C Programming I
2022 Fall
Final

Instructor: Po-Wen Chi

Date: 2022.12.24 09:00 - 13:00

Policies:

- Online test.
- Do not forget to include your Makefile. TA will only use the command make to build your program. If make fails, you will get zero points and no room for bargaining. So if you do not know how to solve a problem, please, do not include it in your Makefile.
- I do not care your source code file names, but the executive binary names should be **fin01, fin02, fin03, fin04**.
- You can ask TA any questions on the moodle forum if you do not understand the problems. But remember, TA QA time is only 09:00-11:00.

1 Riemann Sum(25 pts)

You all know what Riemann Sum is, right? If not, I will worry about your Calculus status very much. I want you to implement a function to derive the Riemann Sum with a given range. Do not be panic, this is a programming class. So I will explain what you should do.

Given a polynomial $f(x)$ and partitions n, you can calculate the Riemann sum of $f(x)$ in the interval $[a, b]$ as follows.

$$\sum_{i=0}^{n-1} f(a + \frac{b-a}{n} \times i) \times \frac{b-a}{n}.$$

You need to prepare **riemann.h** and **riemann.c** for the following function:

```
// Input Arguments:
// n: n-th derivatives
// coefficients[], powers[]: polynomials
// size: the size (item numbers) of coefficients[] and powers[]
// a, b: from a to b
// n: n partitions
double riemann_sum( int32_t coefficients[], int32_t powers[], int32_t size,
                    double a, double b, int32_t n );
```

Please use **double** in this program. The precision is not a concern in this problem. Our TA will prepare **fin01.c** for you and I promise that it will include **riemann.h**. Do not forget to build **fin01.c** in your Makefile. All powers are greater than or equal to zero. However, the order of the power array is not guaranteed.

2 Coloring (25 pts)

Given a canvas, please implement a program to color a specific area on the canvas. First, our TA will prepare **canvas.h** for you and **you must include it in your code**. **canvas.h** have a 32-bit integer two-dimensional array called **canvas**, where each element is 0 or 1, and the array's row size and column size. There is an example here.

```
int32_t canvas[15][15] = { {1,0,0,0,0,0,0,0,0,0,0,0,0,0,1},
                           {0,1,0,0,0,0,0,0,0,0,0,0,0,1,0},
                           {0,0,1,0,0,0,0,0,0,0,0,0,1,0,0},
                           {0,0,0,1,0,0,0,0,0,0,0,1,0,0,0},
                           {0,0,0,0,1,0,0,0,0,0,1,0,0,0,0},
                           {0,0,0,0,0,1,0,0,0,1,0,0,0,0,0},
                           {0,0,0,0,0,0,1,0,1,0,0,0,0,0,0},
                           {0,0,0,0,0,0,0,1,0,0,0,0,0,0,0},
                           {0,0,0,0,0,0,1,0,1,0,0,0,0,0,0},
                           {0,0,0,0,0,1,0,0,0,1,0,0,0,0,0},
                           {0,0,0,0,1,0,0,0,0,0,1,0,0,0,0},
                           {0,0,0,1,0,0,0,0,0,0,0,1,0,0,0},
                           {0,0,1,0,0,0,0,0,0,0,0,0,1,0,0},
                           {0,1,0,0,0,0,0,0,0,0,0,0,0,1,0},
                           {1,0,0,0,0,0,0,0,0,0,0,0,0,0,1} };
int32_t row_size = 15;
int32_t col_size = 15;
```

Here we use element 1 as the boundary and you can see the canvas is separated into four pieces. Now you should make the user input a point on the canvas (x, y), which means the point at x-row and y-column, and a color. Then color the area where the point belongs to on the canvas. Your need to support five colors:

- Color option 0: white.
- Color option 1: red.
- Color option 2: yellow.
- Color option 3: green.
- Color option 4: blue.

The program should be executed as figure 1.

The user can continuously input its choices until Ctrl+C. Note that the boundary cannot be colored and pointed. You should print a warning.

