Elektrotehnički fakultet Univerziteta u Beogradu

Projekat iz predmeta Računarska elektronika

– Projekat broj 3 –

Stefan Stanković 181/2013
Stefan Vukčević 509/2013

Sadržaj

1. Tekst projekta

Napisati program koji pravi sliku koja nastaje poređenjem ulazne slike sa pragom th = 128, tako što sve vrednosti koje su veće od th postavi na 255 (belu boju), a manje na 0 (crnu boju).

2. Kratak opis projektnog koda

Ceo kod je podeljen na tri glavne celine:

1. Učitavanje ulaznog fajla
2. Obrada ulazne datoteke
3. Ispis rezultata obrade u izlaznu datoteku

2.1 Učitavanje ulaznog fajla

- Ovaj deo koda smešten je u ***main*** proceduru

- U početku se od korisnika zahteva da preko standardnog ulaza unese ime ulazne datoteke koja će biti obrađivana

- Nakon što ime ulazne datoteke bude sačuvano u promenljivoj ***infilename*** poziva se procedura za otvaranje ulazne datoteke, proverava se da li je došlo do greške prilikom otvaranja (fajl ne postoji, ili bilo koja druga greška koja može da se javi prilikom otvaranja ulaznog fajla…), i u skladu sa rezultatom provere se nastavlja dalje

- U slucaju da je došlo do greške, ispisuje se na standardnom ulazu podatak o tome i skače se na prekid programa

- Ako greška ne postoji skače se na deo koda za učitavanje sadržaja datoteke u ulazni ***buffer***

- Pozivom procedure ***ReadFromFile*** se sadržaj datoteke učitava u ***buffer***, i zatim se proverava da li je došlo do greške prilikom čitanja

- U slučaju da je došlo do greške, ispisuje se na standardnom ulazu podatak o tome i skače se na prekid programa

- Proverava se da li je veličina buffer-a dovoljno velika, i u slučaju da nije ispisuje se podatak o tome na standardnom izlazu i skače se na prekid programa

- Ako je buffer zadovoljavajuce veličine postavlja se terminator 0 na kraju učitanog stringa, i zatvara se ulazni fajl, nakon čega se prelazi na obradu

2.2 Obrada ulazne datoteke

- Obrada ulazne datoteke smeštena je u proceduru ***proccess***

- Ova procedura u početku ima petlju, čiji je cilj prepisivanje zaglavlja ulazne datoteke iz ulaznog bafera u izlazni bafer, u kojoj poziva drugu, pomoćnu proceduru ***row_copy_paste*** koja služi za prosto prepisivanje redova iz ulaznog u izlazni bafer. Pored toga što prepisuje redove, ova procedura uvećava promenljivu ***counter*** pomoću koje se u petlji registruje da je prepisano zaglavlje ulazne datoteke sa sve komentarima koji se mogu javiti u zaglavlju (Kada ***counter*** dobije vrednost 3 to znači da su pored svih komentara prepisana i tri osnovna reda zaglavlja pa se može nastaviti dalje sa obradom)

- U proceduri ***row_copy_paste*** se jednostavno prepisuje red po red zaglavlja, pri čemu se ukoliko je prvi pročitani karakter u redu # ne uvećava ***counter*** jer je onda taj red komentar, dok se u suprotnom uvećava jer je taj red onda standardni deo zaglavlja

- Nakon što se završi prepisivanje zaglavlja prelazi se na glavni deo obrade, a to je poređenje svakog piksela sa pragom i upisivanje nove vrednosti piksela u izlazni bafer. Detalji ove obrade su opisani u komentarima u kodu

- Algoritam u kratkim crtama:

  - U registru edx se pamti položaj prve cifre vrednosti piksela (npr u jednom redu datoteke 98 253 45… pamti se polozaj cifre 9 ili 2 ili 4 zavisno dokle se stiglo sa čitanjem)
  - Čita se karakter po karakter do momenta kada pročitani karakter nije cifra (u ovom slučaju pročitani karakter je razmak)
  - Proverava se da li je razmak između početne pozicije stavljene na stek, i pozicije poslednje pročitanog karaktera **1** (to bi značilo da su dva uzastopno pročitana karaktera razmak i znak za novi red, pa se u tom slučaju samo poslednji pročitani karakter prepiše u izlazni bafer)
  - U slučaju da je razmak veći od **1** to znači da je je pročitan neki broj, i pozivom procedure ***ParseDecimal32*** se broj iz zapisa string-a konvertuje u decimalni broj

