**Laboratorul 6 și 8**

;1. Se da un sir de dublucuvinte continand date impachetate (4 octeti scrisi ca un singur dublucuvant). Sa se obtina un nou sir ;de dublucuvinte, in care fiecare dublucuvant se va obtine dupa regula: suma octetilor de ordin impar va forma cuvantul de

;ordin impar, iar suma octetilor de ordin par va forma cuvantul de ordin par. Octetii se considera numere cu semn, astfel ca ;extensiile pe cuvant se vor realiza corespunzator aritmeticii cu semn.

; Exemplu:

; pentru sirul initial:

; 127F5678h, 0ABCDABCDh, ...

; Se va obtine:

; 006800F7h, 0FF56FF9Ah

bits 32

global start

extern exit, printf

import exit msvcrt.dll

import printf msvcrt.dll

segment data use32 class=data

sir dd 127F5678h, 0ABCDABCDh

len equ ($-sir)\*4

rez times len db 0

segment code use32 class=code

start:

mov ecx, len

jecxz final

cld

mov esi, sir

mov edi, rez

repeta1:

lodsb

mov bl,al

lodsb

mov cl,al

lodsb

add bl,al

lodsb

add cl,al

mov al,bl

cbw

mov bx,ax

mov al,cl

cbw

mov cx,ax

push bx

push cx

pop eax

stosd

loop repeta1

final:

push dword 0

call [exit]

; 19.Dandu-se doua siruri de octeti sa se calculeze toate pozitiile unde al doilea sir apare ca subsir in primul sir.

bits 32

global start

extern exit

import exit msvcrt.dll

segment data use32 class=data

s1 db 12h, 13h, 14h, 20h, 21h, 12h, 13h, 14h, 18h, 19h, 12h, 13h

lgs1 equ $-s1

s2 db 12h, 13h, 14h

lgs2 equ $-s1-s2

d times 10 db 0

segment code use32 class=code

start:

mov ecx,lgs1

mov esi,0

mov edi,d

rezolvare:

push ecx

mov al,[s1+esi]

mov dl,[s2]

cmp al,dl

je parcurg2

jmp peste

parcurg2:

mov ecx, lgs2

dec ecx

mov ebx,1

inc esi

forj:

mov al,[s1+esi]

mov dl,[s2+ebx]

cmp al,dl

jne peste

cmp esi,lgs1

ja peste

inc ebx

inc esi

loop forj

mov eax,esi

sub eax,ebx

stosb

peste:

pop ecx

inc esi

loop rezolvare

push dword 0

call [exit]

; 29.Se dau doua siruri de octeti. Sa se parcurga cel mai scurt sir dintre cele doua siruri si sa se construiasca un al treilea

; sir care va contine cel mai mare element de acelasi rang din cele doua siruri, iar pana la lungimea celui mai lung sir, sirul

; al treilea se va completa alternativ cu valoarea 1 respectiv 0.

bits 32

global start

extern exit

import exit msvcrt.dll

segment data use32 class=data

s1 db 1,2,3,4,8,9,10,11

lgs1 equ $-s1

s2 db 2,1,5,3

lgs2 equ $-s1-s2

s3 times 100 db 0

poz db 0

segment code use32 class=code

start:

mov edx,1

mov ecx,lgs1

mov byte[poz],lgs2

mov ebx,lgs2

cmp ecx,ebx

jb rezv

mov ecx,lgs2

mov byte[poz],lgs1

mov edx,0

rezv:

mov esi,0

mov ebx,0

mov edi,s3

cmp edx,1

jne rezolv2

rezolv1:

mov al,[s1+esi]

mov dl,[s2+ebx]

cmp al,dl

jb ads2

jmp fin1

ads2:

mov al,[s2+ebx]

fin1:

stosb

inc esi

inc ebx

loop rezolv1

jmp final

rezolv2:

sir2:

mov al,[s2+esi]

mov dl,[s1+ebx]

cmp al,dl

jb ads1

jmp fin2

ads1:

mov al,[s1+esi]

fin2:

stosb

inc esi

inc ebx

loop rezolv2

final:

cmp ebx,[poz]

ja finalt

mov al,0

repeta10:

stosb

cmp al,0

je unu

jne zero

unu:

mov al,1

jmp p

zero:

mov al,0

p:

inc ebx

cmp ebx,[poz]

jb repeta10

finalt:

push dword 0

call [exit]

; 30.Se da un sir de cuvinte. Sa se construiasca doua siruri de octeti, s1 si s2, astfel: pentru fiecare cuvant,

; daca numarul de biti 1 din octetul high al cuvantului este mai mare decat numarul de biti 1 din octetul low,

; atunci s1 va contine octetul high, iar s2 octetul low al cuvantului

; daca numarul de biti 1 din cei doi octeti ai cuvantului sunt egali, atunci s1 va contine numarul de biti 1 din octet,

; iar s2 valoarea 0

; altfel, s1 va contine octetul low, iar s2 octetul high al cuvantului.

