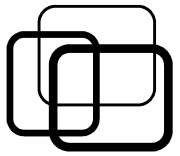
Control Flow Statements Exercises

Inst. Nguyễn Minh Huy

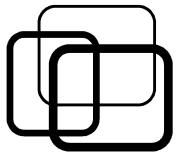
Practice



- Input from keyboard in C/C++:
  - **scanf**, **std::cin >>** do not clean input stream.
  - Clean and safe input from keyboard:
    - Input a whole line (includes '\n').
    - Extract data from the line: sscanf, std::stringstream.
    - Validate data by result value.

```
int  a;
char  line [100];
do {
      printf("Enter a integer = ");
      fgets( line, 100, stdin );
} while ( sscanf( line, "%d", &a ) != 1 );
// C++
// } while ( !( std::stringstream( line ) >> a ) );
```

Practice



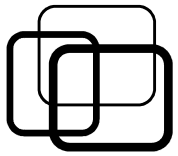
- Floating-point error:
  - Cannot represent real number exactly.

```
float  x = 0.1;
printf(“%.15f”, x);                    // Print: 0.100000001490116
```

  - Accumulate small errors → big error.

```
int n = 1000;
float x = 0.1 * n;
float y = 0.0;
for (int i = 0; i < n; ++i)
    y += 0.1;                          // Accumulate error here!
printf("%.15f\n%.15f\n", x, y);        // x = 100.000000000000000
if (x != y) printf("not equal.");      // y = 99.999046325683594
```

Practice



- Practice 3.6:

  Write C/C++ program to classify a triangle:

  - Enter 3 positive real numbers a, b, c (re-enter if invalid).
  - Check if a, b, c can forms a triangle.
  - If yes, print the triangle type.
    (normal, right, isosceles, right-isosceles, equilateral).

Input format:

Enter 3 positive real numbers = 3 4 5

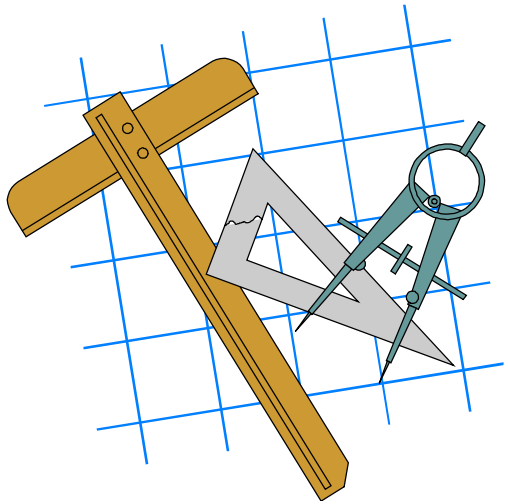
Output format (can form a triangle):

Can form a triangle.

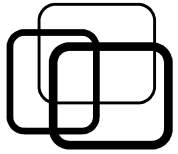
Right triangle.

Output format (cannot form a triangle):

Cannot form a triangle!



Practice



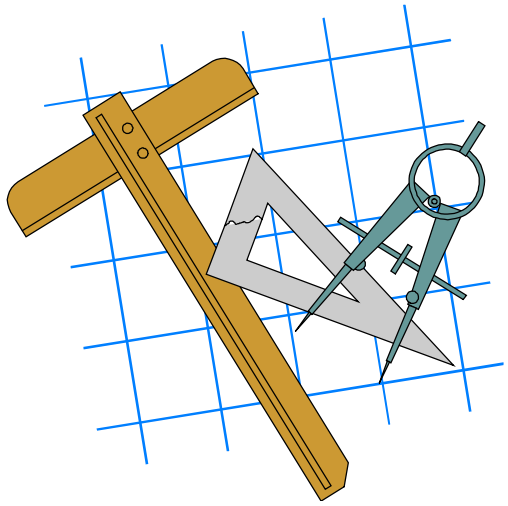
- ## Practice 3.7:

Write C/C++ program to compute electricity bill:

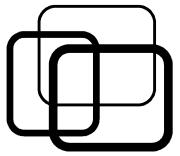
- Enter old and new kWh (re-enter if invalid).
- Compute the price and print result.

