LAB 10/01

1. Homework problem

(A) Write a program to calculate the diameter, the circumference, and the area of a circle with a radius of 6.75.

Assign the radius of float variable, and then output the radius with an appropriate message. Declare a named const PI with the value 3.14159.

The program should output the diameter, the circumference, and the area, each on a separate line. **Print each value to five decimal places within a total field width(欄位) of 10**.

Note1: when the compiler reads this specifier: %10.5f

i. **The compiler prepares 10 columns to output this real number with the five right-most columns for the fraction part.**
ii. **If the real number has less than five digits in the fraction part, the compilers pads the remaining columns with zero.**
iii. **The 6th column from the right is the decimal point.**
iv. **The remaining four columns are the integer part. If the real number has less than four digits in the integer part, the output is padded with blank on the left.**

%10.5f 的意思: 這個 **format** 對應的 **data** 其輸出格式如下:

留十個欄位來輸出實數，其中後五個欄位為小數點後面的位數，不足五位則在後方補「**0**」，後面數來第六位數是小數點，剩下四位放整數，整數不足四位則在前方插入空白。

(B) Write a program that accepts an integer between 7 and 9 digits long.

a. Extracts and prints the third-rightmost digit of the input data.

b. Writes the integer with commas between every third digit starting from the right.

Example:

```
(A)
diameter:   13.50000
circumference:   42.41147
area:  143.13870

--------------------------
(B)
Input an integer between 7 and 9 digits long: 12345678
The third-rightmost digit of the input data is 6
The input data with commas between every third digit is 12,345,678
--------------------------------
```

2. Homework problem

(A) Solve a set of simultaneous equations:

ax + by = c

dx + ey = f

Input data: 6 real numbers (a, b, c, d, e, f).

Formulas for the solution:

$$x = \frac{ce - bf}{ae - bd}$$

$$y = \frac{af - cd}{ae - bd}$$

Output all the input values a, b, c, d, e, and f and the computed values for x and y.

(B) Write a program that prompts for and reads a floating-point value. The program prints the whole part on one line and the decimal (fraction) part on a second line.

For example, if the input value is 123.456, it would print the output:

the input value is 123.456

the whole part is 123

the decimal(fraction) part is 0.456

Example:

```
(A)
Input a,b,c,d,e and f: 1 2 3 4 5 6
a=1.000000, b=2.000000, c=3.000000, d=4.000000, e=5.000000, f=6.000000
x = -1.000000, y = 2.000000
--------------------
(B)
Input a float value: 123.456
the input value is: 123.456
the whole part is: 123
the decimal(fraction) part is: 0.456
--------------------------------
```

3. Correct one error in following source code. Set the value of r, t and pi, then completing following program and output the volume and area of a ball with a given radius. Notice the spacing between numbers, please complete each formula with only one printf().

```c
#include<stdio.h>

int main()
{
    double r, t;
    const float pi;

    r = 7.25, t = 4./3;
    pi = 3.14159;

    printf("r=%lf\nt=%lf\npi=%f\n\n", r, t, pi);

    printf("12345678901234567890123456789012345678901234567890\n");
    printf(……);
    printf(……, t, pi, r, t*pi*r*r*r);  //Volume of ball
    printf(……, 4, pi, r, 4*pi*r*r);  //Area of ball
    printf(……, r, t, r*t);

    return 0;

}
```

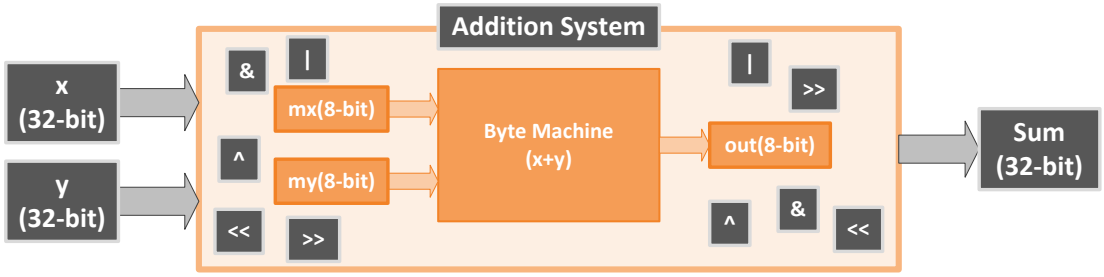
Your output should be like

```
r=7.250000
t=1.333333
pi=3.141590

12345678901234567890123456789012345678901234567890123456789012345678901234567890
   3.14159 = 3.14          + 00000.1500000000 + 0000.00009
1.3333333  * 0003.1415901  * (0000007.2500)^3 =  1596.2550291
4          *       3.1416  * (7.250        )^2 = 000660.5193
007.250000 * 1.3333333333  = 00000009.6666667
```

All the values except ^2, ^3 in the red box should be shown from formatted data (%...f, %...d, …).

4. An addition system can do addition with two “int” numbers whose lengths are both 32 bits (4 bytes) and then output an “int” result (32-bit, 4-byte). But this system has a “byte machine” which can only do addition with two numbers whose lengths are both 8 bits (1 byte) and then output an 8-bit result. Outside the byte machine, the system can only do bit operation, i.e., &, |, ^, >>, <<. Please implement this system as above requirement.



Output Example:

```
Input: 1234567 987654321
1234567 + 987654321 = 988888888
```

How to capture the rightmost 7-bit of a 32-bit number:

		11	10	9	8	7	6	5	4	3	2	1	0
2405	…	1	0	0	1	0	1	1	0	0	1	0	1
0x7F	& …	0	0	0	0	0	1	1	1	1	1	1	1
	…	0	0	0	0	0	1	1	0	0	1	0	1

		11	10	9	8	7	6	5	4	3	2	1	0
4021	…	1	1	1	1	1	0	1	1	0	1	0	1
0x7F	& …	0	0	0	0	0	1	1	1	1	1	1	1
	…	0	0	0	0	0	0	1	1	0	1	0	1

How to put the calculation result into the rightmost 7 bits of the final answer:

		11	10	9	8	7	6	5	4	3	2	1	0
out	...	0	0	0	0	1	0	0	1	1	0	1	0
0x7F	& ...	0	0	0	0	0	1	1	1	1	1	1	1
		0	0	0	0	0	0	0	1	1	0	1	0
sum	or ...	0	0	0	0	0	0	0	0	0	0	0	0
	...	0	0	0	0	0	0	0	1	1	0	1	0

Please follow below code to complete this program.

```c
#include <stdio.h>

int main()
{
    int x, y, sum=0;  // input: x,y ; output: sum
     // machine input: mx,my ; machine output: out
    unsigned char mx, my, out;  //8 bits
    bool ca=0;  // carry

    printf("Input: ");
    scanf("%d %d", &x, &y);

    mx = x&0x7F;
    my = y&0x7F;
    out = mx + my + ca;
    ca = out>>7;
    sum |= out&0x7F;



    // The rest part you have to finish...



    printf("%d + %d = %d\n", x, y, sum);

    return 0;
}
```