IZVEŠTAJ ZA
PROJEKAT IZ RAČUNARSKE ELEKTRONIKE

Studenti:

Katarina Stiković 2012/215

Milutin Marenović 2015/319

Predmetni professor:

prof Milan Prokin

Predmetni asistent:

prof Aleksandra Lekić

TEKST PROJEKTA

Projekat 25∗ - Minesweeper

Napraviti igricu Minesweeper:
Prozor igrice (tabla) je podeljen na 9 × 9 polja koja su na početku igrice sva iste boje. Na 10 lokacija na tabli, odnosno iza 10 polja se nalaze mine koje su prikrivene. Kroz polja se kreće kursorima na tastaturi (na neki način označiti polje na kome se trenutno nalazi kursor). Polja napraviti veličine 2 × 2.
Pritiskom tastera SPACE se otvara polje. Ukoliko se na polju ne nalazi mina, rekurzivno se otvaraju i sva susedna polja (promeniti boju ovih polja) na kojima nema mine. Na ivicama otvorene oblasti bez mina se ispisuje broj koliko se mina nalazi pored datog polja u redu, koloni i djagonalno. Pritiskom na minu se završava igrica. Ako se pretpostavlja da se iza datog polja nalazi mina, onda se to polje označava pritiskom tastera Left Shift i tada dato polje postaje druge boje.
Igrica se završava kada se otvore sva polja na kojima nema mine i označe sva polja sa minama. Ukoliko se otvori polje na kome se nalazi mina, takođe se završava igrica. Pritiskom tastera ESC omogućiti prekid programa.

KRATAK OPIS REŠENJA ZADATKA

U .data sekciji se nalaze podaci koji su upisani u memoriju, a to su sledeći:

- Podaci za postavljanje ekrana
- Podaci za različite potrebne karaktere i boje
- Matrica veličine 9x9 u kojoj se nalaze binarni podaci: 0 – ako se na tom polju ne nalazi mina, i 1 – ako se na tom polju nalazi mina
- Matrica veličine 9x9 u kojoj se, na svakom polju, nalazi broj od 0 do 9, koji nam pokazuje koliko mina se nalazi oko tog polja. Polja ove matrice popunjava sam program
- Matrica veličine 9x9 u kojoj se nalazi podatak o stanju trenutnog polja, i to: 0 – ako je polje i dalje zatvoreno, 1 – ako je polje otvoreno i 2 – ako se na polju nalazi zastavica (odnosno ako igrač smatra da se na tom polju nalazi mina). Menja se sve vreme dok traje program.

Pomoćne procedure koje su iskorišćene su sledeće:

- PROC **popunjavanjeEkrana**: Ovaj podprogram postavlja karaktere koji predstavljaju zatvoreno polje na svih 9x9 polja ekrana. Izvršena je kao dupla FOR petlja
- PROC **pocetnaPozicijaKursora**: U ovom podprogramu se postavlja početna pozicija kursora koja je fiksno podešena na (4, 4). Na tom polju se menja boja pozadine kako bi igrač znao u svakom trenutku gde mu se nalazi kursor.
- PROC **popuniMatricu**: Napravljena je kao dupla FOR petlja, obilazi sva polja i poziva proceduru „izbrojiMine“ da nastavi izvršavanje
- PROC **izbrojiMine**: Ovaj podprogram za prosledjeno polje iz procedure „popuniMatricu“, izračunava koliko se mina nalazi u poljima oko tog polja. Ako su kordinate tog polja (x,y), onda se proveravaju sledeća polja: (x-1, y-1), (x, y-1), (x+1, y-1), (x-1, y), (x+1, y), (x-1, y+1), (x, y+1) i (x+1, y+1). Naravno, program preskače sve specijalne slučajeve, kao što su na primer polja koja se nalaze na ivici, odnosno sa koordinatama 0 ili 8.
- PROC **izracunavanje**: Pomoćna procedura koja na osnovu prosleđenih parametara (x i y koordinate) određuje koji se podatak čita iz određene matrice.
- PROC **promenaNijeOtvoreno**, PROC **promenaOtvoreno** i PROC **promenaZastava**: U zavisnosti od podataka iz matrice o trenutnom stanju polja, zavisi koja će se procedura pozvati od ove tri. Sve tri procedure rade sličnu stvar: na željenje koordinate postavljaju određeni znak (znak za prazno polje, znak za otvoreno polje ili znak za zastavu). U slučaju da je znak za otvoreno polje u pitanju, onda se preko matrice o broju mina, postavlja broj koliko mina se nalazi oko tog polja.
- PROC **izracunajTrenutnoPolje**: Ovaj podprogram u svakom potrebnom trenutku može da pronađe na kakvom se polju nalazi kursor (otvoreno polje, neotvoreno polje ili polje sa zastavom)

Ovi svi podprogrami se pozivaju iz glavnog programa. Tog glavnog programa je sledeći:

Određuju se podaci o ekranu: veličina, kursor, naziv itd. Ekran se popunjava sa karakterima koji su određeni za neotvoreno polje. Kursor se postavlja na lokaciju (4, 4). Određuju se podaci o minama (popunjava se matrica o broju mina). Ovo je deo koji se samo jednom izvršava kada se pokrene. Sledeći deo je deo koji se ponavlja u zavisnosti od dužine igranja igrice.

