C Programming I
2023 Fall
Midterm

Instructor: Po-Wen Chi

Date: 2022.10.28 PM 2:00-6:00

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 **mid01, mid02, mid03, mid04, mid05**.
- You can ask TA if you do not understand the problems.

1 Time Duration (20 pts)

Please develop a program to calculate the time duration, where the time unit is **second**. The format is Year-Month-Day Hour-Minute-Second. Hour is displayed in 24-hour clock. I promise that the answer can be put in a 64-bits unsigned integer. You may worry about the **Leap year （閏年）** issue. For your simplicity, you can follow the rule shown in Fig. 1.

```
$ ./mid01
Start Time: 2023-1-23 13:10:00
End Time:   2023-1-23 13:10:01
Duration:   1 sec
```

2 Colorful Diamond (20 pts)

Figure 2 is a very famous programming example when teaching loops. I believe that this problem is just a piece of cake for all of you. Do you think that I will use this as one of your midterm problems? Come on! You are NTNU CSIE students and you are professional!! I need to enhance this problem because you all deserve a better programming problem.

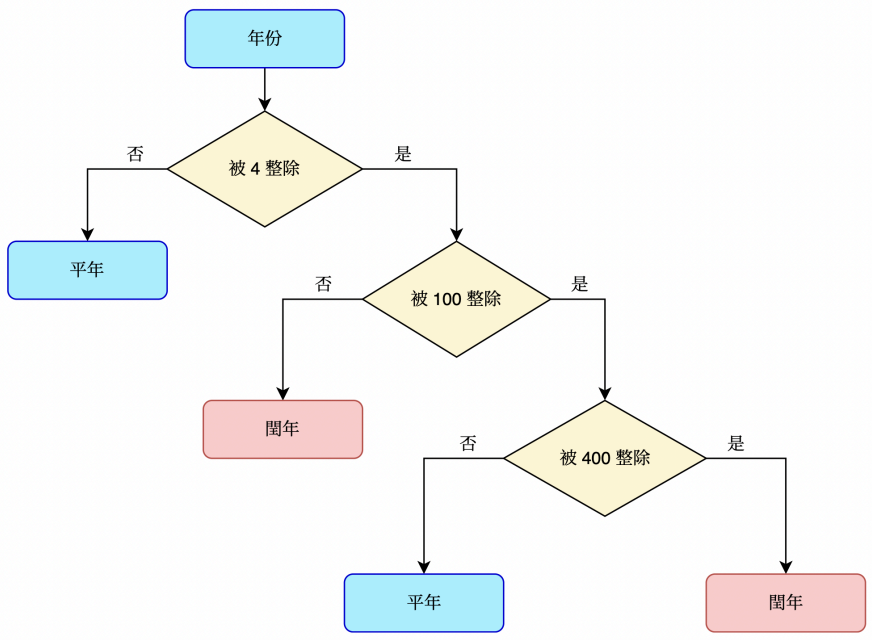


Figure 1: Leap year rule.

Please see figure 3 and this is what you need to do. A user should input the diamond size n and the block edge length l. Note that for the beauty issue, you should use $2 \times l$ as the block horizontal length. The colors from exterior to interior are RED → GREEN → BLUE → RED → GREEN → BLUE → ....

3 Treasure's Password (20 pts)

TL;DR, The Story Behind

During an expedition, the ancient elf and wizard, Frieren, discovered a magical treasure chest. Tempted by its mystical allure, she wanted to open it. Yet, the chest is protected by a powerful spells, and one wrong move would eat Frieren immediately.

As a budding magician in Magical Engineering, you're tasked with developing the magic to safely unlock it.

