Laborator 9 - Suport teoretic

Operatii cu fisiere text

Operatii cu fisiere text

Un fisier reprezinta o secventa de octeti. Pentru a citi dintr-un fisier sau pentru a scrie intr-un fisier, e nevoie de 3 pasi:

1. Deschiderea fisierului, care poate consta in:
    - Deschiderea unui fisier existent
    - Crearea unui fisier nou
2. Efectuarea operatiilor de scriere si/sau citire
3. Inchiderea fisierului.

Deschiderea unui fisier

Pentru deschiderea unui fisier existent sau crearea unui fisier nou, se foloseste functia **fopen**

Sintaxa functiei

fopen in limbaj de programare de nivel inalt este:

```
FILE * fopen(const char* nume_fisier, const char * mod_acces)
```

Functia fopen respecta conventia *cdecl* si se gaseste in *msvcrt.dll* .

Argumentele functiei fopen:

Primul argument al functiei este adresa unui sir de caractere reprezentand numele fisierului: Al doilea argument este adresa unui sir de caractere, reprezentand modalitatea in care se va deschide fisierul.

Mod	Semnificatie	Descriere
r	citire (read)	- Deschide un fisier text pentru citire. Fisierul trebuie sa existe deja pe disc.
w	scriere (write)	- Daca nu exista un fisier cu acel nume, creaza fisierul si il deschide pentru scriere. - Daca un fisier cu acel nume exista deja, deschide acel fisier pentru scriere. Continutul initial va fi sters. Scrierea se va face de la inceputul fisierului.
a	adaugare (append)	- Daca nu exista un fisier cu acel nume, creaza fisierul si il deschide pentru scriere. - Daca un fisier cu acel nume exista deja, deschide acel fisier pentru scriere,

		dar scrierea se va face la sfarsitul fisierului, in continuarea continutului existent. Continutul initial al fisierului va fi pastrat.
r+	citire si scriere fisier existent	- Deschide un fisier text pentru citire si scriere. Fisierul trebuie sa existe deja pe disc.
w+	citire si scriere	- Daca nu exista un fisier cu acel nume, creaza fisierul si il deschide pentru citire si scriere. - Daca un fisier cu acel nume exista deja, deschide acel fisier pentru citire si scriere. Continutul initial va fi sters. Scrierea se va face de la inceputul fisierului.
a+	citire si adaugare	- Daca nu exista un fisier cu acel nume, creaza fisierul si il deschide pentru citire si scriere. - Daca un fisier cu acel nume exista deja, deschide acel fisier pentru citire si scriere. Continutul initial al fisierului va fi pastrat. Citirea se va face de la inceputul fisierului. Scrierea se va face in continuarea continutului existent.

Observatii:

- Numele unui fisier trebuie sa includa si extensia (ex: nume.txt, exemplu.asm).
- Fisierele se vor crea sau deschide din directorul curent (in acelasi director in care se afla fisierul sursa asm). Important: pentru a putea deschide un fisier existent folosind numele acestuia, fisierul trebuie sa se afle in acelasi director cu fisierul sursa .asm, altfel acesta nu va fi gasit.
- Operatiile de scriere nu vor reusi pentru fisiere deschise doar cu drepturi de citire (ex: ”r”). Operatiile de citire nu vor reusi pentru fisiere deschise doar cu drepturi de scriere sau adaugare (ex: ”w”, ”a”)
- Ambele argumente ale functiei *fopen* reprezinta siruri de caractere care trebuie sa se termine cu valoarea 0 (asemenea formatului pentru functia printf).

Valoarea returnata de functia fopen:

Daca fisierul este deschis cu succes, functia fopen va completa in registrul EAX un identificator (descriptor de fisier) care va fi folosit in continuare pentru a lucra cu acel fisier (pentru operatii de scriere, citire, etc.). Altfel (in caz de eroare), functia fopen va completa in registrul EAX valoarea 0.

Alte observatii:

Este important sa se verifice valoarea returnata de functie in EAX (daca nu a fost eroare), inainte de a efectua alte operatii cu acel fisier. Daca in cadrul unui program se deschid mai multe fisiere diferite folosind functia fopen, fiecare valoare returnata de functiei trebuie salvata separat, deoarece reprezinta o valoare distincta prin care este identificat un fisier. Dupa finalizarea lucrului cu un fisier deschis, este important sa se si inchida acel fisier (de obicei se face la finalul programului – inainte de exit). Pentru a inchide un fisier (deschis in prealabil de functia fopen) se foloseste functia *fclose*.