Čeka se na unos karaktera od strane igrača. Karakteri koji se unose mogu biti: LEFT, RIGHT, UP ili DOWN za kretanje kursora po ekranu, SPACEBAR za otvaranje polja, SHIFT LEFT za postavljanje zastave ili ESC za izlazak iz igrice.

Kada se unese polje za kretanje, tada se prvo proverava da li je moguće kretanje kursora u željenom smeru, a zatim, kada se utvrdi da može, boja koja određuje gde se kursor nalazi, se premešta na sledeću lokaciju.

Kada se pritisne SPACEBAR, karakter za otvaranje polja, tada se utvrđuje da li se na tom polju nalazi mina, i ako da, onda se završava igrica. U suprotnom se otvara polje, i na njemu se nalazi podatak o broju mina oko tog polja.

Kada se pritisne SHIFT LEFT tada se ili postavlja zastava na to polje, ili uklanja zastava ako je prethodno bila postavljena na tom polju. Čuva se podatak o broju pogođenih mina i ako dostigne 10 (odnosno ukupan broj mina) znači da je protivnik pogodio sva polja na kojima se nalazi mina, i tako se završava igrica uspešno.

Kada se pritisne ESC izlazi se iz igrice.

Jedina korišćena biblioteka je **Irvine32.inc**

KOD CELOG PROGRAMA:

```
INCLUDE Irvine32.inc

.data

; za postavljanje ekrana
outHandle    DWORD ?
cellsWritten DWORD ?
scrSize COORD <80,30>
xyPos COORD <0,0>
consoleInfo CONSOLE_SCREEN_BUFFER_INFO <>
cursorInfo CONSOLE_CURSOR_INFO <>
titleStr BYTE "Minesweeper",0
windowRect   SMALL_RECT <0,0,18,18>

; za karaktere i boje
karakterPrazan = 0B0h
kursorX byte 4
kursorY byte 4
kursorBoja = yellow
pozadinaBoja = white
trenutnoPolje byte 0
counter byte 0
dataX byte 0
dataY byte 0

matricaMina byte 1, 1, 0, 1, 0, 1, 0, 0, 1
                byte 0, 0, 1, 0, 0, 0, 0, 0, 0
                byte 0, 0, 0, 0, 0, 0, 0, 0, 0
                byte 0, 0, 1, 0, 1, 0, 0, 1, 0
                byte 0, 0, 0, 0, 0, 0, 0, 0, 0
                byte 0, 1, 1, 0, 0, 0, 0, 0, 0
                byte 0, 0, 0, 0, 0, 0, 0, 0, 0
                byte 0, 1, 0, 0, 0, 0, 1, 0, 0
                byte 0, 0, 0, 0, 0, 1, 0, 0, 0

matricaMinaBroj byte 0, 0, 0, 0, 0, 0, 0, 0, 0
                byte 0, 0, 0, 0, 0, 0, 0, 0, 0
```

```
                byte 0, 0, 0, 0, 0, 0, 0, 0, 0
                byte 0, 0, 0, 0, 0, 0, 0, 0, 0
                byte 0, 0, 0, 0, 0, 0, 0, 0, 0
                byte 0, 0, 0, 0, 0, 0, 0, 0, 0
                byte 0, 0, 0, 0, 0, 0, 0, 0, 0
                byte 0, 0, 0, 0, 0, 0, 0, 0, 0
                byte 0, 0, 0, 0, 0, 0, 0, 0, 0

; 0 - zatvoreno polje, 1 - otvoreno polje, 2 - zastava
matricaMinaPodatak byte 0, 0, 0, 0, 0, 0, 0, 0, 0
                byte 0, 0, 0, 0, 0, 0, 0, 0, 0
                byte 0, 0, 0, 0, 0, 0, 0, 0, 0
                byte 0, 0, 0, 0, 0, 0, 0, 0, 0
                byte 0, 0, 0, 0, 0, 0, 0, 0, 0
                byte 0, 0, 0, 0, 0, 0, 0, 0, 0
                byte 0, 0, 0, 0, 0, 0, 0, 0, 0
                byte 0, 0, 0, 0, 0, 0, 0, 0, 0
                byte 0, 0, 0, 0, 0, 0, 0, 0, 0

.code

        ; ******************************
        ; for loop za popunjavanje ekrana
        ; ******************************

popunjavanjeEkrana PROC
                mov eax, pozadinaBoja
                call SetTextColor
                mov ch, 9
                mov cl, 9
                mov dh, 0
                mov dl, 0
                mov al, karakterPrazan
        loop2:
        loop1:
                call Gotoxy
                call WriteChar
                inc dh
                cmp ch, dh
                jnz loop1
                mov dh, 0
```

```
                inc dl
                cmp cl, dl
                jnz loop2
        ret
popunjavanjeEkrana endp

        ; ****************************
        ; pocetna pozicija kursora
        ; ****************************

pocetnaPozicijaKursora PROC
                mov eax, kursorBoja
                call SetTextColor
                mov dh, kursorY
                mov dl, kursorX
                call Gotoxy
