Object Oriented Programming (Week 2)

김민태 (rlaaslxo98@naver.com)

KWANGWOON UNIVERSITY
DEPT. OF COMPUTER ENGINEERING

Contents

Assignment 1-1

- Write a program that draws the Sierpiński triangle pattern given with an input (unsigned char) illustrated as Figure 1-1 & 1-2. The program calculates the size (3N) of the shape using an input (k), where N is a power of 2 (2^{k-1}, $1 \le k \le 8$). Given the input, the program should draw a shape by using a character "$".

Input	Output
2	$ $ $ $$$$$ $ $ $ $ $ $ $$$$$ $$$$$

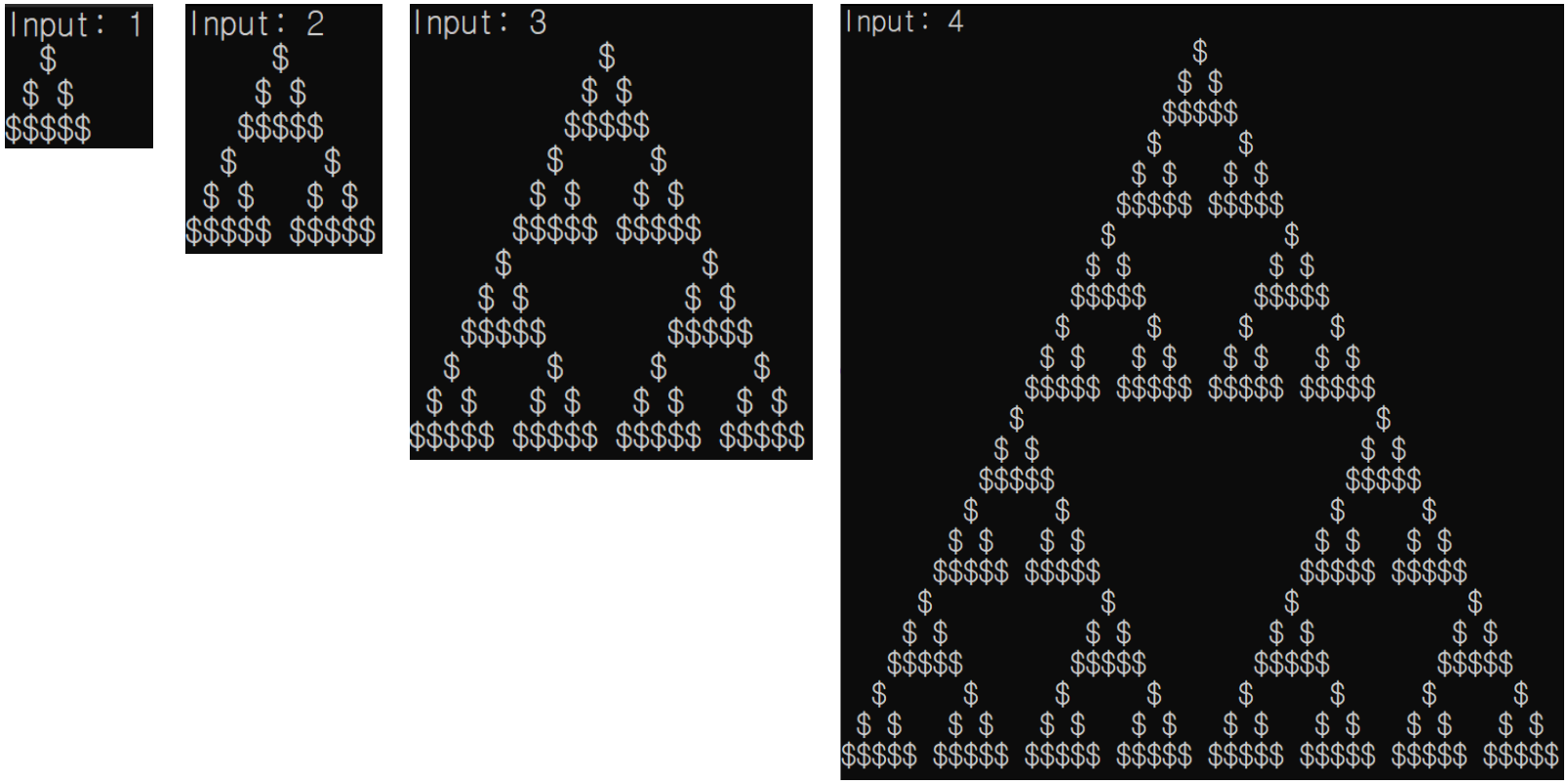
< Figure 1-1 >

Input	Output
3	$ $ $ $$$$$ $ $ $ $ $ $ $$$$$ $$$$$ $ $ $ $ $ $ $$$$$ $$$$$ $ $ $ $ $ $ $ $ $ $ $ $ $$$$$ $$$$$ $$$$$ $$$$$