The chest has 10 buttons marked with 0 to 9, and the following rules to unlock: (spells can be larger than 18446744073709551615)

- Rule1: The spells must contain a pattern which has three set numbers {1, 32}, {65}, {9, 78} and three sets must in order.
  - legal
    - 1659
    - 326578
    - 1653265978
  - illegal

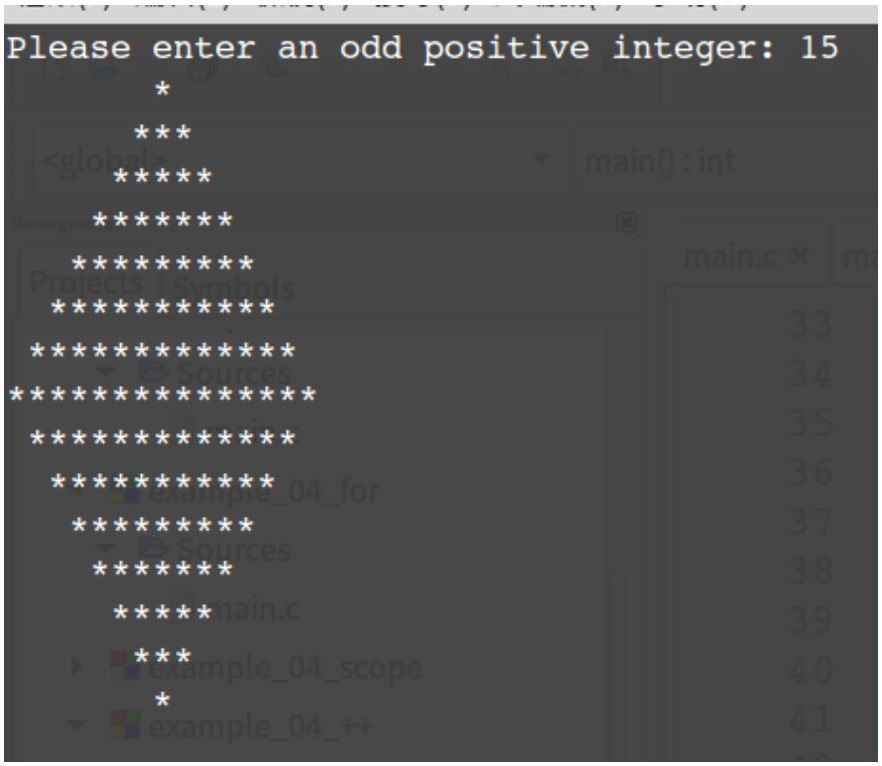
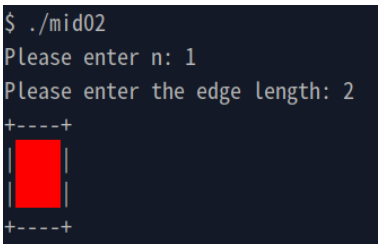


Figure 2: A famous programming example for loops.

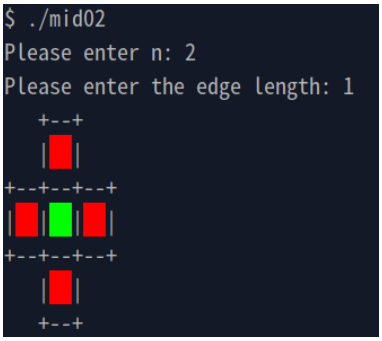
    - 165
    - 786532

- Rule2: The spells must contain a pattern which has "one or two 7s" and "zero or one 5" and "one 3" in order.
    - legal
        - 73
        - 753
        - 77775333
        - 707303
    - illegal
        - 53
        - 77553

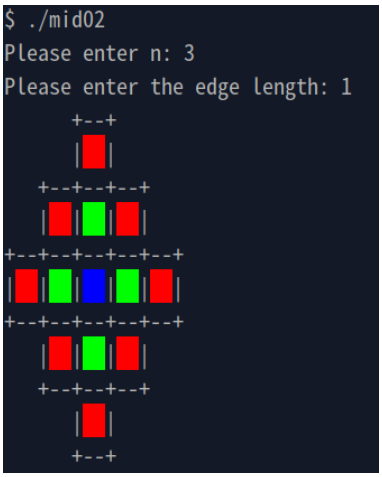
- Rule3: The spells must contain a pattern which has 4 and 8, having one or more "666" between 4 and 8, and 4 must before 8.
    - legal
        - 46668
        - 466668



(a) $n = 1, l = 2$.



