



國立台灣科技大學

微 算 機 概 論 與 應 用

指 導 教 授：陸敬互 教 授

微算機概論實習報告

(有bonus)

期中報告

班 級 ： 電機二甲

學 生 ： 黃偉智、韓承志

學 號 ： B11107035、B11107037

建檔日期： 2023/11/01

一、學習成果(程式功能說明，說明程式分幾個部分，各部分在做甚麼，搭配截圖code說明)

1.輸出字串(pString)和取得鍵盤輸入字元(getKey)的巨集

```
pString macro para1
    push ax
    push dx
    mov dx, offset para1
    mov ah, 09h
    int 21h
    pop dx
    pop ax
endm

getKey macro
    mov ah, 01h
    int 21h
endm
```

2.設定所需要的變數，x和y負責存輸入的兩個數字，result_hex存取x*y且後面會改成存取LCM的16進制結果，result_decimal則是存取最小公倍數

```
.model small

.data
x db 0 ; first input
y db 0 ; second input
result_hex dw 0 ; to store x*y (must be 2-byte)
result_decimal db 4 dup(0)

welcome_mes db 'Please input 2 numbers or press ESC to exit.$'
input1_mes db 'Enter first number: $'
input2_mes db 'Enter second number: $'
error_mes db 'Please enter a 2-digit decimal number!$'

new_line db 10, 13, '$'
output_gcd_mes db 'GCD is $'
output_lcm_mes db 'LCM is $'
```

3.此部分為輸入第一個數字中的第一位，並且判斷輸入的數字是否符合格式

```
.stack

.code
main proc
    mov ax, @data
    mov ds, ax

input1_digit1:|
    pString welcome_mes    ; print the meesages first
    pString new_line
    pString input1_mes
    getKey                 ; getting user input
    cmp al, 1bh            ; check whether input is ESC or not
    jne cont_input1_digit1
    mov ax, 4c00h          ; if yes, immediately end program
    int 21h

cont_input1_digit1:
    sub al, 30h
    cmp al, 09h
    jbe input1_digit2      ; check whether input is a number
    jmp input_error

input_error:
    mov ax, 0
    mov bx, 0
    pString new_line
    pString error_mes      ; error message when user input is not number or ESC
    pString new_line
    pString new_line
    jmp input1_digit1      ; when error, restart input from beginning
```

4.輸入第一個數字的第二部分，並且判斷輸入格式有無錯誤

```
input1_digit2:
    mov bh, al
    getKey          ; get user input
    cmp al, 1bh      ; check whether input is ESC or not
    jne cont_input1_digit2
    mov ax, 4c00h     ; if yes, immediately end program
    int 21h

cont_input1_digit2:
    sub al, 30h
    cmp al, 09h       ; check whether input is a number
    jbe input2_digit1
    jmp input_error
```

5.將此前輸入的第一個數字存進x，輸入第二個數字的第一位，並且判斷格式有無錯誤

```
input2_digit1:
    mov bl, al        ; store the first number in x
    mov cl, 04h
    rol bh, cl
    add bl, bh
    mov [x], bl       ; first number is now in x
    mov bx, 0

pString new_line
pString input2_mes
getKey              ; the logic from here is the same as input for first number
cmp al, 1bh
jne cont_input2_digit1
mov ax, 4c00h
int 21h

cont_input2_digit1:
    sub al, 30h
    cmp al, 09h
    jbe input2_digit2
    jmp input_error
```

6.輸入第二個數字的第二位數，判斷其錯誤，若無錯誤即可開始計算

```
input2_digit2:
    mov bh, al
    getKey
    cmp al, 1bh
    jne cont_input2_digit2
    mov ax, 4c00h
    int 21h

cont_input2_digit2:
    sub al, 30h
    cmp al, 09h
    jbe calculate_gcd ; if input is done, calculate GCD
    jmp input_error
```

7.將第二個數字存進y，便可開始計算，但因為電腦內部是以16進計算，而我們計算最大公因數和最小公倍數的值是以十進制運算，因此必須先將輸入的數字換算成16進制，當計算完成後再換為10進制輸出，

```
calculate_gcd:
    mov bl, al ; store the second number in y
    mov cl, 04h
    rol bh, cl
    add bl, bh
    mov [y], bl ; second number is now in y
    mov ax, 0 ; reset ax, bx, and cx
    mov bx, 0
    mov cx, 0

    mov al, [x] ; converting x into hexadecimal
    mov bl, al ; why? because input from user is always decimal,
    mov cl, 0fh ; but computer calculates using hexadecimal.
    and al, cl
    mov [x], al
    mov al, bl
    mov cl, 0f0h
    and al, cl
    mov cl, 04h
    shr al, cl
    mov cl, 0ah
    mul cl
    add [x], al
```

