National Taiwan Normal University
CSIE Computer Programming I

Instructor: Po-Wen Chi
Due Date: 2023.10.17 PM 11:59

Assignment 2

Policies:

- Zero tolerance for late submission.
- Please pack all your submissions in one zip file. **RAR is not allowed!!**
- For convenience, your executable programs must be named following the rule hwXXYY, where the red part is the homework number and the blue part is the problem number. For example, hw0102 is the executable program for homework #1 problem 2.
- I only accept **PDF** or **TEXT**. MS Word is not allowed.
- Do not forget your Makefile. For convenience, each assignment needs only one Makefile.
- Please provide a README file. The README file should have at least the following information:
  - Your student ID and your name.
  - How to build your code.
  - How to execute your built programs.
  - Anything that you want to notify our TAs.
- **DO NOT BE A COPYCAT!!** You will get ZERO if you are caught.

2.1 $\sqrt{2}$ (20 pts)

The square root of 2 ($\sqrt{2}$) is a positive real number that, when multiplied by itself, equals the number 2. It was probably the first number known to be irrational. So how to derive the value of $\sqrt{2}$? I will show you an approach called **Continued fraction**.

$$\sqrt{2} = 1 + \cfrac{1}{2 + \cfrac{1}{2 + \cfrac{1}{2 + \ddots}}}$$

The sequence will be $1, \frac{3}{2}, \frac{7}{5}, \frac{17}{12}, \ldots$. In mathematics, this sequence will finally converge to a constant. Is this true in programming? Please develop a program to calculate the value of $\sqrt{2}$ from 1 to n. For your simplicity, I promise that n is a 16-bits unsigned integer. You should also calculate the difference between the value and **1.41421356237309504880**[1].

```
$ ./hw0201
Please enter n (16-bits unsigned): 2
n = 1: 1.00000000000000000000 (-0.41421356237309504880)
n = 2: 1.50000000000000000000 (0.085786438............) // 我懶的算了
```

Note that you should use **double** instead of float. The precision should be 20.

2.2 Multiplication v2 (20 pts)

Please write a program for a user to input two **unsigned 32-bits integers** and print the multiplication process.

```
$ ./hw0202
Please enter the first  number: 34
Please enter the second number: 12
    3 4
*)  1 2
-------
    6 8
  3 4
-------
  4 0 8
$ ./hw0202
Please enter the first  number: 5
Please enter the second number: 24
      5
*)  2 4
-------
    2 0
  1 0
-------
  1 2 0
$ ./hw0203
Please enter the first  number: 123
Please enter the second number: 5
  1 2 3
*)    5
-------
  6 1 5
$ ./hw0204
Please enter the first  number: 1234
```

[1]This is a constant for $\sqrt{2}$ defined in standard C

```
Please enter the second number: 12
    1 2 3 4
*)      1 2
-----------
    2 4 6 8
  1 2 3 4
-----------
  1 4 8 0 8
```

If there is any invalid input, print an error message and terminate your program. For your simplicity, I promise that all inputs are integers and can be put in a 32-bits memory space.

2.3 Non-deterministic Finite State Machine (20 pts)

What is Non-deterministic Finite State Machine? Well ... there is an important class called **Automata** taught by Professor Wang. I suggest you to take this course. So I will not introduce what it is but simply describe how it works. Figure 2.1 is an non-deterministic finite state machine example. The circle implies the state. Initially you are in $S0$. When receiving a number, you will move to the next state according to the conditions shown on the directed line. Note that one input may cause two different forwarding states[2].

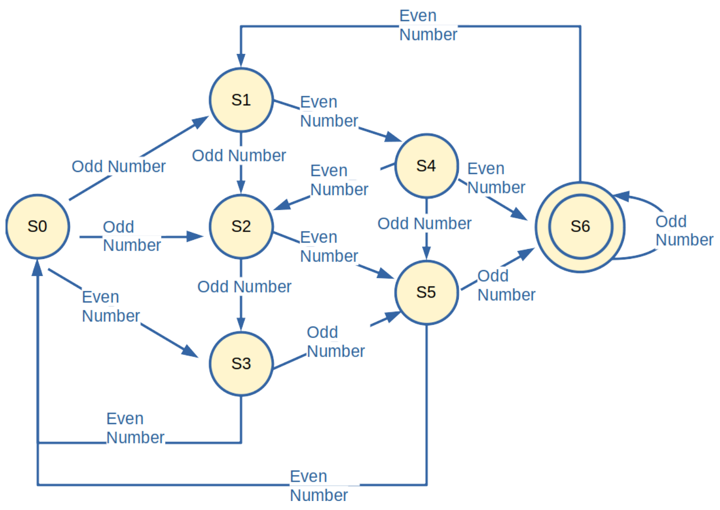

FIGURE 2.1: Non-deterministic Finite Automata.

Now you need to develop a program to simulate the state transition in Fig 2.1. Here we treat **0** as a special input which means the end of

[2]This is the reason why we call it non-deterministic.

input. You need to show all possible states with the given inputs. For your convenience, all inputs are promised to be 32-bits integers.

```
$ ./hw0203
Please enter an integer: 13
Please enter an integer: 9
Please enter an integer: 0
Possible States: S2, S3
```

2.4 Dollar Cost Averaging Calculator (20 pts)

Dollar cost averaging is a strategy to manage price risk when you're buying stocks, exchange-traded funds (ETFs) or mutual funds. Instead of purchasing shares at a single price point, with dollar cost averaging you buy in smaller amounts at regular intervals. For example, you may spend 4000 dollars per month on stock. Today, I want you to develop a dollar cost averaging calculator. You can see the following link for reference.

https://havocfuture.tw/compound-interest-calculator

```
$ ./hw0204
Initial Investment:            50000
Recurring Monthly Investment:  4000
Start Month:                   12
Start Year:                    2023
End Month:                     8
End Year:                      2043
Annual Rate of Return (%):     12
--- Output ---
2023.12) 50000/50000/0/0%
2024.01) 54000/54500/500/0.92%
...
```

Note that the input is **annual rate** but you should use **monthly rate** for compound interests.