                mov al, karakterPrazan
                call WriteChar
                call Crlf ;***
                mov eax, pozadinaBoja
                call SetTextColor
        ret
pocetnaPozicijaKursora endp

        ; ****************************
        ; broji mine oko jednog polja
        ; ****************************

izracunavanje PROC
        mov cl, 9
        mov ch, 9
        xor ebx, ebx
        mov bl, dataY
        xor edx, edx
petlja:
        add edx, ebx
        dec cl
        cmp cl, 0
        jne petlja
        mov bl, dataX
        add edx, ebx
```

```
        mov esi, edx

        ret
izracunavanje endp

        ; ******************************
        ; broji mine oko jednog polja
        ; ******************************
izbrojiMine PROC
                mov dataX, ch
                mov dataY, cl
                mov counter, 0

        LOOPxy0:
                ; x-1, y-1
                dec dataX
                dec dataY
                cmp dataX, 255
                je LOOPxy1
                cmp dataY, 255
                je LOOPxy1
                cmp dataX, 9
                je LOOPxy1
                cmp dataY, 9
                je LOOPxy1
                call izracunavanje
                xor eax, eax
                cmp [matricaMina + esi], 1
                jne LOOPxy1
                inc counter
        LOOPxy1:
                ; x, y-1
                inc dataX
                cmp dataX, 255
                je LOOPxy2
                cmp dataY, 255
                je LOOPxy2
                cmp dataX, 9
                je LOOPxy2
                cmp dataY, 9
                je LOOPxy2
```

```
        call izracunavanje
        cmp [matricaMina + esi], 1
        jne LOOPxy2
        inc counter
LOOPxy2:
        ; x+1, y-1
        inc dataX
        cmp dataX, 255
        je LOOPxy3
        cmp dataY, 255
        je LOOPxy3
        cmp dataX, 9
        je LOOPxy3
        cmp dataY, 9
        je LOOPxy3
        call izracunavanje
        cmp [matricaMina + esi], 1
        jne LOOPxy3
        inc counter
LOOPxy3:
        ; x-1, y
        inc dataY
        dec dataX
        dec dataX
        cmp dataX, 255
        je LOOPxy4
        cmp dataY, 255
        je LOOPxy4
        cmp dataX, 9
        je LOOPxy4
        cmp dataY, 9
        je LOOPxy4
        call izracunavanje
        cmp [matricaMina + esi], 1
        jne LOOPxy4
        inc counter
LOOPxy4:
        ; x+1, y
        inc dataX
        inc dataX
        cmp dataX, 255
```

```
        je LOOPxy5
        cmp dataY, 255
        je LOOPxy5
        cmp dataX, 9
        je LOOPxy5
        cmp dataY, 9
        je LOOPxy5
        call izracunavanje
        cmp [matricaMina + esi], 1
        jne LOOPxy5
        inc counter
LOOPxy5:
        ; x-1, y+1
        dec dataX
        dec dataX
        inc dataY
        cmp dataX, 255
        je LOOPxy6
        cmp dataY, 255
        je LOOPxy6
        cmp dataX, 9
        je LOOPxy6
        cmp dataY, 9
        je LOOPxy6
        call izracunavanje
        cmp [matricaMina + esi], 1
        jne LOOPxy6
        inc counter
LOOPxy6:
        ; x, y+1
        inc dataX
        cmp dataX, 255
        je LOOPxy7
        cmp dataY, 255
        je LOOPxy7
        cmp dataX, 9
        je LOOPxy7
        cmp dataY, 9
        je LOOPxy7
        call izracunavanje
        cmp [matricaMina + esi], 1
```

```
            jne LOOPxy7
            inc counter
        LOOPxy7:
            ; x+1, y+1
            inc dataX
            cmp dataX, 255
            je LOOPxy8
            cmp dataY, 255
            je LOOPxy8
            cmp dataX, 9
            je LOOPxy8
            cmp dataY, 9
            je LOOPxy8
            call izracunavanje
            cmp [matricaMina + esi], 1
            jne LOOPxy8
            inc counter
        LOOPxy8:
            ; upis u matricu sa brojem mina
            dec dataX
            dec dataY
            call izracunavanje