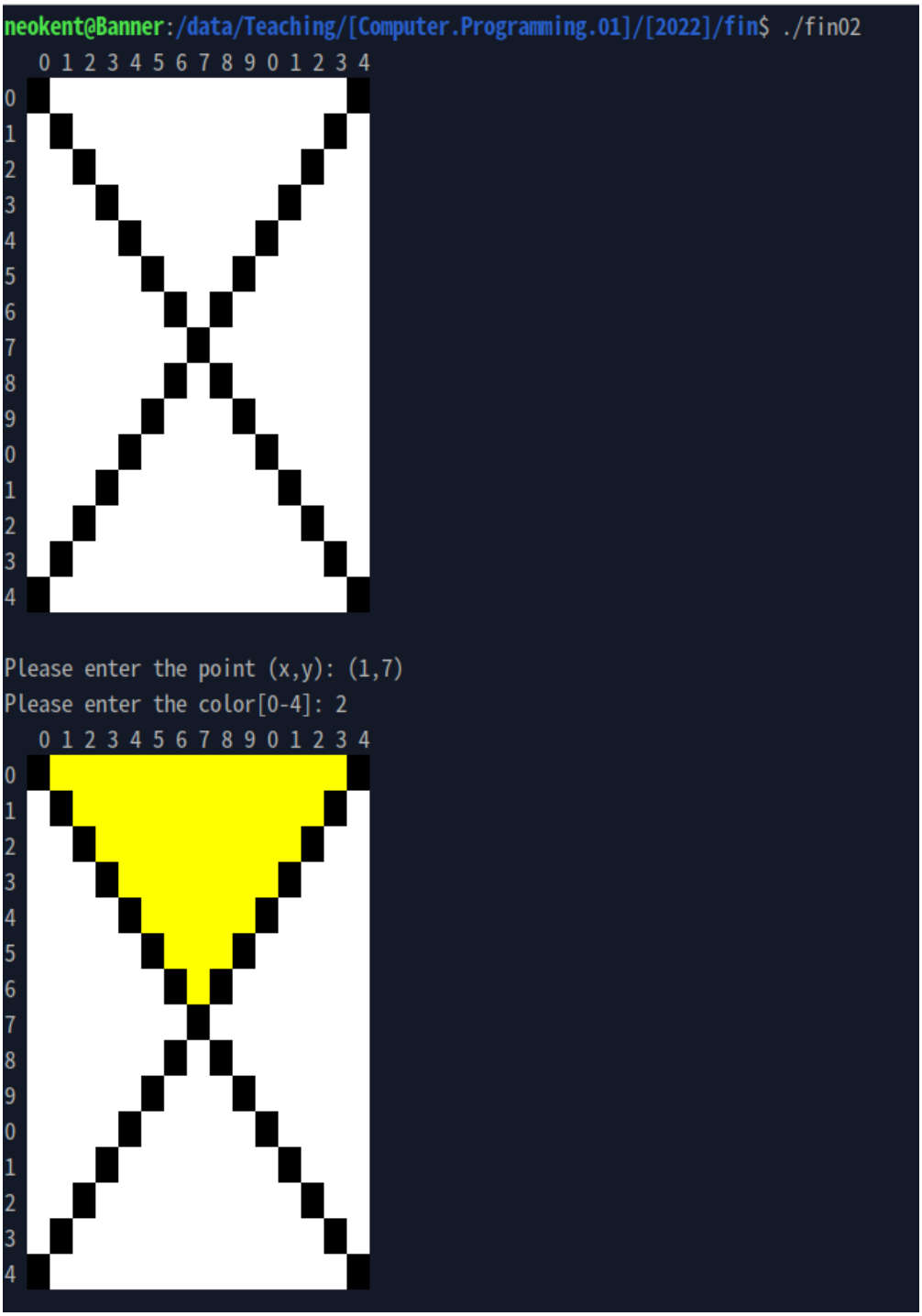

Figure 1: The example output of **fin02**. Boundary line is shown in black. You can see I use two spaces for each point.

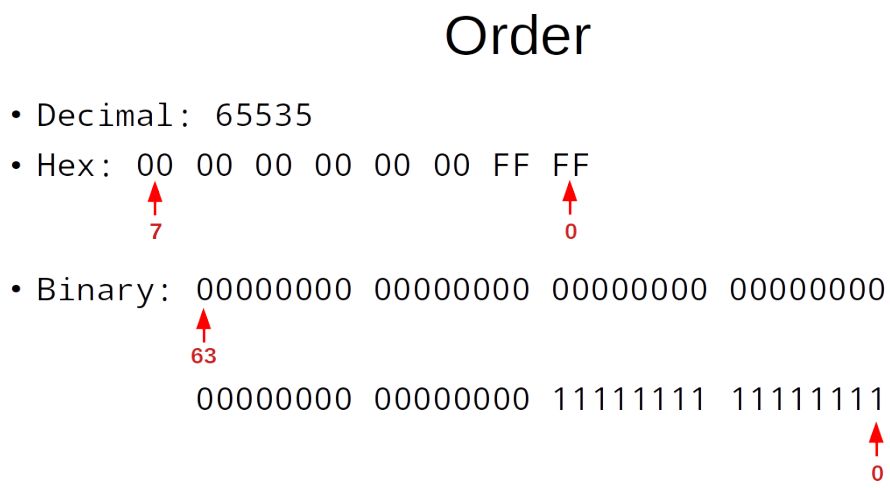

Figure 2: Bit and Byte Order.