8.將第二個數字轉為16進制，並且計算 $x*y$ ，將結果存入result_hex

```
mov al, [y]      ; converting y into hexadecimal
mov bl, al
mov cl, 0fh
and al, cl
mov [y], al
mov al, bl
mov cl, 0f0h
and al, cl
mov cl, 04h
shr al, cl
mov cl, 0ah
mul cl
add [y], al

mov al, [x]      ; calculating x*y and store in result_hex
mov bl, [y]
mul bl
mov [result_hex], ax
```

9.開始計算最大公因數(輾轉相除法)，並將結果存到y

```
mov ah, 0        ; start calculating for GCD
mov al, [x]
mov bl, [y]

while_loop:
  cmp al, 0       ; if al = 0, exit loop
  je exit_loop

  cmp al, bl      ; make sure al is bigger than bl
  ja modulo       ; if yes, jump to modulo
  mov cl, al      ; if not, swap al and bl
  mov al, bl
  mov bl, cl
  mov cl, 0

modulo:
  mov ah, 0
  div bl          ; div bl = al/bl (remainder goes to ah, result goes to al)
  mov al, ah
  mov ah, 0
  mov [x], al
  mov [y], bl     ; the GCD result is in y
```

10.計算最小公倍數(即為兩數相乘除以最大公因數)，並將結果存入 result_hex，然後把y(最大公因數)轉為10進制呈現

```
exit_loop:

    pString new_line
    pString output_gcd_mes

    mov ax, [result_hex] ; LCM is result_hex divided by y
    mov ch, 0
    mov cl, [y]
    mov dx, 0
    div cx
    mov [result_hex], ax ; LCM is now in result_hex

    mov ax, 0 ; converting y back into decimal
    mov bx, 0
    mov cx, 0
    mov al, [y]
    mov cl, 0ah
    div cl
    mov cl, 04h
    rol al, cl
    add al, ah
    mov [y], al
```

11.將result_hex轉為10進制且丟入陣列result_decimal以利於輸出

```
    mov ax, 0 ; converting result_hex into decimal
    mov bx, 0
    mov cx, 0
    mov ax, [result_hex]
    mov cx, 0ah
    mov si, offset result_decimal

up:
    mov dx, 0 ; put each digit of result_hex into result_decimal
    div cx ; (result_decimal is an empty 4-bit array)
    mov [si], dl
    inc si
    cmp ax, cx
    jae up
    ..... [si] = 0
```

12.輸出最大公因數(y)的第一位和第二位，若第一位為0則不輸出第一位，僅輸出第二位

```
mov ah, 02h
mov dl, [y]      ; print the first digit of GCD
mov cl, 4
shr dl, cl
add dl, 30h      ; if first digit is 0, no need to print
cmp dl, 30h
je gcd_digit2
int 21h

gcd_digit2:
mov dl, [y]      ; print the second digit of GCD
shl dl, cl
shr dl, cl
add dl, 30h
int 21h
```

13.輸出最小公倍數，因為little-endian的關係因此要從陣列result_decimal的最後一位開始輸出，若第一位為0就跳到下個情況，否則全部輸出

```
pString new_line
pString output_lcm_mes ; prepare to print LCM

mov si, 03h      ; because memory is little-endian, print from index 3 2 1 to 0
mov ah, 02h

mov dl, result_decimal[si]
add dl, 30h      ; if first digit of LCM is 0, no need to print
cmp dl, 30h
je lcm_3digit
int 21h

dec si
mov dl, result_decimal[si]
add dl, 30h
int 21h

dec si
mov dl, result_decimal[si]
add dl, 30h
int 21h

dec si
mov dl, result_decimal[si]
add dl, 30h
int 21h
jmp exit_code
```


14.若第一位為0就進入此情況，若是第二位仍然為0，就跳到下一個情況，否則輸出2、3、4位

```
lcm_3digit:
    dec si
    mov dl, result_decimal[si]
    add dl, 30h      ; if second digit of LCM is 0, no need to print
    cmp dl, 30h
    je lcm_2digit
    int 21h

    dec si
    mov dl, result_decimal[si]
    add dl, 30h
    int 21h

    dec si
    mov dl, result_decimal[si]
    add dl, 30h
    int 21h
    jmp exit_code
```