< Figure 1-2 >

Assignment 1-1

- 결과 예시



Assignment 1-1

- 화면 출력과 입력 받기

```
#include <iostream>
#include <cmath>
using namespace std;

int main()
{
    int k, N;

    cout << "Input k: ";
    cin >> k;

    cout << "output N: ";
    N = pow(2, k - 1);
    cout << N << endl;

    return 0;
}
```

C언어의 <stdio.h> & <math.h>

std(표준 이름 공간)에 선언된 모든 이름에 대해 std::를 생략 가능
std::cout → cout
std::endl → endl

화면 출력

키 입력

개행 ('₩n')

< 결과화면 >

```
Input k: 3
output N: 4
```

```
Input k: 5
output N: 16
```

Assignment 1-1

- 간단한 삼각형 패턴 찍기

```
#include <iostream>
using namespace std;

int main()
{
    char triangle[3][6] = { "  $  ",
                            " $ $ ",
                            "$$$$$" };

    for (int i = 0; i < 3; i++)
    {
        for (int j = 0; j < 5; j++)
        {
            cout << triangle[i][j];
        }
        cout << endl;
    }

    return 0;
}
```

< 결과화면 >

Assignment 1-2

- Write a program to compute the real roots of a quadratic equation using **float** data types. The quadratic formula states that the roots of $ax^2 + bx + c = 0$, when $a \neq 0$, are

$$x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \quad and \quad x_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

Your program is to prompt the user to enter the constants (a, b, c). It is then to display the roots. Note that you should account for potential errors meticulously before using the formula above to solve for the values of x_1 and x_2.

Input	Output
a: 0 b: 1 c: 1	Unexpected factor of a quadratic term
a: 1 b: -2 c: 1	X1, X2: 1 (double root)
a: 1 b: 1 c: 1	The equation has no real number solutions.
a: 1 b: 62.1 c: 1	X1: - 0.0161072, X2: -62.0839

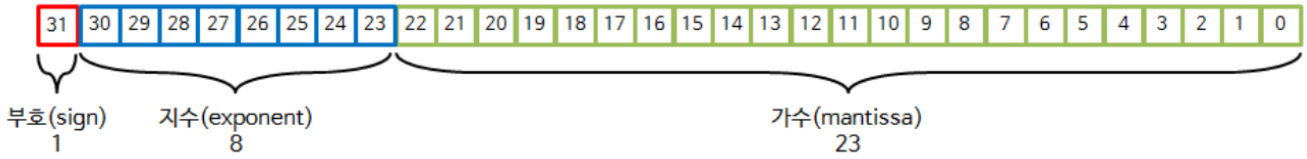
Assignment 1-2

- 제시된 식을 그대로 사용한 경우, 제시한 결과 대비 오차 발생.

```
the roots of 1x^2 + 62.1x + 1 = 0 :
X1 = -0.0161076, X2 = -62.0839
```

$$x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \quad and \quad x_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

- IEEE 754 standard : Single-precision floating-point format (32bits)



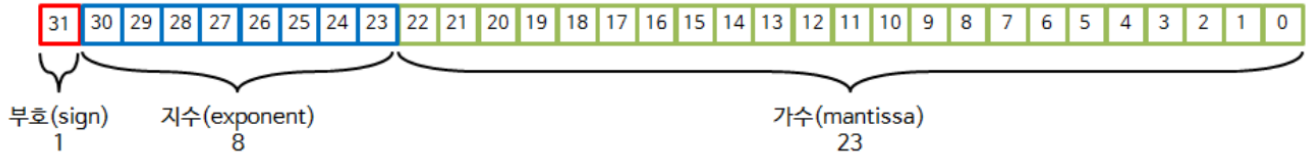
$$\text{Number} = (-1)^s * (1.m) * 2^{e-127}$$

– Example : $(-5.75)_{10} = (-101.11)_2 = 1 * (1.0111) * 2^2$

→ 11000000_10111000_00000000_00000000 = $(-5.75)_{10}$

Assignment 1-2

- Round-off error
  - IEEE 754 standard : Single-precision floating-point format (32bits)



$$\text{Number} = (-1)^s * (1.m) * 2^{e-127}$$

  - Example : $(4.2)_{10}$ = $(100.001100110011\cdots)_2$ = 1 * $(1.00001100110011\cdots)$ * 2^2)

    → 01000000_10000110_01100110_01100110 = $(4.19999980926513671875)_{10}$

  - 유사한 두 숫자 간 뺄셈 연산 시 에러가 증폭됨.

