Assembly Language Week11

第7組

104502517戴逸任

104502518劉冠聲

程式碼(Different Inputs)：

TITLE homework091 (homework091.asm)

INCLUDE Irvine32.inc

.386

.model flat,stdcall

.stack 4096

ExitProcess proto,dwExitCode:dword ;宣告 proto type

DifferentInputs proto, v1:DWORD,v2:DWORD ,v3:DWORD ;宣告 proto type

.data

.code

start@0 PROC

invoke DifferentInputs,2,2,3

invoke DifferentInputs,2,3,2

invoke DifferentInputs,3,5,5

invoke DifferentInputs,2,2,2

invoke DifferentInputs,104522064,104502517,104502518

call WaitMsg

invoke ExitProcess,0

start@0 ENDP

DifferentInputs proc,v1:dword, v2:dword, v3:dword

mov eax,v1 ; 取出 v1

cmp v2,eax ; 與v2比較

je Label\_Equal ; 若相等則跳到Label\_Equal,回傳0

cmp v3,eax ; 與v3做比較

je Label\_Equal ; 若相等則跳到Label\_Equal,回傳0

mov eax,v2 ; 取出 v2

cmp v3, eax ; 與v3做比較

je Label\_Equal ; 若相等則跳到Label\_Equal,回傳0

mov eax,1 ; 回傳1

jmp exit\_label ; return true

Label\_Equal:

mov eax,0 ; return false

exit\_label:

call DumpRegs

ret

DifferentInputs ENDP

END start@0

程式流程說明(Different Inputs)：

寫出一個function，內容是比較第一個數與第二個數，若是兩數相同，則傳0回去給暫存器，接著比較第一個數與第三個數，若是兩數相同，則傳0回去給暫存器，然後再比較第二個數與第三個數，若是兩數相同，則傳0回去給暫存器，反之，則傳1回去給暫存器，利用invoke呼叫function的方式，就可以藉由暫存器的值分辨三個數字是否相同

程式碼(String Remove)：

TITLE homework092 (homework092.asm)

INCLUDE Irvine32.inc

main EQU start@0

Str\_remove PROTO,

pStart:PTR BYTE,

nChars:DWORD

.data

target1 BYTE "104502517",0

target2 BYTE "104502518",0

target3 BYTE "999999999",0

.code

main PROC

invoke Str\_remove, ADDR target1, 5 ; 第一個字串移除5個字元

mov edx,OFFSET target1

call WriteString

call Crlf

mov ebx, OFFSET target2

invoke Str\_remove, ebx, 2 ; 第二個字串移除2個字元

mov edx,OFFSET target2

call WriteString

call Crlf

invoke Str\_remove, ADDR [target2+1], 15 ; 第二個字串移除超過字串長度的字元

mov edx,OFFSET target2

call WriteString

call Crlf

call WaitMsg

exit

main ENDP

Str\_remove PROC,

pStart:PTR BYTE, ; points to first character to delete

nChars:DWORD ; number of characters to delete

invoke Str\_length, pStart

mov ecx,eax ; 抓到字串長度存放在 ecx 給之後複製迴圈用

.IF nChars <= ecx ; 檢查移除字元是否超過字串大小

sub ecx,nChars ; 如果沒有超過，就剪掉移除的字元數量

.ENDIF

mov esi, pStart ; 設定複製來源字串的起始點

add esi, nChars ; 將起始點移到正確要複製的位置

mov edi, pStart ; 將目的起始點設定好

cld ; clear direction flag(forward)

rep movsb ; do the move

mov BYTE PTR [edi],0; insert new null byte

Exit\_proc:

ret

Str\_remove ENDP

END main

程式流程說明(String Remove)：

寫一個function，裡面的pStart指的是將要刪除字串的起始位置，nChars則是刪除字串數字個數，接著呼叫Str\_length得到此字串的長數，將得到的值存到ecx讓之後複製迴圈之使用，然後如果ecx的值大於nChars 就把ecx減掉nChars，這樣就能得到字串中要複製到前面的數量，接著就是設定複製來源的起始點，把起始點一到正確要複製的位置，然後就目地起始點設定好，接著使用rep movsb把字串以Byte一個一個複製過去，然後設定字串的結束字元為0 回到主程式部分，invoke這個function，第一個字串移除5個字元，然後第二個字串移除2個字元，最後一個invoke function，因為是把刪除後留下的部分往前移動，剩下的部分保持不變，然而當初宣告時是一起宣告的，所以記憶體是連在一起的，invoke時給的address是[target2+1]，因此是從複製完之後的第二個字開始刪除，然後再把宣告的target3按到所移除的字元數往前移動

心得：

這次一次出了兩個作業，原本想說會很多要打很久，才發現助教已經把大概的程式碼都完成了，剩下重要的細節部分，由我們填空，再配上講解很詳細的註解，總算是理解了程式的運行方式，順順利利完成了，而且老師當天教的東西，在作業裡馬上就會用到，讓我們能夠好好複習，記得更清楚，而在移除超出位數那邊，我們思考了很久到底是怎麼做的，後來用紙筆畫出記憶體狀況才終於了解在這方面處理的運作模式。