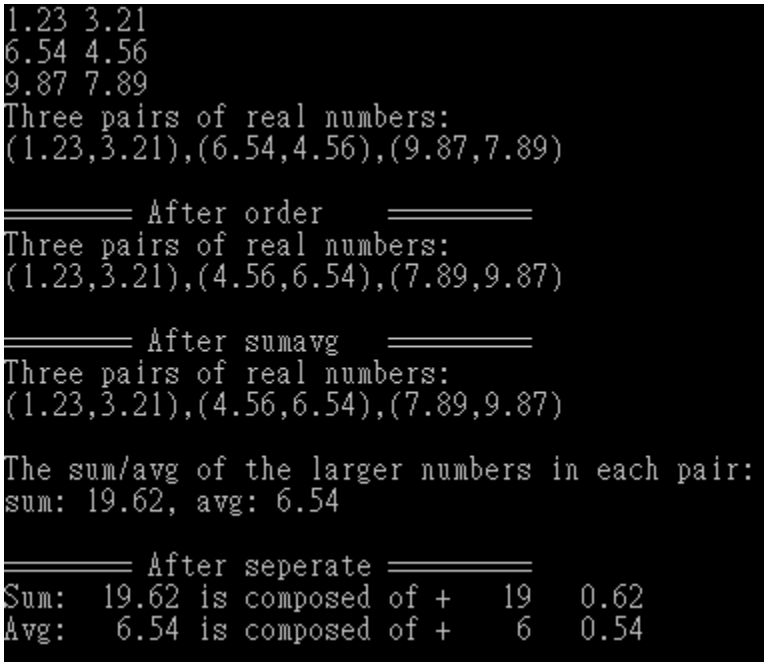
Lab09

1. Homework I

Write a program that makes the user input three pairs of real numbers (totally, six real numbers). Find out the larger one in each pair with a subroutine "***order***", which has two call-by-address parameters. After passing a pair of real numbers as its parameters, the smaller one is stored in the first parameter and the larger one is stored in the second parameter.

Then, the three larger numbers in the input pairs are passed to a subroutine "***sumavg***", in order to find out the sum and average of the three numbers. It is worth mentioning that the three numbers are passed by value, and the two results (sum and average) are passed by address.

Finally, the sum and average numbers computed by ***sumavg*** are separately sent to a subroutine "***separate***". In "separate", the passed number are separated into three parts: sign, whole number magnitude and fractional part. The results of separation are stored in three call-by-address parameters respectively. Print the results in main function as following example. The program should be able to execute repeatedly until user entering Ctrl-D/Ctrl-Z.



2. Homework II

Write a program for an Automatic Teller Machine that dispenses money. The user should enter the amount desired (a multiple of 10 dollars) and the machine dispenses this amount using the least number of bills. The bills dispensed are 50s, 20s, and 10s. **Write a function** that determines how many of each kind of bill to dispense. It is worth mentioning that **all the output should be done in the main function**. The program should be able to execute repeatedly until user entering Ctrl-D/Ctrl-Z.

```
1234567890
1234567890 is dispensed to 50s: 24691357, 20s: 2, 10s: 0.
1230
1230 is dispensed to 50s: 24, 20s: 1, 10s: 1.
43210
43210 is dispensed to 50s: 864, 20s: 0, 10s: 1.
543210
543210 is dispensed to 50s: 10864, 20s: 0, 10s: 1.
^D
```

3. Function Practices

Write a program composed of two cases (a&b) and each case should be executed repeatedly until user enters Ctrl-z.

(a) Write a function that translates the amount of Arabic number into Chinese representation.

(b) A complex number is a number that contains two parts: a real part and an 'imaginary' part. The real part is any real number. The imaginary part is a real number multiplied by the imaginary unit written as the lower-case letter 'i'. Complete the following functions.

```
void C_print (double a, double b)
{
// print out (a + bi) or (a - bi) according to the sign of b
// print out a if b==0
}
void C_add (double a, double b , double c, double d, double *e , double *f)
{
// (e+fi) = (a+bi)+(c+di)
}
```

```
void C_sub (double a, double b , double c, double d, double *e , double *f)
{
// (e+fi) = (a+bi)-(c+di)
}
void C_mul (double a, double b , double c, double d, double *e , double *f)
{
// (e+fi) = (a+bi)*(c+di)
}
void C_div (double a, double b , double c, double d, double *e , double *f)
{
// (e+fi) = (a+bi)/(c+di)
}
```

