save the above code in a `.c` file and run the code.

6.2 Task 2

Write a program that takes two floats as argument and returns their product using function.

6.3 Task 3

Write a program that calculates the *nth* element of the Fibonacci series using function.

7 Arrays

In this exercise, we will look at some basic operations on arrays.

7.1 Task 1

Write a function that asks user to input 10 integers and stores them in an array. Then, print all elements of the array on your terminal.

7.2 Task 2

The following code read the content of file ***Array.txt*** which includes 10 integers and put these integers into an array A. Write your code to (a) print each element of A on the terminal, (b) calculate the summation of all elements in A, and (c) print the summation on the terminal.

```
#include<stdio.h>
#include <stdlib.h>
int main() {
         int i;
         int A[10];
         int Sum = 0;
         freopen("Array.txt", "r", stdin);

         for (i = 0; i < 10; i++) {
              scanf("%d", &A[i]);
              // your code here
              // print A[i] on the screen

              // Calculate the sum of the arrays

              // End of your code
        }
```

```
        // print the summary of the array on the screen
        fclose (stdout);
        return 0;
}
```

8 Exercises

8.1 Exercise 1 (30 points)

Write a program to calculate the summation of the first 100 integers. **Note:** 0 is an integer number.

8.2 Exercise 2 (30 points)

Write a program that reads all integer numbers of an input text file and print outs the number of zero-valued elements. If the file does not exit, print out an error reading message.

Hints: If you read a certain number of elements from the input text file or put them into an array with a fixed size, you will have problem when you do not know the size of the file or when the sizes of the file and array are not equal. There are several ways to avoid this problem. One of which is to use `fopen()` to open a file, `fscanf()` to read each integer of the file, and `fopen(feof())` to check if reading approaches the end of the file. For example:

```
FILE* File;
file = fopen("Array.txt", "r"); // if file Array.txt does not exit, file = NULL.
fscanf(file, "%d", &i); // read each element
                        // and put the element into an integer variable i
feof(file);             // check if reading approaches the end of file
```

Remember to close the file by using `fclose(file)`.

8.3 Exercise 3 (40 points)

Write a program to print the first n lines of the following series numbers. n is the input from terminal.

```
1
1    1
1    2    1
1    3    3    1
1    4    6    4    1
1    5    10   10   5    1
1    6    15   20   15   6    1
```

Sample output:

```
Enter number of lines(n) you want to print:
Printed lines:
1
1    1
1    2    1
1    3    3    1
1    4    6    4    1
1    5    10   10   5    1
1    6    15   20   15   6    1
1    7    21   35   35   21   7    1
```

(Suppose that the user input 8).

9 What To Submit

Complete all tasks under each section and submit your source files to GitHub. Complete the exercises in this lab, each exercise with its corresponding `.c` file and make your `lab3_report.txt` file. After that, put all of files into the **lab3** directory of your repository. Run `git add .` and `git status` to ensure the file has been added and commit the changes by running `git commit -m "Commit Message"`. Finally, submit your files to GitHub by running `git push`. Check the GitHub website to make sure all files have been submitted.