Price table:

Levels	Consumed kWh	Unit prices
1	0 – 100	1549 đ
2	101 – 150	1600 đ
3	151 – 200	1858 đ
4	201 – 300	2340 đ
5	301 – 400	2615 đ
6	401 – more	2701 đ



Practice



- Practice 3.8:

  Write C/C++ program find max and min as follow:

  - Enter a sequence of positive integers. (re-enter if invalid, stop if enter zero).
  - Print max and min (do not include zero).

  Notes: do not use array.

  Input format:

  Number 1 = 5

  Number 2 = 3

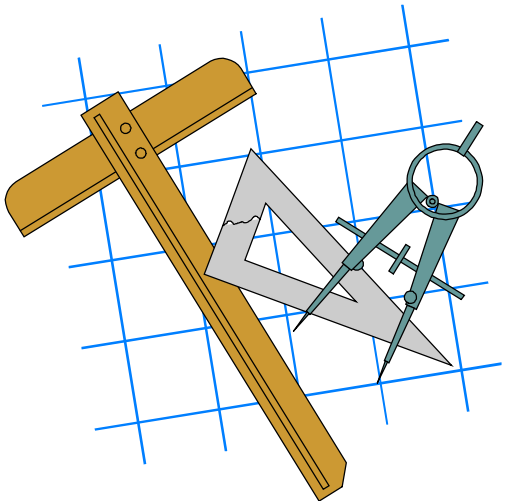
  Number 3 = 9

  Number 4 = 0

  Output format:

  Max = 9

  Min = 3



Practice

Practice 3.9:

Write C/C++ program to find prime numbers:

- Enter a positive integer N (re-enter if invalid).
- Print all prime numbers <= N.

Input format:

Enter a positive integer = 11

Output format:

#1 = 2

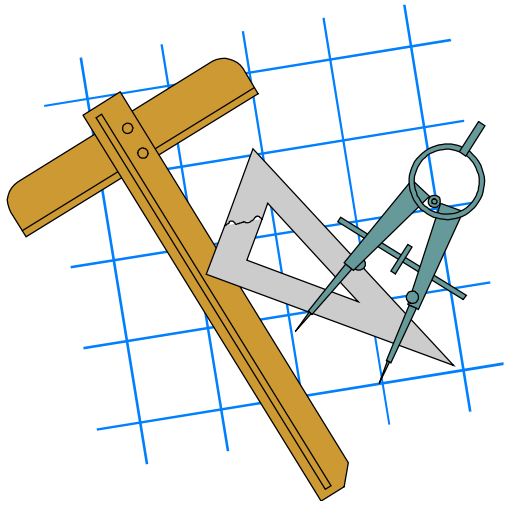
#2 = 3

#3 = 5

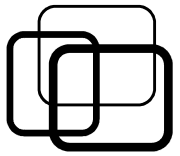
#4 = 7

#5 = 11

There are 5 prime numbers.



Practice



- **Practice 3.10:**

  Write C/C++ program to find GCD (greatest common divisor) and LCM (least common multiple):

  - Enter 2 positive integers a, b (re-enter if invalid).
  - Print:
    - GCD of a, b.
    - LCM of a, b.

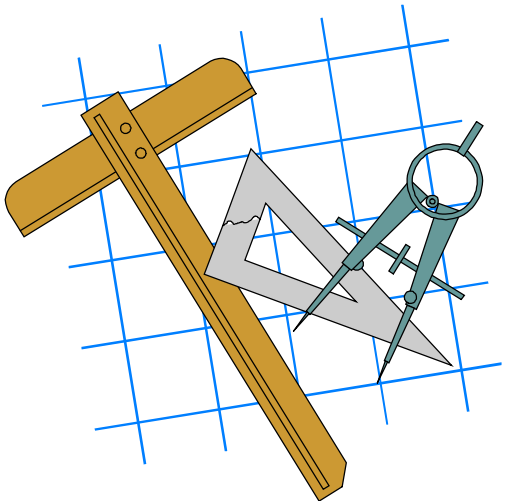
Input format:

Enter 2 positive integers = 8 12

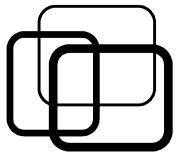
Output format:

GCD = 4

LCM = 24



Practice



- **Practice 3.11:**

  Write C/C++ program to print bit representation of an integer:

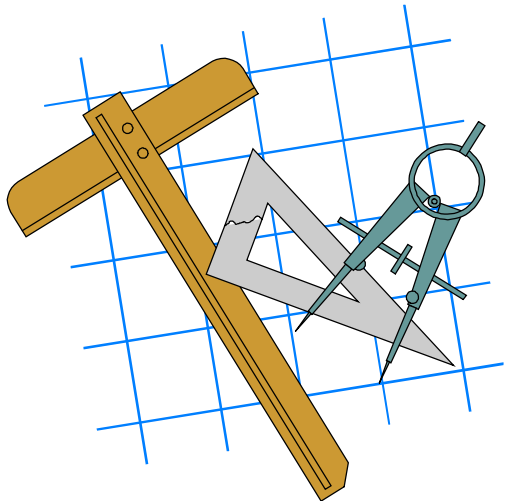
  - Enter integer N and positive integer M.
  - Print the first M bit from the right of N.

  Input format:

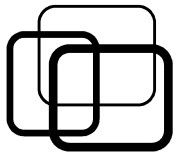
  Enter N, M = 11 3

  Output format:

  The first 3 bit from the right of 11: 011



Practice



- ## Practice 3.12:

Write C/C++ program to read phone number as follow:

- Enter a phone number (10 digits max) (re-enter if invalid).
- Print how to read the phone number.

Notes:

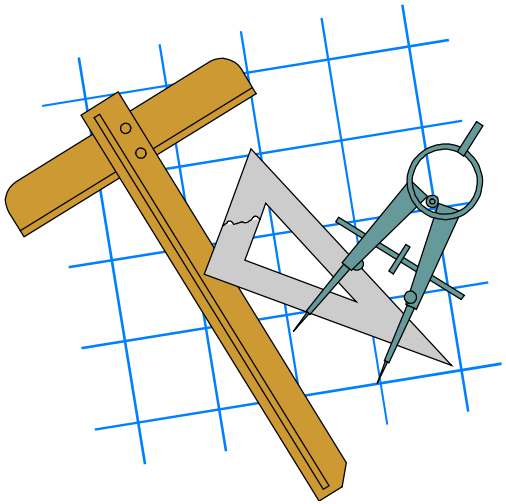
- Do not use array/string processing (get element, get length).

Input format:

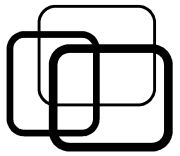
Phone number: 838272727.

Output format:

Eight three eight two seven two seven two seven.



Practice



- Practice 3.13:

  Write C/C++ program as follow:

  - Enter a positive integer N (re-enter if invalid).
  - Check if:
    a) Digits of N is in descending order from the ones.
    b) Digits of N is symmetric.

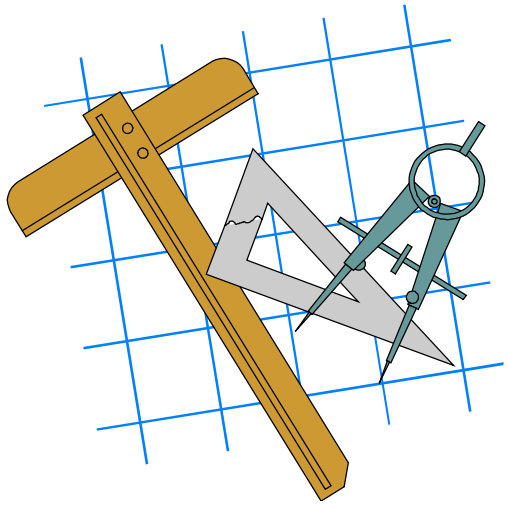
Input format:

Enter N = 12344321

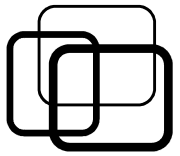
Output format:

Not descending.