    → 이러한 연산이 포함된 경우, 식을 변형하여 사용해 에러를 줄일 수 있음.

$$x_1 = \frac{-62.1 + \sqrt{(62.1)^2 - 4}}{2} \quad \Rightarrow \quad x_1 = \boxed{?}$$

Assignment 1-2

- pow()
  - Returns the base raised to power exponent.
  - format

```
#include <cmath>
float pow(float base, float exponent);
double pow(double base, double exponent);
long double pow(long double base, long double exponent);
```

  - usage

```
cout << pow(2, 3);
// print 8
```

- sqrt()
  - Returns square root of x.
  - format

```
#include <cmath>
float sqrt(float x);
double sqrt(double x);
long double sqrt(long double x);
```

  - usage

```
cout << sqrt(49);
// print 7
```

Assignment 1-3

- Write a program that calculates LCM (Least common multiple) using a GCD (Greatest common divisor) function. The greatest common divisor of integers x and y is the largest number that divides both of them without leaving a remainder, while the least common multiple is the smallest positive integer that is divided by both x and y. Note that you are not allowed to use the built-in gcd function.

  If y is equal to 0, then gcd(x, y) is x; otherwise, gcd(x, y) is recursively defined as gcd(y, x%y), where % is the modulus operator. Receive x and y as integer data types and make sure to avoid arithmetic overflow.

Assignment 1-3

- Pseudo code of Euclidean algorithm

```
if y < x, swap (x, y)
while x does not equal 0
    r = y mod x
    y = x
    x = r
endwhile
output y
```

  - Example : GCD of 695 and 1112
    - ① 1112 mod 695 = 417
    - ② 695 mod 417 = 278
    - ③ 417 mod 278 = 139
    - ④ 278 mod 139 = 0
    - **GCD of 695 and 1112 is 139**

  - You have to write this algorithm using recursive function in this assignment.

Assignment 1-3

- Arithmetic overflow
  - data type **int**
    - size : 4btye = 32bit
    - range : –2,147,483,648 ~ 2,147,483,647 (= -2^{31} ~ $2^{31}-1$)
  - 연산 결과가 범위의 최대값 초과시, 32bit를 넘어가는 bit이 소멸되는 overflow 발생.

```
#include <iostream>
using namespace std;

int main()
{
    int a = 50000;
    int b = 100000;

    int result = a * b;

    cout << a << " * " << b << " = ";
    cout << result << endl;

    return 0;
}
```

```
50000 * 100000 = 705032704
```

≠ 5,000,000,000

Assignment 1-4

- Write a program that finds the inverse of the matrix below :

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

If we have A matrix, then, the inverse matrix A^{-1} of 3x3 is $A^{-1} = \frac{1}{det(A)} C^T$. C means Cofactor, which means you have to calculate for each value to find to get all cells about A^{-1} with determinant illustrated as below.

$$A = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}, C = \begin{matrix} +\begin{vmatrix} e & f \\ h & i \end{vmatrix} & -\begin{vmatrix} d & f \\ g & i \end{vmatrix} & +\begin{vmatrix} d & e \\ g & h \end{vmatrix} \\ -\begin{vmatrix} b & c \\ h & i \end{vmatrix} & +\begin{vmatrix} a & c \\ g & i \end{vmatrix} & -\begin{vmatrix} a & b \\ g & h \end{vmatrix} \\ +\begin{vmatrix} b & c \\ e & f \end{vmatrix} & -\begin{vmatrix} a & c \\ d & f \end{vmatrix} & +\begin{vmatrix} a & b \\ d & e \end{vmatrix} \end{matrix} = \begin{matrix} +(ei - fh) & -(di - fg) & +(dh - eg) \\ -(bi - ch) & +(ai - cg) & -(ah - bg) \\ (bf - ce) & -(af - cd) & +(ae - bd) \end{matrix}$$

Assume that a, b, c, d, e, f, g, h and i are integer values and they can be initialized by a user. You must find the inverse matrix composed of 9 elements represented in double precision.