bits 32

global start

extern exit

import exit msvcrt.dll

segment data use32 class=data

s dw 1234h,1000h,2000h

len equ ($-s)/2

nrc db 0

s1 times 10 db 0

s2 times 10 db 0

segment code use32 class=code

nroctet:

pushad

mov byte[nrc],0

mov bl,al

mov ecx,8

rep1:

shr bl,1

jnc sari

add byte[nrc],1

sari:

loop rep1

popad

ret

start:

mov ecx,len

mov esi,s

mov edi,0

rezolvare:

push ecx

cld

lodsb

call nroctet

mov bl,[nrc]

cld

lodsb

call nroctet

mov dl,[nrc]

cmp bl,dl

jb maimic

ja maimare

mov [s1+edi],dl

mov cl,0

mov [s2+edi],cl

jmp peste

maimic:

mov [s1+edi],dl

mov [s2+edi],bl

jmp peste

maimare:

mov [s1+edi],bl

mov [s2+edi],dl

peste:

inc edi

pop ecx

loop rezolvare

push dword 0

call [exit]

; 6.Se da un sir de cuvinte s. Sa se construiasca sirul de octeti d, astfel incat d sa contina pentru fiecare pozitie din s:

; - numarul de biti de 0, daca numarul este negativ

; - numarul de biti de 1, daca numarul este pozitiv

; Exemplu:

; s: -22, 145, -48, 127

; in binary:

; 1111111111101010, 10010001, 1111111111010000, 1111111

; d: 3, 3, 5, 7

bits 32

global start

extern exit

import exit msvcrt.dll

segment data use32 class=data

s dw -22, 145, -48, 127

lg equ ($-s)/2

d times lg db 0

segment code use32 class=code

start:

mov esi,0

mov edi,d

mov ecx,lg

rezolvare:

push ecx

mov dx,[s+esi]

cmp dx,0

jl nrzero

jg nrunu

nrzero:

mov eax,0

mov ecx,16

nrz:

shr dx,1

jnc adaugaz

inap:

loop nrz

jmp next

adaugaz:

inc al

jmp inap

nrunu:

mov eax,0

mov ecx,16

nru:

shr dx,1

jc adaugau

inapoi:

loop nru

jmp next

adaugau:

inc al

jmp inapoi

next:

stosb

pop ecx

inc esi

inc esi

loop rezolvare

push dword 0

call [exit]

; 22.Se citesc de la tastatura doua numere a si b.

; Sa se calculeze valoarea expresiei (a+b)\*k, k fiind o constanta definita in segmentul de date.

; Afisati valoarea expresiei (in baza 10).

bits 32

global start

extern exit, printf, scanf

import exit msvcrt.dll

import printf msvcrt.dll

import scanf msvcrt.dll

segment data use32 class=data

a dw 0

b dw 0

k equ 2

mesaja db "Dati a=",0

mesajb db "Dati b=",0

format db "%d",0

formats db "%s",0

formath db "%x",0

contor dd 0

nu db "Nu",0

da db "Da",0

suma dd 0

dif dd 0

rez dd 0

segment code use32 class=code

start:

;apelez printf(mesaja)

push dword mesaja

call [printf]

add esp,4\*1

;apelez scanf(format,a)

push dword a

push dword format

call [scanf]

add esp,4\*2

;apelez printf(mesajb)

push dword mesajb

call [printf]

add esp,4\*1

;apelez scanf(format,b)

push dword b

push dword format

call [scanf]

add esp,4\*2

mov eax,0

mov ax,[a]

add ax,[b]

mov bx,k

mul bx

push dword eax

push dword formath

call [printf]

add esp,4\*2

push dword 0

call [exit]

; 19.Sa se citeasca de la tastatura un octet si un cuvant. Sa se afiseze pe ecran daca bitii octetului citit se regasesc consecutiv

; printre bitii cuvantului. Exemplu:

; a = 10 = 0000 1010b

; b = 256 = 0000 0001 0000 0000b

; Pe ecran se va afisa NU.

; a = 0Ah = 0000 1010b

; b = 6151h = 0110 0001 0101 0001b

; Pe ecran se va afisa DA (bitii se regasesc pe pozitiile 5-12).

bits 32

global start

extern exit, printf, scanf

import exit msvcrt.dll

import printf msvcrt.dll

import scanf msvcrt.dll

segment data use32 class=data

a db 0

b dw 0

mesaja db "Dati a=",0

mesajb db "Dati b=",0

format db "%d",0

formats db "%s",0

formath db "%x",0

contor dd 0

nu db "Nu",0

da db "Da",0

segment code use32 class=code

start:

;apelez printf(mesaja)

push dword mesaja

call [printf]

add esp,4\*1

;apelez scanf(format,a)

push dword a

push dword formath

call [scanf]

add esp,4\*2

;apelez printf(mesajb)

push dword mesajb

call [printf]

add esp,4\*1

;apelez scanf(format,b)

push dword b

push dword formath

call [scanf]

add esp,4\*2

mov ax,[b]

mov ecx,8

mov dl,[a]

mov ebx,0

rezv:

cmp al,dl

je peste

shr al,1

loop rezv

jmp sari

peste:

mov ebx,1

sari:

cmp ebx,1

jne nunu

push dword da

push dword formats

call [printf]

add esp,4\*2

jmp p

nunu:

push dword nu

push dword formats

call [printf]

add esp,4\*2

p:

push dword 0

call [exit]

; 21.Sa se citeasca de la tastatura doua numere a si b de tip word. Sa se afiseze in baza 16 numarul c de tip dword pentru care

; partea low este suma celor doua numere, iar partea high este diferenta celor doua numere. Exemplu:

; a = 574, b = 136

; c = 01B602C6h

bits 32

global start

extern exit, printf, scanf

import exit msvcrt.dll

import printf msvcrt.dll

import scanf msvcrt.dll

segment data use32 class=data

a dw 0

b dw 0

mesaja db "Dati a=",0

mesajb db "Dati b=",0

format db "%d",0

formats db "%s",0

formath db "%x",0

contor dd 0

nu db "Nu",0

da db "Da",0

suma dd 0

dif dd 0

rez dd 0

segment code use32 class=code

start:

;apelez printf(mesaja)

push dword mesaja

call [printf]

add esp,4\*1

;apelez scanf(format,a)

push dword a

push dword format

call [scanf]

add esp,4\*2

;apelez printf(mesajb)

push dword mesajb

call [printf]

add esp,4\*1

;apelez scanf(format,b)

push dword b

push dword format

call [scanf]

add esp,4\*2

mov ax,[a]

add ax,[b]

mov edi,rez

cld

stosw

mov ax,[a]

sub ax,[b]

cld

stosw

push dword [rez]

push dword formath

call [printf]

add esp,4\*2

push dword 0

call [exit]

; 6.Se da un fisier text. Sa se citeasca continutul fisierului, sa se determine cifra cu cea mai mare frecventa si sa se afiseze

; acea cifra impreuna cu frecventa acesteia. Numele fisierului text este definit in segmentul de date.

bits 32

global start

extern exit, printf, scanf, fopen, fscanf

import exit msvcrt.dll

import printf msvcrt.dll

import scanf msvcrt.dll

import fopen msvcrt.dll

import fscanf msvcrt.dll

segment data use32 class=data

lg equ $-voc

nume\_fisier db "test.txt",0

mod\_acces db "r",0

descriptor\_fis dd -1

len equ 100

text times len db 0

numar dd 0

format\_citire db "%d",0

contor dd 0

maxim dd 0

format db "Maxim este %d de %d",0

segment code use32 class=code

start:

;eax=fopen(nume\_fisier,mod\_acces)

push dword mod\_acces

push dword nume\_fisier

call [fopen]

add esp,4\*2

mov [descriptor\_fis],eax

cmp eax,0

je final

repeta:

;eax=fscanf(caracter,format\_citire,descriptor\_fis)

push dword numar

push dword format\_citire

push dword [descriptor\_fis]

call [fscanf]

add esp, 4\*3

cmp eax,1

jne final

mov ebx,[numar]

cmp ebx,[maxim]

jb sari

mov dword[maxim],ebx

add dword[contor],1

sari:

jmp repeta

final:

;fclose(descriptor\_fis)

push dword [contor]

push dword [maxim]

push dword format

call [printf]

add esp,4\*2

push dword 0

call [exit]

; 27.Se da un fisier text. Fisierul contine numere (in baza 10) separate prin spatii. Sa se citeasca continutul acestui fisier,

; sa se determine minimul numerelor citite si sa se scrie rezultatul la sfarsitul fisierului.

bits 32

global start

extern exit,fopen,fclose,fscanf,fprintf,printf

import exit msvcrt.dll

import fopen msvcrt.dll

import fclose msvcrt.dll

import fscanf msvcrt.dll

import printf msvcrt.dll

import fprintf msvcrt.dll

segment data use32 class=data

fisier db "lab8ex27.txt",0

mod\_acces db "a+",0

descriptor dd -1

nr dd 0

format db "%d",0

minim dd 100

fspatiu db 13,10,"%d",0

segment code use32 class=code

start:

push dword mod\_acces

push dword fisier

call [fopen]

add esp,4\*2

mov [descriptor],eax

cmp eax,0

je final

repeta:

push dword nr

push dword format

push dword [descriptor]

call [fscanf]

cmp eax,1

jne f

mov ebx,[nr]

cmp ebx,[minim]

jae peste

mov dword[minim],ebx

peste:

jmp repeta

f:

push dword [minim]

push dword fspatiu

push dword [descriptor]

call [fprintf]

add esp,4\*2

push dword[descriptor]

call [fclose]

final:

push dword 0

call [exit]