Scrierea intr-un fisier

Pentru a scrie un text intr-un fisier se foloseste functia **fprintf**.

Sintaxa functiei

fprintf in limbaj de programare de nivel inalt este:

```
int fprintf(FILE * stream, const char * format, <variabila_1>, <constanta_2>, <...>)
```

Functia fprintf respecta conventia *cdecl* si se gaseste in *msvcrt.dll* . Sintaxa functiei *fprintf* este asemantoare cu sintaxa functiei *printf* (folosita pentru afisare in consola). Diferenta este faptul ca functia fprintf are ca prim parametru identificatorul fisierului in care se va scrie textul, in plus fata de parametrii functiei printf.

Argumentele functiei fprintf

Primul argument al functiei reprezinta descriptorul de fisier (identificatorul) returnat de apelul functiei *fopen*. Urmatorul argument al functiei este un sir de caractere ce contine formatul afisarii, urmat de un numar de argumente (valori constante sau nume de variabile) egal cu cel specificat in cadrul formatului. Asemenea functiei printf, sirul de caractere transmis in parametrul *format* poate contine anumite marcaje de formatare, ce incep cu caracterul '%', care vor fi inlocuite de valorile specificate in urmatoarele argumente, formatate corespunzator.

Specificator	Ce se afiseaza	Exemplu	Dimensiune reprezentare valoare
c	Caracter	a	byte
d sau i	Intreg zecimal cu semn	392	dword
u	Intreg zecimal fara semn	7235	dword
x	Numar in hexazecimal fara semn	7fa	dword
s	String (sir de caractere, terminat cu 0)	exemplu	sir de bytes terminat in 0

Citirea dintr-un fisier

Pentru a citi un text dintr-un fisier se foloseste functia **fread**.

Sintaxa functiei

fread in limbaj de programare de nivel inalt este:

```
int fread(void * str, int size, int count, FILE * stream)
```

Functia fread respecta conventia *cdecl* si se gaseste in *msvcrt.dll* .

Argumentele functiei fread

Primul argument al functiei fread reprezinta adresa unui sir de elemente in care se vor completa datele citite din fisier. Al doilea argument reprezinta dimensiunea unui element care va fi citit din fisier. Al treilea argument reprezinta numarul maxim de elemente care se vor citi din fisier. Ultimul argument al functiei reprezinta descriptorul de fisier (identificatorul) returnat de apelul functiei *fopen*. In cazul citirii fisierelor text, primul argument al functiei fread este un sir de bytes si al doilea argument este 1 (=dimensiunea unui byte). Al treilea argument este dimensiunea sirului de bytes (numarul de elemente).

Valoarea returnata de functia fread:

Functia fread va completa in registrul EAX numarul de elemente citite. Daca acest numar este mai mic decat valoarea argumentului *count*, atunci fie apelul functiei fread a intampinat o eroare la citire, fie s-a ajuns la finalul fisierului.

Observatii:

Fisierele text pot fi avea dimensiuni prea mari pentru putea citi continutul acestora cu un singur apel al functiei fread. In acest caz este nevoie de apeluri repetate ale functiei fread, pana cand intreg continutul fisierului este citit. In sectiunea „Exemple" vom prezenta un program care exemplifica acest scenariu. Pentru a verifica daca s-a ajuns la finalul fisierului cu operatia de citire se poate verifica daca valoarea returnata de fread este 0.

Inchiderea unui fisier deschis

Dupa finalizarea lucrului cu un fisier deschis, acesta trebuie inchis. Acest pas nu trebuie sa lipseasca dintr-un program care a deschis fisiere. Pentru inchiderea unui fisier se foloseste functia **fclose**.

Sintaxa functiei

fclose in limbaj de programare de nivel inalt este:

```
int fclose(FILE * descriptor)
```

Functia fclose respecta conventia *cdecl* si se gaseste in *msvcrt.dll* .

Argumentul functiei fclose

Argumentul functiei fclose este descriptorul de fisier (identificatorul) returnal de apelul functiei *fopen*.