Input	Output
2 2 1 -1 1 0 0 0 0	The inverse matrix does not exist.
1 2 3 0 1 4 5 6 0	-24 18 5 20 -15 -4 -5 4 1

Assignment 1-4

- Method to get inverse of 3x3 matrix :

$$A = \begin{bmatrix} 1 & 0 & 2 \\ 2 & -1 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

- ① Get and check determinant of the matrix

$$\det(A) = 5$$

- ② Get the cofactor matrix

$$C = \begin{bmatrix} -1 & -2 & 3 \\ 2 & -1 & -1 \\ 2 & 4 & -1 \end{bmatrix}$$

- ③ Transpose the cofactor matrix

$$C^T = \begin{bmatrix} -1 & 2 & 2 \\ -2 & -1 & 4 \\ 3 & -1 & -1 \end{bmatrix}$$

- ④ Divide each elements using determinant

$$A^{-1} = \frac{1}{det(A)} C^T = \frac{1}{5} \begin{bmatrix} -1 & 2 & 2 \\ -2 & -1 & 4 \\ 3 & -1 & -1 \end{bmatrix} = \begin{bmatrix} -0.2 & 0.4 & 0.4 \\ -0.4 & -0.2 & 0.8 \\ 0.6 & -0.2 & -0.2 \end{bmatrix}$$

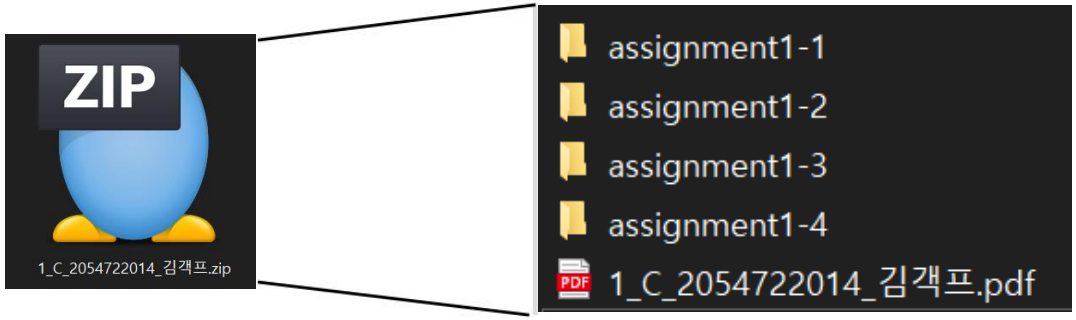
과제 제출 방법

- **FTP Upload (Klas 과제 제출 X)**
  - Address : ftp://223.194.8.1:1321
  - username : IPSL_OBJ
  - password : ipslobj_2023

- **Due date**
  - Soft copy: 마감일 3/17(금) 23:59:59까지 제출 (서버시간 기준)
  - Delay
    - 마감일 이후 +7일까지 제출 가능
    - 단, 1일 초과마다 과제 총점의 10%씩 감점

과제 제출 방법

- Soft copy
  - 과제(보고서, 소스 코드)를 압축한 파일 제출
    - 설계반_실습반_학번_이름.zip
      - 예) 설계1반 수강, 실습 A반: 1_A_학번_이름.zip
      - 예) 설계 수강, 실습 미수강: 2_0_학번_이름.zip
      - 예) 설계 미 수강, 실습 C반: 0_C_학번_이름.zip



  - 과제 수정하여 업로드 시 버전 명시
    - 설계반_실습반_학번_이름_verX.zip

과제 제출 방법

- Soft copy
  - 과제 보고서
    - 영문 또는 한글로 작성
    - **반드시 PDF**로 제출 (PDF 외 파일 형식으로 제출시 0점 처리)
    - 문제 및 설명(문제 capture 금지) / 결과 화면 / 고찰
    - **소스코드 제외**
    - 분량 제한 없음
    - **표절 적발 시 0점 처리**
  - 소스 코드
    - Visual Studio 2022 community 사용 필수
      - https://docs.microsoft.com/ko-kr/visualstudio/install/install-visual-studio?view=vs-2022
    - STL (Standard Template Library) 사용 금지 (vector, map, algorithm 등)
    - Debug 폴더를 제외한 모든 파일 제출
      - .sln 파일 포함(.cpp 만 제출하지 말것)
    - **각 문제마다 프로젝트 파일 생성 필수**
    - **주석 반드시 달기**
    - **소스코드 표절 적발 시 0점 처리**

END OF PRESENTATION

Q&A