UNIVERZITET U BEOGRADU
ELEKTROTEHNIČKI FAKULTET

Katedra za elektroniku

Računarska elektronika

Grupa br. 11

Projekat br. 8

Studenti:

Stefan Vukašinović 466/2013

Jelena Urošević 99/2013

Tekst projekta :

Napisati program koji će analizirati ulazni fajl. Svaki red u ulaznom fajlu je sekvenca karaktera: tekst [pauza], gde je tekst-sekvenca karaktera (ne sme da sadrži blanko znak), pauza-vreme u sekundama nakon kojeg treba obrisati poruku sa ekrana I ispisati novu. Na osnovu sadržaja fajla formira se odgovarajući ispis na ekranu. Na kraju izvršavanja programa potrebno je ispisati njegovo vreme izvršavanja.

Objašnjenje koda:

Formirana je lista korišćenih promenljivih, uključujući i output.txt fajl iz kog se čita željeni tekst, odnosno niz karaktera za ispis. Forma reda u output.txt fajlu je tekst+' '+[x], gde je x broj koji oznacava trajanje pauze, a tekst niz karaktera bez blanko znaka. Svaki sledeci red u output.txt fajlu je iste forme.

Program je realizovan tako da iščitava karakter po karakter, dok ne dodje do blanko znaka. To se postiže manipulacijom registara, nakon sto se adresa od koje kreće output.txt fajl smesti u registar edx. Iščitavanjem karaktera se postize formiranje sekvence za ispis koja će nakon x vremena biti izbrisana da bi se ispisao sledeci red itd.

U petlji loop1 se iščitava niz karaktera sve dok se ne dođe do blanko znaka, dok se u loop2 iščitava vremenski period, odnosno trajanje pauze tokom koje taj niz karaktera ostaje ispisan na ekranu.

Takođe, na kraju obrade fajla, biće ispisano ukupno trajanje izvršavanja programa uz pomoć GetTickCount funkcije izraženo u milisekundama.

Kod:

```
; Reading a File                (ReadFile.asm)

; Opens, reads, and displays a text file using
; procedures from Irvine32.lib.

INCLUDE Irvine32.inc
INCLUDE macros.inc

BUFFER_SIZE = 50
BUFFER_SIZE1 = 5

.data
buffer BYTE BUFFER_SIZE DUP(?)
buffer1 BYTE BUFFER_SIZE1 DUP(?)
filename    BYTE "output.txt",0
fileHandle  HANDLE ?
eax_cnt WORD 0
edx_cnt WORD 0
startTime DWORD ?

.code
main PROC
```

```
INVOKE GetTickCount
mov     startTime,eax


; Open the file for input.
        mov     edx,OFFSET filename
        call    OpenInputFile
        mov     fileHandle,eax

; Check for errors.
        cmp     eax,INVALID_HANDLE_VALUE        ; error opening file?
        jne     file_ok                         ; no: skip
        mWrite <"Cannot open file",0dh,0ah>
        jmp     quit                            ; and quit
file_ok:


loop1:

; Read the file into a buffer.
   mov ecx,1
        mov     edx,OFFSET buffer
        add dx,[edx_cnt]
        mov     eax,fileHandle
        call    ReadFromFile   ;Return arg: If CF = 0, EAX contains the number of bytes read.
        cmp eax,0              ;
        je close_file          ;kraj fajla

        ;mov    edx,OFFSET buffer       ;provera za kraj fajla
        ;add dx,[edx_cnt]
        ;movzx ecx,BYTE PTR[edx]
        ;cmp ecx,'EOF'
        ;je close_file


        mov ecx,1

        mov ax,[edx_cnt]
        inc ax
        mov [edx_cnt],ax

        mov ax,[eax_cnt]    ;eax_cnt sluzi da bi eax mogao da uzima pravilne vrednosti
        add ax,1
        mov [eax_cnt],ax

        cmp     buffer[eax-1],' '

        jne loop1
        mov     buffer[eax],0           ; insert null terminator
```

```
; Display the buffer.
        ;mWrite <"Buffer:",0dh,0ah,0dh,0ah>
        mov     edx,OFFSET buffer    ; display the buffer
        call    WriteString
        call    Crlf


        mov ecx,1
        mov     edx,OFFSET buffer1
        mov     eax,fileHandle
        call    ReadFromFile

        mov ax,[edx_cnt]       ;edx_cnt=0
        xor ax,ax
        mov [edx_cnt],ax

        mov ax,[eax_cnt]       ;eax_cnt=0
        xor ax,ax
        mov [eax_cnt],ax

loop2:

        mov ecx,1
        mov     edx,OFFSET buffer1
        add     dx,[edx_cnt]
        mov     eax,fileHandle
        call    ReadFromFile

        mov ax,[edx_cnt]
        inc ax
        mov [edx_cnt],ax

        mov ax,[eax_cnt]    ;eax_cnt sluzi da bi eax mogao da uzima pravilne vrednosti
        add ax,1
        mov [eax_cnt],ax

        cmp     buffer1[eax-1],']'

        jne loop2

        mov     buffer1[eax],0          ; insert null terminator posle znaka ']'

        ; Display the buffer.
        ;mWrite <"Buffer:",0dh,0ah,0dh,0ah>
        ;mov    edx,OFFSET buffer1  ; display the buffer1 koji sadrzi vreme pauze
        ;call   WriteString
        ;call   Crlf
```

```
        xor eax, eax
        ;xor ecx,ecx
        mov edx,OFFSET buffer1
top:
        movzx ecx,BYTE PTR[edx]
        inc edx
        ;cmp ecx, '0' ; valid?
   ;jb done
   cmp ecx, '9'     ;prekida kad naidje na ']'
   ja done
        sub ecx,'0'
        imul eax,10
        add eax,ecx
        jmp top
done:
   imul eax,1000

   INVOKE SLEEP,eax

        INVOKE Clrscr


        mov ax,[edx_cnt]
        xor ax,ax
        mov [edx_cnt],ax

        mov ax,[eax_cnt]    ;eax_cnt sluzi da bi eax mogao da uzima pravilne vrednosti
        xor ax,ax
        mov [eax_cnt],ax

        jmp loop1


close_file:
   INVOKE GetTickCount
        sub     eax,startTime
        call    WriteDec
        mWrite <" milliseconds have elapsed",0dh,0ah>

        mov     eax,fileHandle
        call    CloseFile

quit:
        exit
main ENDP

END main
```