Let user input 4 numbers indicating a, b, c, d of the two complex number (a+bi) and (c+di). Output the results of these 4 operations:

```
{    // Case (b)
      …… // Something you should implement.

    // The output should be like: (5 + 3i) + (4 - 3i) = 9
      C_print(……);
      printf(……);
      C_print(……);
      C_add(……);
      printf(……);
      C_print(……);
      printf("\n");

      …… // Something you should implement.

      return 0;
}
```

Your output should look like this:

```
(5 + 3i) + (4 - 3i) = 9
(5 + 3i) - (4 - 3i) = (1 + 6i)
(5 + 3i) * (4 - 3i) = (29 - 3i)
(5 + 3i) / (4 - 3i) = (0.44 + 1.08i)
```

FYI:

$$(a+bi)+(c+di)=(a+c)+(b+d)i$$

$$(a+bi)\times(c+di)=a\times c+a\times di+c\times bi+bi\times di=(ac-bd)+(ad+bc)i$$

$$\frac{a+bi}{c+di}=\frac{(a+bi)(c-di)}{(c+di)(c-di)}=\frac{(ac+bd)+(bc-ad)i}{c^2+d^2}$$

```
Active code page: 65001
═══ Case (a) ═══
7609802
柒佰陸拾萬玖仟捌佰零貳元整
1009040300
拾億零玖佰零肆萬零參佰元整
1000000001
拾億零壹元整
1000000000
拾億元整
1234567890
拾貳億參仟肆佰伍拾陸萬柒仟捌佰玖拾元整
^Z

═══ Case (b) ═══
5 3 4 -3
(5.00 + 3.00i) + (4.00 - 3.00i) = 9.00
(5.00 + 3.00i) - (4.00 - 3.00i) = (1.00 + 6.00i)
(5.00 + 3.00i) * (4.00 - 3.00i) = (29.00 - 3.00i)
(5.00 + 3.00i) / (4.00 - 3.00i) = (0.44 + 1.08i)
5 -3 4 -3
(5.00 - 3.00i) + (4.00 - 3.00i) = (9.00 - 6.00i)
(5.00 - 3.00i) - (4.00 - 3.00i) = 1.00
(5.00 - 3.00i) * (4.00 - 3.00i) = (11.00 - 27.00i)
(5.00 - 3.00i) / (4.00 - 3.00i) = (1.16 + 0.12i)
^Z
```

4. Play Cards

Write a program to deal a user five cards randomly without repeated cards and calculate the probability of :

a. Royal flush : 0.0001539%
b. Straight flush : 0.0013852%
c. Four of a kind : 0.0240096%
d. Full house : 0.1440576%

Each probability from a ~ d should be simulated with 2 methods of generating random numbers, one is uniform and the other is not (use %). Compare the simulation results of the 2 method and the theoretical analysis given above.

You should implement and use all the following functions to complete your program.

Function Name	Description	Parameters
deal_uniform	Deal five cards randomly with uniform generating method.	Use call by address to return 10 parameters for 5 cards, each card has two parameters (suit & number).
deal_notUniform	Deal five cards randomly with %.	
checkAll	Check if the dealt cards fit any kinds of Royal flush, straight flush …	Use call by address to return the amount of each types (4 parameter). And use call by value with 10 parameters for 5 cards for the following checking functions.
checkRF	Check if the cards are Royal flush. If they are, add the amount.	Use call by address to return the updated amount. And use call by value with 10 parameters for passing 5 cards.
checkSF	Check if the cards are Straight Flush. If they are, add the amount.	Use call by address to return the updated amount. And use call by value with 10 parameters for passing 5 cards.
checkFOAK	Check if the cards are Four of a kind. If they are, add the amount.	Use call by address to return the updated amount. And use call by value with 5 parameters for passing necessary information of 5 cards.
checkFH	Check if the cards are Full house. If they are, add the amount.	Use call by address to return the updated amount. And use call by value with 5 parameters for passing necessary information of 5 cards.
printResult	Print out all the result.	8 parameters for 2 methods, 4 for uniform and 4 for not uniform. [call by value]

Hint: The following simulation result is based on 2000000000 times dealing, and it takes 5653 seconds to generate the result.

```
-- (a) Rotal Flush    --
Theoretical: 0.0001539%
Uniform    : 0.0001586%
Not Uniform: 0.0001610%

-- (b) Straight Flush --
Theoretical: 0.0013852%
Uniform    : 0.0013956%
Not Uniform: 0.0014515%

-- (c) Four of A Kind --
Theoretical: 0.0240096%
Uniform    : 0.0239993%
Not Uniform: 0.0239765%

-- (d) Full House     --
Theoretical: 0.1440576%
Uniform    : 0.1440087%
Not Uniform: 0.1439027%
```