15.若是第二位為0，就進入此情況，若第三位為0就跳到下一個情況，否則輸出3、4位

```
lcm_2digit:
    dec si
    mov dl, result_decimal[si]
    add dl, 30h      ; if third digit of LCM is 0, no need to print
    cmp dl, 30h
    je lcm_1digit
    int 21h

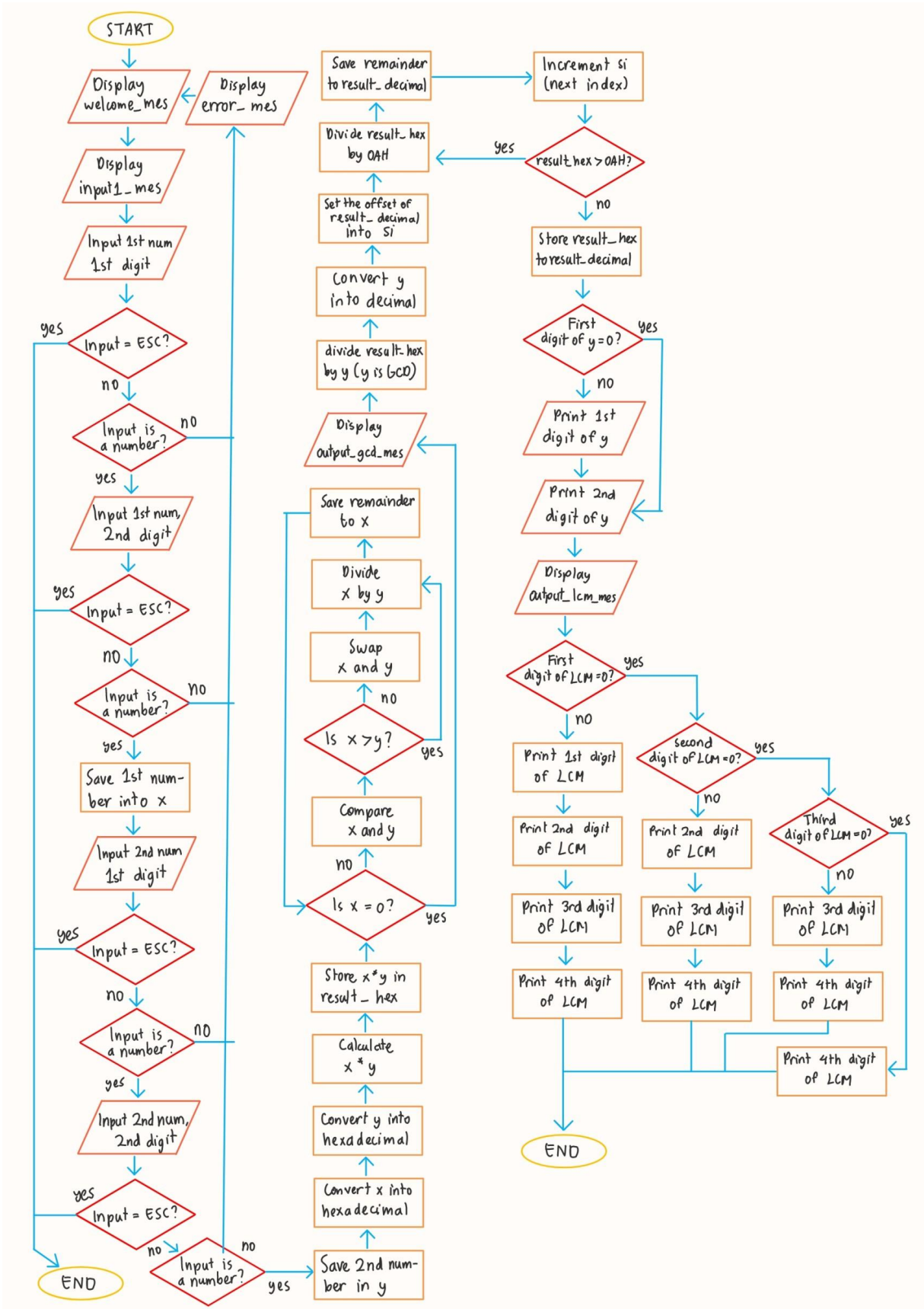
    dec si
    mov dl, result_decimal[si]
    add dl, 30h
    int 21h
    jmp exit_code
```