Laborator 9 - Exemple

Operatii cu fisiere text

Crearea unui fisier nou

Fisier asm: create_file.asm

```
; Codul de mai jos va crea un fisier gol, numit "ana.txt" in directorul
curent.
; Programul va folosi functia fopen pentru deschiderea/crearea fisierului si
functia fclose pentru inchiderea fisierului creat.
; Deoarece in apelul functiei fopen programul foloseste modul de acces "w",
daca un fisier cu acelasi nume exista deja in directorul curent, continutul
acelui fisier va fi sters. Detalii despre modurile de acces sunt prezentate in
sectiunea "Suport teoretic"

bits 32

global start

; declare external functions needed by our program
extern exit, fopen, fclose
import exit msvcrt.dll
import fopen msvcrt.dll
import fclose msvcrt.dll

; our data is declared here (the variables needed by our program)
segment data use32 class=data
    nume_fisier db "ana.txt", 0  ; numele fisierului care va fi creat
    mod_acces db "w", 0          ; modul de deschidere a fisierului -
                                 ; w - pentru scriere. daca fiserul nu exista,
se va crea
    descriptor_fis dd -1         ; variabila in care vom salva descriptorul
fisierului - necesar pentru a putea face referire la fisier

; our code starts here
segment code use32 class=code
    start:
        ; apelam fopen pentru a crea fisierul
        ; functia va returna in EAX descriptorul fisierului sau 0 in caz de
eroare
        ; eax = fopen(nume_fisier, mod_acces)
        push dword mod_acces
        push dword nume_fisier
        call [fopen]
        add esp, 4*2                ; eliberam parametrii de pe stiva

        mov [descriptor_fis], eax   ; salvam valoarea returnata de fopen in
variabila descriptor_fis
```

```
        ; verificam daca functia fopen a creat cu succes fisierul (daca EAX !=
0)
        cmp eax, 0
        je final

        ; apelam functia fclose pentru a inchide fisierul
        ; fclose(descriptor_fis)
        push dword [descriptor_fis]
        call [fclose]
        add esp, 4

      final:

        ; exit(0)
        push    dword 0
        call    [exit]
```

Scrierea unui text intr-un fisier nou

Fisier asm: create_write_file.asm

```
; Codul de mai jos va crea un fisier numit "ana.txt" in directorul curent si
va scrie un text in acel fisier.
; Programul va folosi functia fopen pentru deschiderea/crearea fisierului,
functia fprintf pentru scrierea in fisier si functia fclose pentru inchiderea
fisierului creat.
; Deoarece in apelul functiei fopen programul foloseste modul de acces "w",
daca un fisier cu acelasi nume exista deja in directorul curent, continutul
acelui fisier va fi suprascris. Detalii despre modurile de acces sunt
prezentate in sectiunea "Suport teoretic"

bits 32

global start

; declare external functions needed by our program
extern exit, fopen, fprintf, fclose
import exit msvcrt.dll
import fopen msvcrt.dll
import fprintf msvcrt.dll
import fclose msvcrt.dll

; our data is declared here (the variables needed by our program)
segment data use32 class=data
    nume_fisier db "ana.txt", 0  ; numele fisierului care va fi creat
    mod_acces db "w", 0          ; modul de deschidere a fisierului -
                                 ; w - pentru scriere. daca fiserul nu exista,
se va crea
    text db "Ana are mere.", 0    ; textul care va fi scris in fisier

    descriptor_fis dd -1         ; variabila in care vom salva descriptorul
fisierului - necesar pentru a putea face referire la fisier
```

```
; our code starts here
segment code use32 class=code
    start:
        ; apelam fopen pentru a crea fisierul
        ; functia va returna in EAX descriptorul fisierului sau 0 in caz de
eroare
        ; eax = fopen(nume_fisier, mod_acces)
        push dword mod_acces
        push dword nume_fisier
        call [fopen]
        add esp, 4*2                ; eliberam parametrii de pe stiva

        mov [descriptor_fis], eax   ; salvam valoarea returnata de fopen in
variabila descriptor_fis

        ; verificam daca functia fopen a creat cu succes fisierul (daca EAX !=
0)
        cmp eax, 0
        je final

        ; scriem textul in fisierul deschis folosind functia fprintf
        ; fprintf(descriptor_fis, text)
        push dword text
        push dword [descriptor_fis]
        call [fprintf]
        add esp, 4*2

        ; apelam functia fclose pentru a inchide fisierul
        ; fclose(descriptor_fis)
        push dword [descriptor_fis]
        call [fclose]
        add esp, 4

      final:

        ; exit(0)
        push    dword 0
        call    [exit]
```