(b) $n = 2, l = 1$



(c) $n = 3, l = 1$

Figure 3: Colorful Diamonds.

- illegal
  - 4668
  - 466068

So, such as 16597346668 can be a spells.

The spells will be entered digit by digit until the number "-1" is entered, which means the end of the input. Besides, Frieren is smart wizard, so she wouldn't not input invalidly, means the input will only within 0 9. If the input could pass these rules, please output the message "SUCCESS!". Otherwise, please tell Frieren which rule(s) does not.

Hint: HW2

```
$./mid01
Please input the digit: 1
Please input the digit: 6
Please input the digit: 5
Please input the digit: 9
Please input the digit: 7
Please input the digit: 3
Please input the digit: 4
Please input the digit: 6
Please input the digit: 6
Please input the digit: 6
Please input the digit: 8
Please input the digit: -1
SUCCESS!
```

```
$./mid01
Please input the digit: 4
Please input the digit: 6
Please input the digit: 6
Please input the digit: 6
Please input the digit: 8
```

Figure 4: An Illustration by DALL • E

Table 1: Odd Parity Bit Example

Original Data	(Count of 1-bits)	Additional Bit	Final Data
00000000	0	1	00000000**1**
01010001	3	0	01010001**0**
01101001	4	1	01101001**1**
01111111	7	0	01111111**0**

```
Please input the digit: -1
Rule1 Rule2 not follow!
```

```
$./mid01
Please input the digit: 1
Please input the digit: 6
Please input the digit: 5
Please input the digit: 9
Please input the digit: 7
Please input the digit: 3
Please input the digit: -1
Rule3 not follow!
```

4 Parity Bits (20 pts)

A parity bit, or check bit, is a bit added to a string of binary code. Parity bit is a simple form of error detecting code. I will show you how it works. The parity bit ensures that the total number of 1-bits in data is **odd**[1]. Therefore, this approach append one additional bit to the data to pass the parity check. If the count of 1-bits is even, the appended bit will be 1; otherwise, the appended bit will be 0. Table 1 is an example. Parity check is a common way to detect data errors.

You can see the following link for reference.

https://en.wikipedia.org/wiki/Parity_bit

This time, I want you to implement a function for 2-dimensional parity generation. Please see Fig. 5 for reference. You need to develop a function for given 5 32-bits **signed** integer and return a 64-bits **unsigned** integer, like 369 in Fig. 5. Since we only use the last 33 bits in a 64-bits integer[2], all unrelated bits are set to zero. The function should be as follows. Note these five inputs are not appended with the parity bits.

```
/* Example
 00000000 00000000 00000000 00000001 | 0
 00000000 00000000 00000000 00000001 | 0
 00000000 00000000 00000000 00000001 | 0
 00000000 00000000 00000000 00000001 | 0
 00000000 00000000 00000000 00000001 | 0
---------
```

[1]This is called odd parity check. Of course, even parity is fine. In this problem, we just use odd parity check.

[2]32 bits for input data and one appended parity bit.

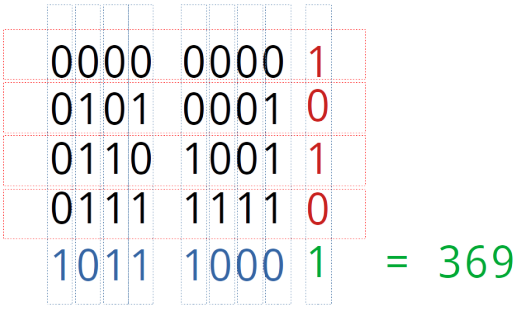


Figure 5: Two Dimensional Parity Check.

```
 11111111 11111111 11111111 11111110 | 1
 So  2dparity( 1, 1, 1, 1, 1 ) will return 8589934589
*/
uint64_t 2dparity( int32_t, int32_t, int32_t, int32_t, int32_t );
```

You should also prepare a header file called parity.h. Our TAs will prepare mid03.c which includes parity.h and uses these functions. **Do not forget to make mid03.c to mid03 in your Makefile.**

5 Gymnastic Formation 疊羅漢 (20 pts)

Gymnastic formation is a collective art based on gymnastics using only human bodies without instruments. As you can guess, the member on the bottom center needs to afford the most weights. Please develop a function to calculate the weight a member should afford.

