Seminar 6

Programare multi-modul

Transmiterea parametrilor unei proceduri

1. prin intermediul registrilor

x = a/b, a – dublucuvant, b – cuvant, fs

main.asm

bits 32

global start

import exit msvcrt.dll

extern exit

extern impartire

segment data use32 class=data public

a dd 12345

b db 315

x resw 1

segment code use32 class=code public

start

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

1. Scrieti un program in limbaj de asamblare care concateneaza doua siruri prin apelul unei functii scrise intr-un modul secundar si apoi va afisa rezultatul pe ecran.

main.asm

extern concatenare

segement data ... public

a db ‘a’, ‘b’, ‘c’

la equ $-a

b db ‘e’, ‘f’, ‘g’, ‘h’

lb equ $-b

c times la+lb+1 db 0

segment code ... public

start:  
push dword a

push dword la

push dword b

push dword lb

push dword c

|  |  |
| --- | --- |
| adresa de revenire | <- esp |
| adresa c | <- esp + 4 |
| lb | <- esp + 8 |
| adresa b | <- esp + 12 |
| la | <- esp + 16 |
| adresa a | <- esp + 20 |

call concatenare

push dword c

call [printf]

add esp, 4\*1

modul.asm

global concatenare

segment code ... public

concatenare:

mov esi, [esp+20]

mov edi, [esp+4]

mov ecx. [esp+16]

repeta:

mov al, [esi]

mov [edi], al

inc esi

inc edi

loop repeta

f1:

mov esi, [esp+12]

cld

mov ecx,[esp+8]

jecxz f2

re[ ,pvsb

f2:

ret 4\*5

1. Se da un numar repereznetat pe 32biti fara semn. Sa se afiseze suma cifrelor lui.

main.asm

extern suma

segment data ... public

a db 21357

format db “%d”, 0

s dd 0

segment code ... public

start:  
 push dword [a]

call suma

add esp, 4\*1

mov s, eax

modul.asm

global suma

segment code... public

suma:

|  |  |
| --- | --- |
| adresa de revenire | <- esp |
| val a | <- esp+4 |

mov eax, [esp+4]

mov edx, 0

mov ebx, 10

mov ecx, 0

repeta:

div ebx

add ecx, edx

mov edx, 0

cmp eax, 0

jnz repeta

mov eax, ecx

ret

1. Se citeste un cuvant (sir de caractere) de la tastatura. Criptati acest cuvant adunand 20 la fiecare caracter si afisati pe ecran rezultatul.

main.asm

extern criptare

segment data ... public

a times 11 db 0

format db “%s”, 0

segment code ... public

start:

push a

push format

call [scanf]

add esp, 4\*2

push a

call criptare

add esp, 4\*1

push a

call [printf]

add esp, 4\*1

|  |  |
| --- | --- |
| adresa de revenire | <- esp |
| adresa lui a | <-esp + 4 |

modul.asm

global criptare

segment code ... public

repeta:

mov ebx, [esp+4]

cmp ebx, 0

je final

add byte[ebx], 20

inc ebx

jmp repeta

final:

ret

1. Se dau doua siruri de caractere. Sa se afiseze sirul cu numar maxim de caractere speciale (orice caracter care nu este cifra sau litera este considerat caracter special)

main.asm

extern count

segment data ... public

sir1 db “abx120\*”, 0

sir2 db “cd#?a!”, 0

sirn db “abcdefghjiklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890”

segment code ... public

start:

push dword sir1

push dword sirn

call count

add esp, 4\*2

push eax

push dword sir2

push dword sirn

call count

add esp, 4\*2

push eax

pop eax

pop ebx

cmp eax, ebx

ja afiseazas2

push dword s1

call [printf]

add esp, 4\*1

afiseazas2:

push dword sir2

call [printf]

add esp, 4\*1

|  |  |
| --- | --- |
| adresa de revenire | <-esp |
| adresa sirn | <- esp + 4 |
| adresa sir1 | <- esp + 8 |

modul.asm

...

count:

mov eax, 0

mov esi, [esp+8]

repeta:

mov edi, [esp+4]

mov bl, [esi]

cmp al, 0

je afara

repeta2:

mov bh, edi