Adaugarea unui text intr-un fisier

Fisier asm: create_append_file.asm

```
; Codul de mai jos va deschide un fisier numit "ana.txt" in directorul curent
si va adauga un text la finalul acelui fisier.
; Programul va folosi functia fopen pentru deschiderea/crearea fisierului,
functia fprintf pentru scrierea in fisier si functia fclose pentru inchiderea
fisierului creat.
; Deoarece in apelul functiei fopen programul foloseste modul de acces "a",
daca un fisier cu numele exista deja in directorul curent, la continutul
acelui fisier se va adauga un text. Daca fisierul cu numele dat nu exista, se
va crea un fisier nou cu acel nume si se va scrie textul in fisier. Detalii
despre modurile de acces sunt prezentate in sectiunea "Suport teoretic"

bits 32
```

```
global start

; declare external functions needed by our program
extern exit, fopen, fprintf, fclose
import exit msvcrt.dll
import fopen msvcrt.dll
import fprintf msvcrt.dll
import fclose msvcrt.dll

; our data is declared here (the variables needed by our program)
segment data use32 class=data
    nume_fisier db "ana.txt", 0  ; numele fisierului care va fi creat
    mod_acces db "a", 0          ; modul de deschidere a fisierului -
                                 ; a - pentru adaugare. daca fiserul nu
exista, se va crea
    text db "Ana are si pere.", 0 ; textul care va fi adaugat in fisier

    descriptor_fis dd -1         ; variabila in care vom salva descriptorul
fisierului - necesar pentru a putea face referire la fisier

; our code starts here
segment code use32 class=code
    start:
        ; apelam fopen pentru a crea fisierul
        ; functia va returna in EAX descriptorul fisierului sau 0 in caz de
eroare
        ; eax = fopen(nume_fisier, mod_acces)
        push dword mod_acces
        push dword nume_fisier
        call [fopen]
        add esp, 4*2                ; eliberam parametrii de pe stiva

        mov [descriptor_fis], eax   ; salvam valoarea returnata de fopen in
variabila descriptor_fis

        ; verificam daca functia fopen a creat cu succes fisierul (daca EAX !=
0)
        cmp eax, 0
        je final

        ; adaugam/scriem textul in fisierul deschis folosind functia fprintf
        ; fprintf(descriptor_fis, text)
        push dword text
        push dword [descriptor_fis]
        call [fprintf]
        add esp, 4*2

        ; apelam functia fclose pentru a inchide fisierul
        ; fclose(descriptor_fis)
        push dword [descriptor_fis]
        call [fclose]
        add esp, 4

      final:

        ; exit(0)
```

```
        push    dword 0
        call    [exit]
```

