Seminar 6

1. Schreibe ein Modul, das eine Funktion enthält,die zwei Zeichenketten verkettet (concatenate two strings). Schreibe ein Hauptprogramm, das die Funktion aus dem anderen Modul aufruft und das Ergebnis auf dem Bildschirm ausdruckt.

|  |  |
| --- | --- |
| Main.asm | Function.asm |
| **bits 32** **global** **start**  **extern** **exit, printf, conc**  **import** **exit msvcrt.dll**  Import printf msvcrt.dll  **segment** **data use32 class=data**  **Sir1 db “a”,”b”,”c”**  **Len1 equ $-Sir1**  **Sir2 db “def”**  **Len2 equ $-Sir2**  **Sirrez times Len1+Len2+1 db 0**  **Segment code use32 class=code**  **Start:**  **Push dword Sir1**  **Push dword Len1**  **Push dword Sir2**  **Push dword Len2**  **Push dword Sirrez**  **Call conc**  **Add esp, 4\*5**  **Push dword Sirrez**  **Call [printf]**  **Add esp, 4\*1**    **Push dword 0**  **Call [exit]** | Bits 32  Global conc  Segment code use32 class=code  Conc:     |  |  | | --- | --- | | esp | Adresa de retur | | Esp+4 | Sirrez | | Esp+8 | len2 | | Esp+12 | sir2 | | Esp+16 | len1 | | Eps+20 | sir1 |   Mov ecx, [esp+16]  cld ; df=0  Mov esi, [esp+20]  Mov edi, [esp+4]  Rep movsb ; <ES:EDI> <- <DS:ESI>  Mov ecx, [esp+8]  Mov esi, [esp+12]  Rep movsb ;echivalent cu loop de ecx ori  ret |

2. Schreibe ein Programm, das die Summe der Ziffern einer Zahl ausdruckt, wobei das Programm eine Funktion aus einem anderen Modul aufruft, welche diese Summe berechnet.

|  |  |
| --- | --- |
| Main.asm | Function.asm |
| Bits 32  Global start  Extern exit, printf, suma\_cif  Import exit msvcrt.dll  Import printf msvcrt.dll  Segment data:  Nr dd 123456  Format db “%u”, 0  Suma dd 0  Segment code:  Start:  Push dword [Nr]  Push dword Suma  Call suma\_cif  Add esp, 4\*2  ESP ESP+4 ESP+8  Adr retur ~~adr retur~~ | format ~~suma~~ | Suma ~~nr~~ |  ;printf(format, suma)  Push dword [Suma]  Push dword format  Call [printf]  Add esp, 4\*2  Push dword 0  Call [exit] | Bits 32  Global suma\_cif  Segment code:   |  |  | | --- | --- | | esp | Adresa de retur | | Esp+4 | Suma - adr | | Esp+8 | nr |   ;  ESP | ESP+4 | ESP+8  adr retur | suma -dword | nr – dword (4 bytes) |  ;doubleword -> qword  Mov ebx, 0 ;suma cifrelor  Mov edx, 0  Mov eax, [esp+8] ;edx:eax = nr  ...  ; adunam in ebx restul (EDX)  …  ;Mov [esp+4], ebx ->incerc sa suprascriu ce am in stiva  ;La adresa suma ar trebui sa zic mov [[esp+4]]  Mov esi, **[esp+4]** ;esi = adresa variabilei suma  Mov **[esi]**, ebx  ret |
|  |  |