- Od tako dobijenog broja se oduzima vrednost praga. U slučaju da je rezultat negativan, znači da je vrednost piksela bila manja od vrednosti praga pa se u izlazni bafer upisuje 0, dok se u suprotnom u izlazni bafer upisuje 255
- Algoritam se ponavlja sve dok se ne dodje do kraja ulaznog bafera, a zatim se iz procedure ***proccess*** vraća u glavni program kako bi rezultati obrade bili ispisani u izlaznu datoteku

2.3 Ispis rezultata obrade u izlaznu datoteku

- Ovaj deo koda je kao i čitanje smešten u ***main*** proceduru.

- U početku se od korisnika zahteva da preko standardnog ulaza unese ime izlazne datoteke u koju će biti upisani rezultati obrade

- Nakon što ime izlazne datoteke bude sačuvano u promenljivoj ***outfilename*** poziva se procedura za pravljenje izlazne datoteke, proverava se da li je došlo do greške prilikom pravljenja, i u skladu sa rezultatom provere se nastavlja dalje

- U slučaju da je došlo do greške, ispisuje se na standardnom ulazu podatak o tome i skače se na prekid programa

- U suprotnom se rezultati obrade ispisuju u izlaznu datoteku, zatvara se izlazni fajl i uspešno se završava se program