Citirea unui text scurt (max 100 caractere) dintr-un fisier

Fisier asm: read_short_text_from_file_no_display.asm

```
; Codul de mai jos va deschide un fisier numit "ana.txt" din directorul curent
si va citi un text de maxim 100 de caractere din acel fisier.
; Programul va folosi functia fopen pentru deschiderea fisierului, functia
fread pentru citirea din fisier si functia fclose pentru inchiderea fisierului
deschis.
; Deoarece in apelul functiei fopen programul foloseste modul de acces "r",
daca un fisier cu numele dat nu exista in directorul curent, apelul functiei
fopen nu va reusi (eroare). Detalii despre modurile de acces sunt prezentate
in sectiunea "Suport teoretic".

bits 32

global start

; declare external functions needed by our program
extern exit, fopen, fread, fclose
import exit msvcrt.dll
import fopen msvcrt.dll
import fread msvcrt.dll
import fclose msvcrt.dll

; our data is declared here (the variables needed by our program)
segment data use32 class=data
    nume_fisier db "ana.txt", 0  ; numele fisierului care va fi deschis
    mod_acces db "r", 0          ; modul de deschidere a fisierului -
                                 ; r - pentru scriere. fisierul trebuie sa
existe
    descriptor_fis dd -1         ; variabila in care vom salva descriptorul
fisierului - necesar pentru a putea face referire la fisier
    len equ 100                  ; numarul maxim de elemente citite din
fisier.
    text times len db 0          ; sirul in care se va citi textul din fisier

; our code starts here
segment code use32 class=code
    start:
        ; apelam fopen pentru a deschide fisierul
        ; functia va returna in EAX descriptorul fisierului sau 0 in caz de
eroare
        ; eax = fopen(nume_fisier, mod_acces)
        push dword mod_acces
        push dword nume_fisier
        call [fopen]
        add esp, 4*2                ; eliberam parametrii de pe stiva

        mov [descriptor_fis], eax   ; salvam valoarea returnata de fopen in
variabila descriptor_fis
```

```
        ; verificam daca functia fopen a creat cu succes fisierul (daca EAX !=
0)
        cmp eax, 0
        je final

        ; citim textul in fisierul deschis folosind functia fread
        ; eax = fread(text, 1, len, descriptor_fis)
        push dword [descriptor_fis]
        push dword len
        push dword 1
        push dword text
        call [fread]
        add esp, 4*4                    ; dupa apelul functiei fread EAX contine
numarul de caractere citite din fisier

        ; apelam functia fclose pentru a inchide fisierul
        ; fclose(descriptor_fis)
        push dword [descriptor_fis]
        call [fclose]
        add esp, 4

      final:

        ; exit(0)
        push    dword 0
        call    [exit]
```

Citirea unui text scurt (max 100 caractere) dintr-un fisier – cu afisarea textului

Fisier asm: read_short_text_from_file_and_display.asm

```
; Codul de mai jos va deschide un fisier numit "ana.txt" din directorul
curent, va citi un text scurt din acel fisier, apoi va afisa  in consola
numarul de caractere citite si textul citit din fisier.
; Programul va folosi functia fopen pentru deschiderea fisierului, functia
fread pentru citirea din fisier si functia fclose pentru inchiderea fisierului
creat.
; Deoarece in apelul functiei fopen programul foloseste modul de acces "r",
daca un fisier numele dat nu exista in directorul curent,  apelul functiei
fopen nu va reusi (eroare). Detalii despre modurile de acces sunt prezentate
in sectiunea "Suport teoretic".

; In acest program sirul de caractere in care se va citi textul din fisier
trebuie sa aiba o lungime cu 1 mai mare decat numarul maxim  de elemente care
vor fi citite din fisier deoarece acest sir va fi afisat in consola folosind
functia printf.
; Orice sir de caractere folosit de functia printf trebuie sa fie terminat in
0, altfel afisarea nu va fi corecta.
; Daca fisierul ar contine mai mult de <len> caractere si dimensiunea sirului
destinatie era exact <len>, intregul sir ar fi fost completat cu valori citite
din fisier, astfel sirul nu se mai termina cu valoarea 0.

bits 32
```

```
global start

; declare external functions needed by our program
extern exit, fopen, fread, fclose, printf
import exit msvcrt.dll
import fopen msvcrt.dll
import fread msvcrt.dll
import fclose msvcrt.dll
import printf msvcrt.dll

; our data is declared here (the variables needed by our program)
segment data use32 class=data
    nume_fisier db "ana.txt", 0  ; numele fisierului care va fi creat
    mod_acces db "r", 0          ; modul de deschidere a fisierului -
                                 ; r - pentru scriere. fisierul trebuie sa
existe
    len equ 100                  ; numarul maxim de elemente citite din
fisier.
    text times (len+1) db 0      ; sirul in care se va citi textul din fisier
(dimensiune len+1 explicata mai sus)
    descriptor_fis dd -1         ; variabila in care vom salva descriptorul
fisierului - necesar pentru a putea face referire la fisier
    format db "Am citit %d caractere din fisier. Textul este: %s", 0  ;
formatul - utilizat pentru afisarea textului citit din fisier
                                 ; %s reprezinta un sir de caractere

; our code starts here
segment code use32 class=code
    start:
        ; apelam fopen pentru a deschide fisierul
        ; functia va returna in EAX descriptorul fisierului sau 0 in caz de
eroare
        ; eax = fopen(nume_fisier, mod_acces)
        push dword mod_acces
        push dword nume_fisier
        call [fopen]
        add esp, 4*2                ; eliberam parametrii de pe stiva

        mov [descriptor_fis], eax   ; salvam valoarea returnata de fopen in
variabila descriptor_fis

        ; verificam daca functia fopen a creat cu succes fisierul (daca EAX !=
0)
        cmp eax, 0
        je final

        ; citim textul in fisierul deschis folosind functia fread
        ; eax = fread(text, 1, len, descriptor_fis)
        push dword [descriptor_fis]
        push dword len
        push dword 1
        push dword text
        call [fread]
        add esp, 4*4                ; dupa apelul functiei fread EAX contine
numarul de caractere citite din fisier
```

```
        ; afisam numarul de caractere citite si textul citit
        ; printf(format, eax, text)
        push dword text
        push dword EAX
        push dword format
        call [printf]
        add esp, 4*3

        ; apelam functia fclose pentru a inchide fisierul
        ; fclose(descriptor_fis)
        push dword [descriptor_fis]
        call [fclose]
        add esp, 4

    final:

        ; exit(0)
        push    dword 0
        call    [exit]
```