Please see Fig. 6 for reference. The formation is composed of boys and girls. For your simplicity, each girl has the same weight and so does each boy. Undoubtedly, boys and girls have different weights. The top of the formation starts from a girl. She does not need to bear any weight since there is no one on her shoulder. Her weight will be equally shared by the two boys. So the boys at (1,0) and (1,1) should bear $\frac{g}{2}$ kg, where g is the girl's weight. Then, the girls at (2,0) and (2,1) will equally share $\frac{g}{2} + b$, where b is the boy's weight. Note the girl at (2,1) also needs to bear the weight from (1,1). Now you need to develop functions to implement the weight calculator.

```
// Setup weights for boys and girls
void setup_girl_weight( uint32_t );
void setup_girl_weight( uint32_t );

// Calculate the weight that the member at (x,y) should bear.
// If there is any error inputs or weights are not initialized, return -1.
int64_t afford_weight( int32_t x, int32_t y );
```

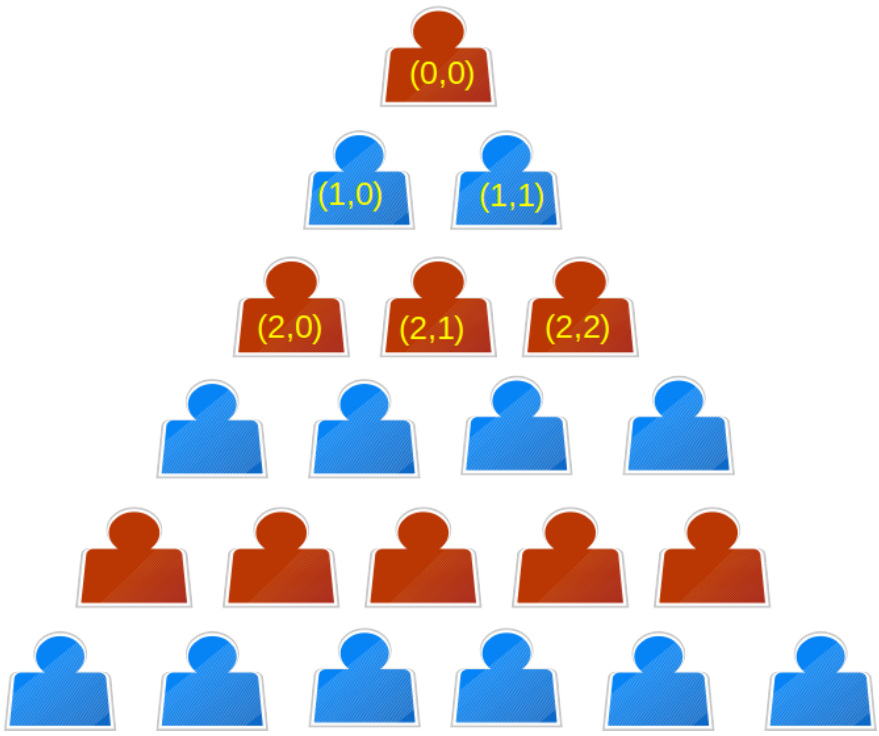


Figure 6: Example.

You should also prepare a header file called weight.h. Our TAs will prepare mid04.c which includes weight.h and uses these functions. **Do not forget to make mid04.c to mid04 in your Makefile.**

6 Bonus: Your Comments (5 pts)

Your comments about this class. Any comments are welcomed. Do not worry about typos or any grammar errors. However, you will get nothing if you leave this question blank.