Symmetric.



Practice



- **Practice 3.14:**

  Write C/C++ program to print month calendar as follow:

  - Enter month and year (re-enter if invalid).
  - Enter day of week of the first day in the month (for alignment).
  - Print the calendar of the month.

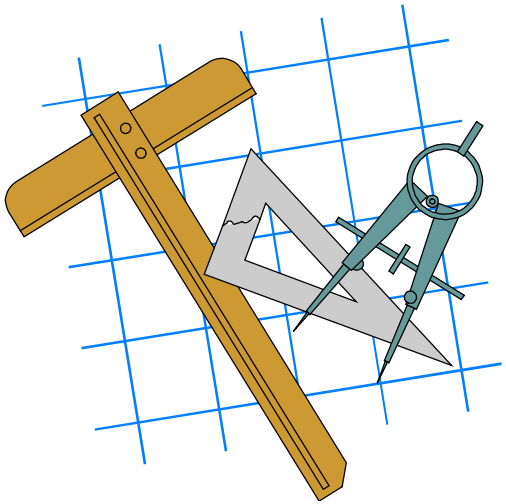
  Input format:

  What month and year to view calendar? 9 2022

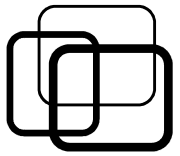
  What is the day of week of the first day? 5

  Output format:

```
Mon Tue Wed Thu Fri Sat Sun
                  1   2   3   4
  5   6   7   8   9  10  11
 12  13  14  15  16  17  18
 19  20  21  22  23  24  25
 26  27  28  29  30
```



Practice



- **Practice 3.15:**

  Write C/C++ game for grade 1 student to practice +, - as follow:

  - Computer asks randomly + or – between two positive integers.
  - Player answers the result (re-enter if wrong).
  - Computer continues to asks until player answers zero.
  - Constraint: operands, result in range [1..99].

  Notes: use srand, rand (<stdlib.h>) to generate random numbers.

  Game format:

  Question 1: 2 + 5 = ?

  Answer: 7

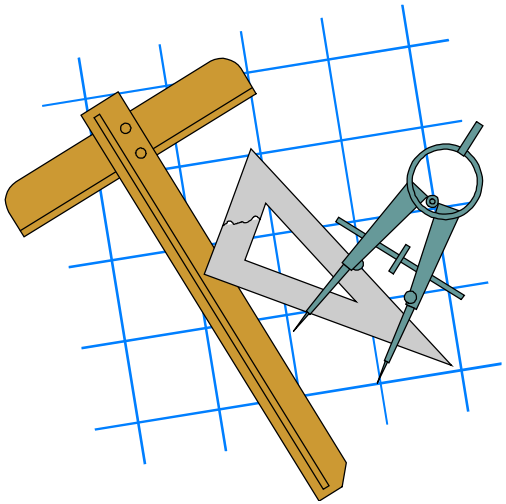
  Question 2: 18 – 9 = ?

  Answer: 8

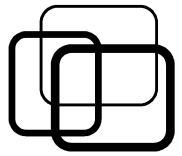
  Answer: 9

  Question 3: 6 + 25 = ?

  Answer: 0



Practice



- Generate random number in C/C++:
  - Random ~ unpredictable, uniformly distributed.
  - Hardware source: dice, cards, atmospheric noise.
  - Software source: uniform sequence.
    - A0 = random seed, An = F(An-1).
  - C random: library <stdlib.h>, <time.h>

```
srand( time( 0 ) );                                  // Weak random seed
for (int i = 0; i < 10; ++i)
    printf("%d ", rand( ) % (max – min + 1) + min); // Not uniform
```

  - C++ random: library <random>

```
std::random_device  hardRand;                  // Hardware random source.
std::mt19937  softRand( hardRand( ) );         // Software random source.
std::uniform_int_distribution  dist( min, max ); // Uniform function.
for (int i = 0; i < 10; ++i)
    std::cout << dist( softRand ) << ' ';
```