NWEN 241 Exercise 3

Linked Lists, File I/O and More

Release Date: **23 April 2024**

Submission Deadline: **01 May 2024, 23:59**

Objective:

The objective of this exercise is to write and debug programs using linked lists, file stream input/output, and command-line arguments.

At the end of this exercise, you should submit the required files to the Assessment System (`https://apps.ecs.vuw.ac.nz/submit/NWEN241/Exercise_3`) on or before the submission deadline. You may submit as many times as you like in order to improve your mark before the final deadline. Submissions beyond the deadline will not be marked and will receive 0 marks.

Exercise Requirements

For NWEN 241, it is highly recommended that you undertake all development using the computers in CO246. The computers in this lab use the Linux operating system. *This guide is written with the assumption that you are in CO246 lab.*

If you are not able to go the lab, you can remotely access similar computers via secure shell (ssh). Consult one of the remote study guides (see `https://ecs.wgtn.ac.nz/Courses/NWEN241_2024T1/RemoteStudyGuides`) and follow one that suits you the most.

Exercises

You may download a copy of the base source files used in the activities from `https://ecs.wgtn.ac.nz/foswiki/pub/Courses/NWEN241_2024T1/Exercises/nwen241_exercise3_files.zip`.

Activity 1: Linked List [30 Marks]

Copy and paste the following C program[1] to your favorite text editor:

```c
#include <stdio.h>

// (1) Complete linked list node declaration below.
//     Do not change the name of the structure.
struct node {

};

int main(void)
{
    struct node *head = NULL; // head points to the head of list
    struct node *tmp;

    // This for-loop will build a list
    for(int i = 0; i < 10; i++) {
        // (2) Use malloc() or calloc() to allocate memory for one node
        //     and let tmp point the the allocated memory.

        // (3) Assign 100*(i+1) to data and let next point to NULL.

        // (4) Append tmp to the list pointed by head
    }

    struct node *rover = head;
    while(rover) {
        printf("%d ", rover->data);
        rover = rover->next;
    }
    printf("\n");

    return 0;
}
```

[1] You can extract a copy of this file from `https://ecs.wgtn.ac.nz/foswiki/pub/Courses/NWEN241_2024T1/Exercises/nwen241_exercise3_files.zip`.

Save the file as `activity1.c`. Study the source file and do the following within the file:

1. Define a structure named `node` to represent a singly linked list node. The structure should have two parameters
   - `data`: an integer
   - `next`: pointer to next node in the list
2. Use either `malloc()` or `calloc()` to allocate memory for one node and let `tmp` point the the allocated memory.
3. Assign `100*(i+1)` to the `data` member of the node and let the `next` member point to `NULL`.
4. Append the newly created node pointed by `tmp` to the list pointed by `head`. (Append means insert after the current last element in the list.)

Your program should not output anything aside from the output generated by `printf()` in lines 26 and 29.

Compile and run the program. If you are happy with the program, submit it to the Assessment System for marking.

Activity 2: Text File Processing 2 [30 Marks]

In this activity, you will perform simple text file processing using file stream I/O. Suppose you are given a text file named `raw.txt` which contains 2 decimal numbers per line:

```
12 134
100 94
23 324
10 45
```

Copy and paste the following C program[2] to your favorite text editor:

```c
#include <stdio.h>

int main(void)
{
    int a, b;
    FILE *in; // use for handling input file
    FILE *out; // use for handling output file

    // Open raw.txt for reading

    // Open processed.txt for writing

    // Go thru raw.txt file and generate processed.txt file accordingly

    return 0;
}
```

Save the file as `activity2.c`. Use `activity2.c` as a starting point to write a program that will add the numbers in every line of `raw.txt` and output the numbers together with the sum to a file named `processed.txt`. The contents of `processed.txt` should look like this:

```
12 134 146
100 94 194
23 324 347
10 45 55
```

You can use the file `raw.txt` included in the archive `https://ecs.wgtn.ac.nz/foswiki/pub/Courses/NWEN241_2024T1/Exercises/nwen241_exercise3_files.zip` to test your code. If you are happy with the program, submit it to the Assessment System for marking.

[Hint: If you are using `fscanf()` to scan the contents of the file, you may want to pay close attention to its return value.]

[2] You can extract a copy of this file from `https://ecs.wgtn.ac.nz/foswiki/pub/Courses/NWEN241_2024T1/Exercises/nwen241_exercise3_files.zip`.

Activity 3: Command-Line Arguments [40 Marks]

In this activity, you will write a C program that accepts an arbitrary number of command line arguments. The program should treat every command line argument (excluding the program name itself) as an integer. The program should add all the integers and print the sum to the display. For instance, suppose you compile the program into an executable file named `activity3`, then if the program is executed as

```
./activity3 1 2 3 4 5 6
```

the output should be

```
21
```

If the program is executed as

```
./activity3
```

the output should be

```
0
```

Save the program as `activity3.c`. To have an easier time with the auto-marking script, make sure that the output only consists of the sum followed by a newline character.

If you are happy with the program, submit it to the Assessment System for marking.

[Hint: You may use `sscanf()` to convert a string to an integer.]