3 64-bits Integer Editor (25 pts)

In this problem, I want you to develop a 64-bits integer editor. The program should support the following operations:

- Option 1: Switch two given bits. The user will provide two integers a and b which are in $[0, 63]$ and the program needs to switch a-th bit and b-th bit.
- Option 2: Switch two given bytes. The user will provide two integers a and b which are in $[0, 7]$ and the program needs to switch a-th byte and b-th byte.
- Option 3: Flip a given bit. The user will provide an integer a which is in $[0, 63]$ and the program needs to flip a-th bit.
- Option 4: Flip a given byte. The user will provide an integer a which is in $[0, 7]$ and the program needs to flip a-th byte.
- Option 5: Set a given bit. The user will provide two integers a, which is in $[0, 63]$, and b, which is in $[0, 1]$, the program needs to set a-th bit to b.
- Option 6: Set a given byte. The user will provide two integers a, which is in $[0, 7]$, and b, which is in $[0, 255]$, the program needs to set a-th byte to b.
- Option 7: Exit.

The bit and byte orders are shown in 2.

The operational example is as follows. First, the user needs to input an integer. Then, for each round, the user inputs its operational option, followed by arguments if necessary.

```
$ ./fin03
Please enter a 64-bit integer: 123456789123456789
--> Decimal: 123456789123456789
--> Hex: 01B69B4BACD05F15
```

```
--> Binary: 00000001 10110110 10011011 01001011 10101100 11010000 01011111
    00010101
Operation: 1 0 1
--> Decimal: 123456789123456790
--> Hex: 01B69B4BACD05F16
--> Binary: 00000001 10110110 10011011 01001011 10101100 11010000 01011111
    00010110
Operation: 2 0 1
--> Decimal: 123456789123438175
--> Hex: 01B69B4BACD0165F
--> Binary: 00000001 10110110 10011011 01001011 10101100 11010000 00010110
    01011111
Operation: 3 0
--> Decimal: 123456789123438174
--> Hex: 01B69B4BACD0165E
--> Binary: 00000001 10110110 10011011 01001011 10101100 11010000 00010110
    01011110
Operation: 4 0
--> Decimal: 123456789123438241
--> Hex: 01B69B4BACD016A1
--> Binary: 00000001 10110110 10011011 01001011 10101100 11010000 00010110
    10100001
Operation: 5 0 0
--> Decimal: 123456789123438240
--> Hex: 01B69B4BACD016A0
--> Binary: 00000001 10110110 10011011 01001011 10101100 11010000 00010110
    10100000
Operation: 6 0 161
--> Decimal: 123456789123438241
--> Hex: 01B69B4BACD016A1
--> Binary: 00000001 10110110 10011011 01001011 10101100 11010000 00010110
    10100001
Operation: 7
```

For any invalid inputs, make the user re-input again.

4 My Memory Management Functions(25 pts)

In this class, I have taught you how to manage memories in C. Sometimes, we want more additional features for memory management. For example, if there is someone who modifies data in my memory space, is it possible to alert the user? So, in this problem, I want you to implement some additional features in memory management functions.

First, when a user request a memory space, you need to prepare additional space before and after the user data, as shown in Fig. 3.

- The first four bytes must be **0x4E, 0x54, 0x4E, 0x55** since this is a proprietary memory allocation.
- Then, you should use four bytes to record the user requested memory size.

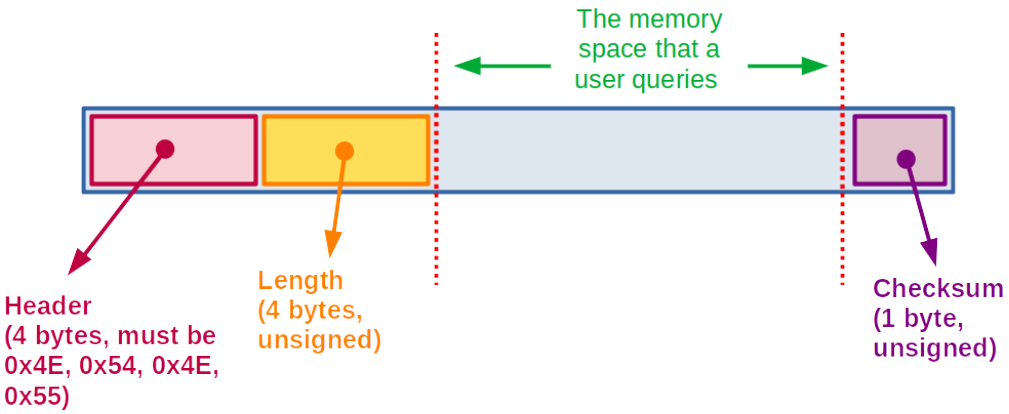

Figure 3: Memory layout.

- You should append one additional byte at the end of the requested memory space, which is called **checksum**. Here we use a simple checksum mechanism, where **the sum of all user data bytes and the checksum byte must be 0x00**.