Citirea intregului text dintr-un fisier (in etape)

Fisier asm: read_full_file.asm

```
; Codul de mai jos va deschide un fisier numit "input.txt" din directorul
curent si va citi intregul text din acel fisier, in etape, cate 100 de
caractere intr-o etapa.
; Deoarece un fisier text poate fi foarte lung, nu este intotdeauna posibil sa
citim fisierul intr-o singura etapa pentru ca nu putem defini un sir de
caractere suficient de lung pentru intregul text din fisier. De aceea,
prelucrarea fisierelor text in etape este necesara.
; Programul va folosi functia fopen pentru deschiderea fisierului, functia
fread pentru citirea din fisier si functia fclose pentru inchiderea fisierului
creat.
; Deoarece in apelul functiei fopen programul foloseste modul de acces "r",
daca un fisier numele dat nu exista in directorul curent,  apelul functiei
fopen nu va reusi (eroare). Detalii despre modurile de acces sunt prezentate
in sectiunea "Suport teoretic".

bits 32

global start

; declare external functions needed by our program
extern exit, fopen, fclose, fread
import exit msvcrt.dll
import fopen msvcrt.dll
import fread msvcrt.dll
import fclose msvcrt.dll

; our data is declared here
segment data use32 class=data
    nume_fisier db "input.txt", 0   ; numele fisierului care va fi deschis
```

```
    mod_acces db "r", 0             ; modul de deschidere a fisierului; r -
pentru scriere. fisierul trebuie sa existe
    descriptor_fis dd -1            ; variabila in care vom salva descriptorul
fisierului - necesar pentru a putea face referire la fisier
    nr_car_citite dd 0              ; variabila in care vom salva numarul de
caractere citit din fisier in etapa curenta
    len equ 100                     ; numarul maxim de elemente citite din
fisier intr-o etapa
    buffer resb len                 ; sirul in care se va citi textul din
fisier

; our code starts here
segment code use32 class=code
    start:
        ; apelam fopen pentru a deschide fisierul
        ; functia va returna in EAX descriptorul fisierului sau 0 in caz de
eroare
        ; eax = fopen(nume_fisier, mod_acces)
        push dword mod_acces
        push dword nume_fisier
        call [fopen]
        add esp, 4*2

        ; verificam daca functia fopen a creat cu succes fisierul (daca EAX !=
0)
        cmp eax, 0
        je final

        mov [descriptor_fis], eax   ; salvam valoarea returnata de fopen in
variabila descriptor_fis

        ; echivalentul in pseudocod al urmatoarei secvente de cod este:
        ;repeta
        ;   nr_car_citite = fread(buffer, 1, len, descriptor_fis)
        ;   daca nr_car_citite > 0
        ;       ; instructiuni pentru procesarea caracterelor citite in
aceasta etapa
        ;pana cand nr_car_citite == 0

        bucla:
            ; citim o parte (100 caractere) din textul in fisierul deschis
folosind functia fread
            ; eax = fread(buffer, 1, len, descriptor_fis)
            push dword [descriptor_fis]
            push dword len
            push dword 1
            push dword buffer
            call [fread]
            add esp, 4*4

            ; eax = numar de caractere / bytes citite
            cmp eax, 0          ; daca numarul de caractere citite este 0, am
terminat de parcurs fisierul
            je cleanup

            mov [nr_car_citite], eax        ; salvam numarul de caractere
citie
```

```
            ; instructiunile pentru procesarea caracterelor citite in aceasta
etapa incep aici
            ; ...

            ; reluam bucla pentru a citi alt bloc de caractere
            jmp bucla

      cleanup:
        ; apelam functia fclose pentru a inchide fisierul
        ; fclose(descriptor_fis)
        push dword [descriptor_fis]
        call [fclose]
        add esp, 4

      final:
        ; exit(0)
        push    dword 0
        call    [exit]
```

