Multiplication and Division

Programming Assignment # 3

By

407261128

康智絜

X86 Assembly Language

Fall 2019

Date Submitted: November 18, 2019

**Test Plan :**

The greatest common divisor ( of two integers is the largest

integer that will evenly divide both integers The GCD algorithm

involves integer division in a loop, described by the following

pseudocode

int GCD(int x, int y)

{

x = abs(x)

x)// absolute value

y = abs(y)

do {

int n = x % y

x = y

y = n

} while (y > 0)

return x

}

|  |  |  |
| --- | --- | --- |
| **Test Case Number** | **Input Values** | **Expected output** |
| 1 | 10  24 | Enter a 32 bit number: 10  Enter a 32 bit number: 24  Greatest common divisor is: 2 |
| 2 | -100  48 | Enter a 32 bit number: -100  Enter a 32 bit number: 48  Greatest common divisor is: 4 |

**Feedback: (required)**

1. 在寫mod的時候，一開始看到網路上有很簡單的方法，x = x%8的話就把x AND 7,但試了很久發現答案都很奇怪，後來才發現這個方法只適合用在2^n的數字

2. 在試”0”這個測資時，發現沒有做意外狀況處理，所以後來有加上判斷X==0或Y==0的判斷

**Appendix A: Test Log (required)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case Number** | **Input Values** | **Actual Output** | **Result** |
| 1 | 10  24 | Enter a 32 bit number: 10  Enter a 32 bit number: 24  Greatest common divisor is: 2 | True |
| 2 | -100  48 | Enter a 32 bit number: -100  Enter a 32 bit number: 48  Greatest common divisor is: 4 | True |
| 3 | 3  0 | Enter a 32 bit number: 3  Enter a 32 bit number: 0  Greatest common divisor is: 0 | True |
| 4 | 0  0 | Enter a 32 bit number: 0  Enter a 32 bit number: 0  Greatest common divisor is: 0 | True |

**Appendix B: Source Code (required)**

INCLUDE Irvine32.inc

.data

InputText BYTE 'Enter a 32 bit number: ',0

GCDText BYTE 'Greatest common divisor is: ', 0

IsZeroText BYTE '0', 0

x DWORD ?

y DWORD ?

.code

main proc

mov edx, OFFSET InputText

call WriteString ; print 'Enter a 32 bit number: '

call readint

mov x, eax

mov edx, OFFSET InputText

call WriteString ; print 'Enter a 32 bit number: '

call readint

mov y, eax

CompareX:

cmp x, 0

JE IsZero ; if(x==0) jmp IsZero

JG CompareY ; if(x>0) jmp CompareY

mov eax, x ; else{ x \*= -1 }

mov ebx, -1

mul ebx

mov x, eax

jmp CompareY

CompareY:

cmp y, 0

JE IsZero ; if(y==0) jmp IsZero

JGE CalcGcd ; if(y>0) jmp CalcGcd

mov eax, y ; else{ y \*= -1 }

mov ebx, -1

mul ebx

mov y, eax

jmp CalcGcd

IsZero:

mov edx, OFFSET GCDText

call WriteString ; print 'Greatest common divisor is: '

mov edx, OFFSET IsZeroText

call WriteString

call crlf

jmp ExitTag

CalcGcd:

mov eax, x

XOR edx, edx ; clear the destinations for outputs. this stops the garbage remainder

mov ecx, y

div ecx

mov eax, edx ; eax = x % y

mov ebx, y

mov x, ebx ; x = y

mov y, eax ; y = x % y

cmp y, 0 ; while( y>0 ) do{ GCD }

JG CalcGcd

mov edx, OFFSET GCDText

call WriteString ; print 'Greatest common divisor is: '

mov eax, x

call WriteDec

call crlf

ExitTag:

call crlf

invoke ExitProcess,0

main endp

end main