You need to implement the following functions.

```
// Memory allocation with initializing the allocated data space to zero.
// Return -1 if allocation fails; otherwise return 0.
int32_t mymem_alloc( uint8_t **ptr, size_t size );

// Set value to ptr[i].
// Return -1 if the input pointer is NULL or the memory space is invalid or idx is invalid;
// Otherwise return 0.
int32_t mymem_set( uint8_t *ptr, size_t idx, uint8_t value );

// Copy n bytes from array to *(ptr+start).
// Return -1 if the input pointer is NULL or the memory space is invalid or the input is invalid (out-of-bound);
// Otherwise return 0.
// For simplicity, I promise that n is definitely valid for array.
int32_t mymem_set_array( uint8_t *ptr, size_t start, uint8_t array[], size_t n );

// Allocate a new memory space according to new_size and copy data from old space to the new space.
// If the new space is smaller than the old one, only copy new_size bytes.
// Return -1 if the input pointer is NULL or the memory space is invalid or allocation fails;
// Otherwise return 0.
int32_t mymem_rellocate( uint8_t **ptr, size_t new_size );

// Free.
// Do not forget to check if ptr is NULL.
void mymem_free( uint8_t *ptr );
```

A simple example is in Fig. 4.

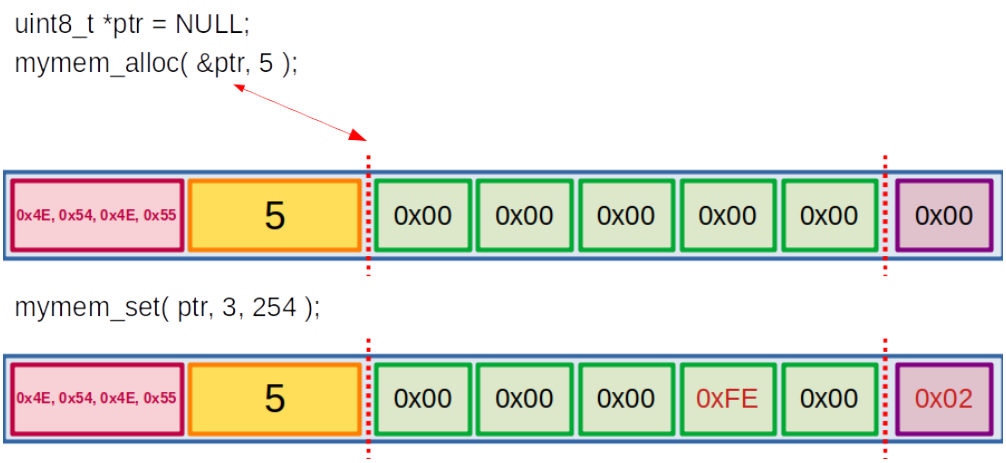

Figure 4: Example.

The checksum is 0x02 because

$$0x00+0x00+0x00+0xFE+0x00+0x02=0x00.$$

You need to prepare **mem.h** and **mem.c**. Our TA will prepare **fin04.c** for you and I promise that it will include **mem.h**. Do not forget to build **fin04.c** in your Makefile.

5 Bonus: Your Comments (5 pts)

Again, any comments are welcomed. However, you will get nothing if you leave this question blank.

6 Bonus: Questionnaire (5 pts)

Your upperclassman, Tsai Fong Chun, who is in an exchange at Uppsala university, currently does research on computer science education. He asked me for help to collect questionnaires from students in this class. If you submit this questionnaire, you will be rewarded additional 5 points.

Questionnaire: `https://forms.office.com/e/ki3DpuRr8j`

Some notes:

- So please focus on the exam first and you can complete this form after due time. You only need to submit this form before Next Tuesday.
- **You must answer the last question**, your ID and your name, if you want to be rewarded.

```
1 紀老師您好
```

```
我是資工112的蔡鳳駿，目前在烏普薩拉交換，修習一堂關於資訊工程教育的課程，目前與其他兩位研究生在研究「學生對於新科技應用在資訊教育上的期望」，哪些工具或是課程的哪部分是對學生最具吸引力的。我們想透過表單的形式調查學生，能否請老師將連結發給「程式設計」的學生們並填寫表單呢？

Dear professor,
I am Tsai Fong Chun from department of computer science and information technology, and currently doing an exchange at Uppsala university. I am studying a course that does research on computer science education. My classmates and I am doing a research about students expectation of new technology being used in teaching programming. For example, which tools or which part of the course attracts student the most. We would like to investigate using google form. Therefore, we are wondering if you can help us to send the link to the students studying in computer programming.
```

Advertisement: Software Engineering Final Presentation

Welcome to join the final presentation of the class, Software Engineering.

- Date: 2022.12.27 AM 9:00
- Location: Applied Science Building B1 Lecture Room