16.若前三位都為0就進入此情況，輸出最後一個位數

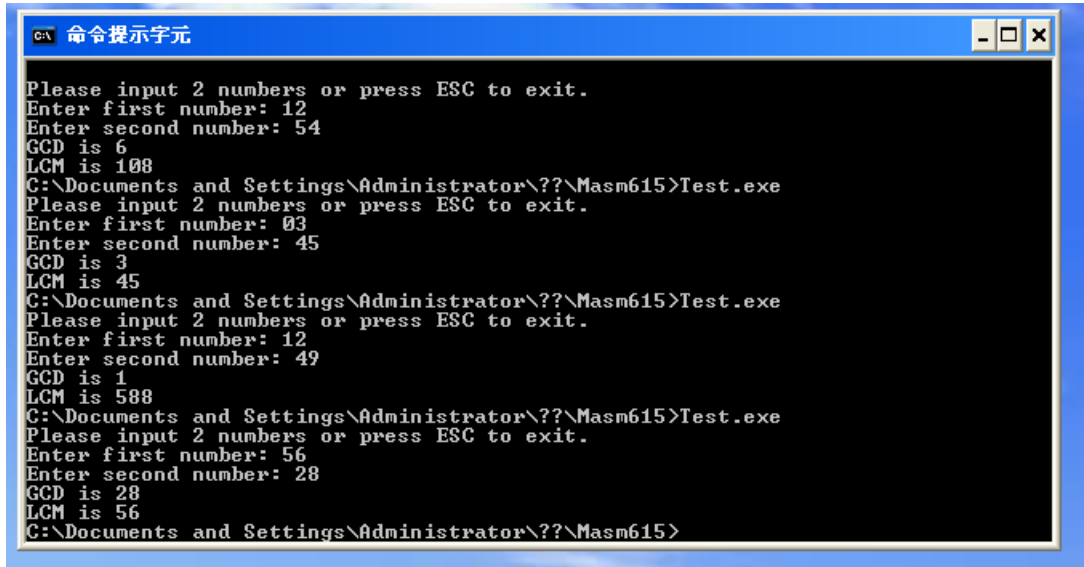
```
lcm_1digit:
    dec si
    mov dl, result_decimal[si]
    add dl, 30h      ; print last digit of LCM
    int 21h

exit_code:
    mov ax, 4c00h    ; code is done
    int 21h
main endp
end main
```

二、 流程圖



三、實習結果（輸入兩個數字後最大公因數與最小公倍數的小黑窗結果截圖）



```
CA 命令提示字元
Please input 2 numbers or press ESC to exit.
Enter first number: 12
Enter second number: 54
GCD is 6
LCM is 108
C:\Documents and Settings\Administrator\??\Masm615>Test.exe
Please input 2 numbers or press ESC to exit.
Enter first number: 03
Enter second number: 45
GCD is 3
LCM is 45
C:\Documents and Settings\Administrator\??\Masm615>Test.exe
Please input 2 numbers or press ESC to exit.
Enter first number: 12
Enter second number: 49
GCD is 1
LCM is 588
C:\Documents and Settings\Administrator\??\Masm615>Test.exe
Please input 2 numbers or press ESC to exit.
Enter first number: 56
Enter second number: 28
GCD is 28
LCM is 56
C:\Documents and Settings\Administrator\??\Masm615>
```

四、心得

- 黃偉智 (B11107035) :

我覺得這次的題目雖然很難，但是還蠻好玩的。我曾經多次做過同樣的問題，但是從來都沒有使用過組合語言來解決這個問題。透過這個期中作業，我學到了很多東西，包括怎麼如何將十進制數轉換為十六進制數，組合語言的除法與乘法，還有復習學過的輸入與輸出功能。希望這些新知識對未來有益。

- 韓承志 (B11107037) :

這次的微算機作業，結合了過往我們實習課所練習到的各個功能，包含從鍵盤讀取資料，存取資料，組合語

言的迴圈、乘法、除法，以及16進制跟10進制的轉換，要對於不同進制的理解外、如何利用迴圈找出最大公因數也是非常重要的部分，此外針對輸入的2個二位數如何從暫存器去存取等都是我們需要學習的，當然最後輸出結果的部分也對應到了剛開學時所練習的shr、shl