Laborator 9 - Probleme propuse

Operatii cu fisiere text

Exerciții

Atentie: fisierele text pot avea dimensiuni foarte mari!

1. Se da un fisier text. Sa se citeasca continutul fisierului, sa se contorizeze numarul de vocale si sa se afiseze aceasta valoare.
   Numele fisierului text este definit in segmentul de date.
2. Se da un fisier text. Sa se citeasca continutul fisierului, sa se contorizeze numarul de consoane si sa se afiseze aceasta valoare.
   Numele fisierului text este definit in segmentul de date.
3. Se da un fisier text. Sa se citeasca continutul fisierului, sa se contorizeze numarul de cifre pare si sa se afiseze aceasta valoare.
   Numele fisierului text este definit in segmentul de date.
4. Se da un fisier text. Sa se citeasca continutul fisierului, sa se contorizeze numarul de cifre impare si sa se afiseze aceasta valoare.
   Numele fisierului text este definit in segmentul de date.
5. Se da un fisier text. Sa se citeasca continutul fisierului, sa se contorizeze numarul de caractere speciale si sa se afiseze aceasta valoare.
   Numele fisierului text este definit in segmentul de date.
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.
7. Se da un fisier text. Sa se citeasca continutul fisierului, sa se determine litera mica (lowercase) cu cea mai mare frecventa si sa se afiseze acea litera, impreuna cu frecventa acesteia.
   Numele fisierului text este definit in segmentul de date.
8. Se da un fisier text. Sa se citeasca continutul fisierului, sa se determine litera mare (uppercase) cu cea mai mare frecventa si sa se afiseze acea litera, impreuna cu frecventa acesteia.
   Numele fisierului text este definit in segmentul de date.
9. Se da un fisier text. Sa se citeasca continutul fisierului, sa se determine caracterul special (diferit de litera) cu cea mai mare frecventa si sa se afiseze acel caracter, impreuna cu frecventa acestuia.
   Numele fisierului text este definit in segmentul de date.
10. Sa se citeasca de la tastatura un nume de fisier si un text. Sa se creeze un fisier cu numele dat in directorul curent si sa se scrie textul in acel fisier.
    Observatii: Numele de fisier este de maxim 30 de caractere. Textul este de maxim 120 de caractere.
11. Se da un nume de fisier (definit in segmentul de date). Sa se creeze un fisier cu numele dat, apoi sa se citeasca de la tastatura cuvinte si sa se scrie in fisier cuvintele citite pana cand se citeste de la tastatura caracterul '$'.
12. Se da un nume de fisier (definit in segmentul de date). Sa se creeze un fisier cu numele dat, apoi sa se citeasca de la tastatura numere si sa se scrie valorile citite in fisier pana cand se citeste de la tastatura valoarea 0.
13. Se dau un nume de fisier si un text (definite in segmentul de date). Textul contine litere mici, litere mari, cifre si caractere speciale. Sa se transforme toate literele mici din textul dat in litere mari. Sa se creeze un fisier cu numele dat si sa se scrie textul obtinut in fisier.

14. Se dau un nume de fisier si un text (definite in segmentul de date). Textul contine litere mici, litere mari, cifre si caractere speciale. Sa se transforme toate literele mari din textul dat in litere mici. Sa se creeze un fisier cu numele dat si sa se scrie textul obtinut in fisier.
15. Se dau un nume de fisier si un text (definite in segmentul de date). Textul contine litere mici, litere mari, cifre si caractere speciale. Sa se inlocuiasca toate caracterele speciale din textul dat cu caracterul 'X'. Sa se creeze un fisier cu numele dat si sa se scrie textul obtinut in fisier.