2.5 Catan Simple Islands Engine (20 pts)

Using array or array-like(ex: dynamic array) in this problem is prohibited.

Last semester, NingQung was frustrated with his final project, Catan. Catan is a famous board game. If you don't know what it is, you can learn how to play on this website. But in this problem, you don't have to know how to play.

In Catan, the map is built with a hexagon, which is like Figure 2.2.

NingQung is a Frontend Engineer in his team, and he had no idea about how to generate the map. TA Howard was thinking for several hours, and he realized that last year, ex-TA SnowEx built a Cuboid Super Infinity Exporter, and then Howard came up with a solution!

FIGURE 2.2: Catan map.

As a Computer Science Ice-cream Engineer, you have to build a ***CSIE***, aka ***Catan Simple Islands Engine***!

The user will input N and L, N is the number of layers, and L is the length of an island. Figure 2.3 shows the above description.

You should print the road with '`/`', '\\', or '`-`', and print the intersection with '`*`'. The number of points are same for each edge. You can see some examples in the end.

Otherwise, you need to align the map to the top-left corner. Figure 2.4 shows the correct and wrong examples.

Some cases can't generate the map:
1. N is less than 1.
2. L is less than 3.
3. The width of the map is larger than 80.

I promise that all inputs can fit in int32_t. You should handle these error cases.

I know this is a tough challenge. As a Super Friendly TA, I will use the

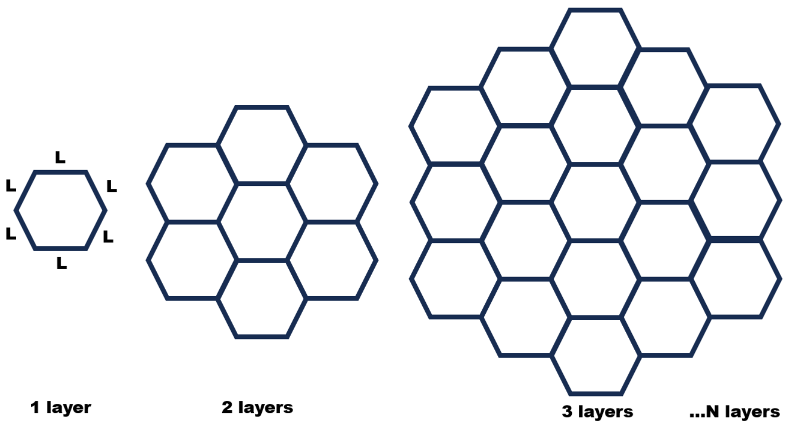

FIGURE 2.3: What is N and L?

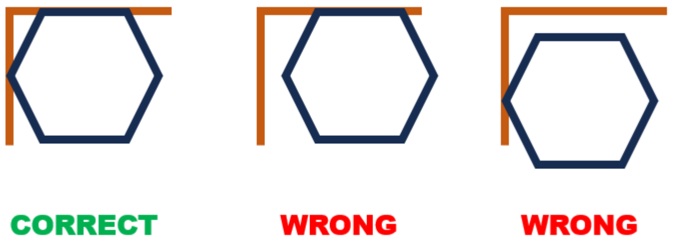

FIGURE 2.4: A correct and two wrong alignment examples.

following Grading Criteria:

Grading Criteria	
Error case	x pts(x >0)
N=1	y pts(y >x)
L=3	z pts(z >= y)
Others	20 - x - y - z pts

As a very very Super Friendly TA, I give you some hints:
1. Look at hw0205 last year, and here is the solution: link
Some examples:
Example 1:

```
Please input the length: 3
Please input the number of layer: 2
      *-*
     /   \
  *-*     *-*
 /   \   /   \
*     *-*     *
 \   /   \   /
  *-*     *-*
 /   \   /   \
*     *-*     *
 \   /   \   /
  *-*     *-*
     \   /
      *-*
```

Example 2:

```
Please input the length: 3
Please input the number of layer: 3
          *-*
         /   \
      *-*     *-*
     /   \   /   \
  *-*     *-*     *-*
 /   \   /   \   /   \
*     *-*     *-*     *
 \   /   \   /   \   /
  *-*     *-*     *-*
 /   \   /   \   /   \
*     *-*     *-*     *
 \   /   \   /   \   /
  *-*     *-*     *-*
 /   \   /   \   /   \
*     *-*     *-*     *
 \   /   \   /   \   /
  *-*     *-*     *-*
     \   /   \   /
      *-*     *-*
         \   /
          *-*
```

Example 3:

```
Please input the length: 5
Please input the number of layer: 1
    *---*
   /     \
  /       \
 /         \
*           *
 \         /
  \       /
   \     /
    *---*
```

Using array or array-like(ex: dynamic array) in this problem is prohibited.

2.6 Bonus: What Happens?? (5 pts)

Please read the following code and guess the output. Compile this code and run it. Is the output the same with what you guess? Please explain the reason of the output in **Chinese**.

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

uint32_t ui = 0;
uint16_t us = 0;
int32_t si = -1;

int main()
{
    int64_t r1 = ui + si;
    int64_t r2 = us + si;
    printf("%ld %ld\n", r1, r2);
}
```

Hint: You can look up the reason from C11. I have uploaded C11 draft on my website. The keyword: **Integer Conversion** and **Integer Promotion**.