3. Projektni kod

```
INCLUDE Irvine32.inc
INCLUDE macros.inc

BUFFER_SIZE = 256 * 256 * 10

.data
buffer BYTE BUFFER_SIZE DUP(? )
infilename    BYTE 80 DUP(0)
outfilename    BYTE 80 DUP(0)
fileHandle  HANDLE ?
stringLength DWORD ?
outBuffer BYTE BUFFER_SIZE DUP(? )
counter BYTE 0


.code

;Procedura za prepisivanje reda u izlazni bafer
row_copy_paste PROC
   lodsb                ;u slucaju da je prvi karakter u redu # to znaci da
   stosb                ;je taj red komentar (neobavezni deo zaglavlja) i samo
   dec ecx              ;se prepisuje karakter po karakter u izlazni bafer
   cmp eax, '#'         ;
   je paste             ;u clucaju da nije # to je povecava se counter kojim
   inc counter          ;se  u proccess registruje da li je prepisano zaglavlje
paste:                  ;(ako je counter=3, zato sto postoje tri reda zaglavlja
   lodsb                ;u sadrzaju slike koja je u formatu pgm:
   stosb                ;P2
   cmp eax, 0ah         ;broj redova broj kolona
   je endProc           ;maksimalna vrednost pixela)
   loop paste           ;pri cemu se komentar moze naci izmedju svakog
endProc:                ;od ova tri reda
   ret
row_copy_paste ENDP
```

```
;Procedura za obradu ulazne datoteke
proccess PROC
    cld
    mov esi, OFFSET buffer ;izvorisni string
    mov edi, OFFSET outBuffer ;odredisni string
    mov ecx, LENGTHOF buffer ;brojac za petlju

;petlja kojom se prepisuje zaglavlje ulazne datoteke u izlaznu
copy:
    cmp counter, 3
    je move_on
    call row_copy_paste
    loop copy

;glavna obrada pixela
;poredjenje svakog pixela sa th = 128,
;i postavljanje nove vrednosti pixela u outBuffer
move_on:
    mov edx, esi            ;pamti se pocetni polozaj broja unutar stringa
loop1:
    lodsb                   ;petlja se vrti dokle god
    call IsDigit            ;je procitani karakter cifra
    jnz notDigit            ;u suprotnom skok na notDigit
    loop loop1              ;
    jmp finish              ;ako je kraj bafera zavrsi obradu
notDigit:
    push esi                ;u slucaju dva uzastopna karaktera
    sub esi, edx            ;koji nisu cifre (razmak + novi red)
    cmp esi, 1              ;prepisuje se procitani karakter i
    jne compare             ;vraca se na move_on
    stosb                   ;u suprotnom znaci da se moze procitati broj
    pop esi                 ;tj. skociti na poredjenje sa th (compare)
    loop move_on            ;
    jmp finish              ;ako je kraj bafera zavrsi obradu
compare:
    push ecx                ;stavljaju se na stek vrednosti ecx i eax
    push eax                ;jer su ti registri potrebni za dalji rad
    mov ecx, esi            ;u ecx se prebacuje broj cifara vrednosti pixela
    call ParseDecimal32     ;konvertuje se string u decimalni broj
    sub eax, 128            ;oduzima se th=128 od dobijenog broja
```

```
    jc zero                 ;ako je rezultat negativan broj je manji od th
                            ;i skace se na zero

    mov eax, '2'            ;ako je broj bio veci od th u izlazni bafer
    stosb                   ;se upisuje vrednost 255
    mov eax, '5'
    stosb
    stosb
    pop eax                 ;skida se sa steka prethodno stavljena vrednost eax
    stosb                   ;i upisuje u izlazni bafer (to je char koji nije bio cifra)
    jmp stek                ;skace se na labelu stek
zero:
    mov eax, '0'            ;ako je broj bio manji od th u izlazni bafer
    stosb                   ;se upisuje vrednost 0
    pop eax                 ;skida se sa steka prethodno stavljena vrednost eax
    stosb                   ;i upisuje u izlazni bafer (to je char koji nije bio cifra)

stek:
    pop ecx                 ;skidaju se vrednosti registara koje su prethodno stavljene
    pop esi                 ;na stek kako bi se nastavilo sa normalnim radom
    loop move_on            ;ceo proces se ponavlja dok se ne dodje do kraja bafera

finish:
    ret                     ;povratak iz obrade
proccess ENDP


;Glavni program
main PROC

;cekaj ime infajla
    mWrite "Ime ulazne datoteke?: "
    mov edx, OFFSET infilename
    mov ecx, SIZEOF infilename
    call ReadString

;Otvori fajl
    mov edx, OFFSET infilename
    call OpenInputFile
    mov fileHandle, eax
```

```
;Proveri greske
    cmp eax, INVALID_HANDLE_VALUE ;nesto ne radi ?
    jne file_ok_in ;ako je ok skoci
    mWrite <"Greska prilikom otvaranja ulazne datoteke.", 0dh, 0ah>
    jmpquit ;zavrsi program u slucaju greske

;Speak friend, and enter
file_ok_in :
    mov edx, OFFSET buffer
    mov ecx, BUFFER_SIZE
    call ReadFromFile
    jnc check_buffer_size ;greska citanja
    mWrite "Greska u citanju." ;ako jeste, kazi da je tako
    call WriteWindowsMsg
    jmp close_file

;Provera da li je dovoljno veliki bafer
check_buffer_size :
    cmp eax, BUFFER_SIZE ;da li je dovoljno veliki ?
    jb   buf_size_ok ;ako jeste skoci
    mWrite <"Greska: mali je bafer", 0dh, 0ah>
    jmp quit

;ovde predje ako je dovoljno veliki
buf_size_ok :
    mov buffer[eax], 0 ;terminator na kraju
    mWrite "Koliko je veliko: "
    mov stringLength, eax
    call WriteDec
    call Crlf

;zatvori ulazni fajl
close_file :
    mov eax, fileHandle
    call CloseFile

;obrada bafera u proceduri proccess
    call proccess
```

```
;cekaj ime outfajla
    mWrite "Ime izlazne datoteke?: "
    mov edx, OFFSET outfilename
    mov ecx, SIZEOF outfilename
    call ReadString

; Napravi novi fajl
    mov edx, OFFSET outfilename
    call CreateOutputFile
    mov fileHandle, eax

; Greske?
    cmp eax, INVALID_HANDLE_VALUE; error found ?
    jne file_ok_out ;ako je sve u redu ispisi bafer u izlaznu datoteku
    mWrite <"Greska prilikom pravljenja izlazne datoteke.", 0dh, 0ah>
    jmp quit ;Kill it before it lays eggs!

;bafer u output!
file_ok_out :
    mov eax, fileHandle
    mov edx, OFFSET outBuffer
    mov ecx, LENGTHOF outBuffer
    call WriteToFile
    mov eax, fileHandle
    call CloseFile

;kraj programa
quit :
    exit
    main ENDP

END main
```