            mov bl, counter
            add bl, 30h
            mov matricaMinaBroj[esi], bl
            mov cl, dataY
            mov ch, dataX
            mov eax, 0
            mov counter, al
        ret
izbrojiMine endp


        ; ******************************
        ; izracunavanje matrice mina
        ; ******************************

popuniMatricu PROC
        xor cl, cl ; brojac za y
        xor ch, ch ; brojac za x
        FORX:
```

```
	FORY:
		call izbrojiMine
		inc ch
		cmp ch, 9
		jnz FORX
		mov ch, 0
		inc cl
		cmp cl, 9
		jnz FORY
	ret
popuniMatricu endp

	; ******************************
	; promena - nije otvoreno polje
	; ******************************
promenaNijeOtvoreno PROC
		mov dl, kursorX
		mov dh, kursorY
		call Gotoxy
		mov al, karakterPrazan
		call WriteChar
	ret
promenaNijeOtvoreno endp

	; ******************************
	; promena - otvoreno polje
	; ******************************
promenaOtvoreno PROC
		xor eax, eax
		mov al, kursorX
		mov dataX, al
		mov al, kursorY
		mov dataY, al
		call izracunavanje
		mov al, matricaMinaBroj[esi]
		mov dl, kursorX
		mov dh, kursorY
		call Gotoxy
		call WriteChar
	ret
promenaOtvoreno endp
```

```
	; ******************************
	; promena - zastava
	; ******************************
promenaZastava PROC
		;*****
	ret
promenaZastava endp

	; ******************************
	; u 'trenutnoPolje' upisuje se podatak o trenutnom polju iz 'matricaMinaPodatak'
	; ******************************
izracunajTrenutnoPolje PROC
		xor eax, eax
		mov al, kursorX
		mov dataX, al
		mov al, kursorY
		mov dataY, al
		call izracunavanje
		mov al, matricaMinaPodatak[esi]
		mov trenutnoPolje, al
	ret
izracunajTrenutnoPolje endp

	; ******************************
	; main program
	; ******************************

main PROC

	INVOKE GetStdHandle,STD_OUTPUT_HANDLE
	mov outHandle,eax

	INVOKE GetConsoleCursorInfo, outHandle,
	  ADDR cursorInfo

	INVOKE SetConsoleScreenBufferSize,
	  outHandle,scrSize

	INVOKE SetConsoleCursorPosition, outHandle, xyPos
```

```
INVOKE SetConsoleTitle, ADDR titleStr

INVOKE GetConsoleScreenBufferInfo, outHandle,
  ADDR consoleInfo

INVOKE SetConsoleWindowInfo,
        outHandle,
        TRUE,
        ADDR windowRect

call popuniMatricu

call popunjavanjeEkrana

call pocetnaPozicijaKursora

; ****************************
; unos karaktera
; ****************************
        xor al, al
loopKey:
        mov eax, 50
        call Delay
        call ReadKey
        jz loopKey

        ; ****************************
        ; levo
        ; ****************************

_if1:
        cmp al, 'a'
        je THEN1
        jmp ENDIF1
THEN1:
        call izracunajTrenutnoPolje

        ; provera da nije kraj ekrana
        cmp kursorX, 0
        je ENDIF1
```

```
        ; vraca pozadinu
        cmp trenutnoPolje, 0
        je A0
        jmp AX0
A0:
        call promenaNijeOtvoreno
AX0:
        cmp trenutnoPolje, 1
        je A1
        jmp AX1
A1:
        call promenaOtvoreno
AX1:
        cmp trenutnoPolje, 2
        je A2
        jmp AX2
A2:
        call promenaZastava
AX2:

        ; pomera kursor
        mov eax, kursorBoja
        call SetTextColor
        dec kursorX
        call izracunajTrenutnoPolje
        cmp trenutnoPolje, 0
        je B0
        jmp BX0
B0:
        call promenaNijeOtvoreno
BX0:
        cmp trenutnoPolje, 1
        je B1
        jmp BX1
B1:
        call promenaOtvoreno
BX1:
        cmp trenutnoPolje, 2
        je B2
        jmp BX2
B2:
```

```
        call promenaZastava
BX2:
        mov eax, pozadinaBoja
        call SetTextColor
ENDIF1:

        ; ******************************
        ; desno
        ; ******************************

_if2:
        cmp al, 'd'
        je THEN2
        jmp ENDIF2
THEN2:
        call izracunajTrenutnoPolje

        ; provera da nije kraj ekrana
        cmp kursorX, 8
        je ENDIF2

        ; vraca pozadinu
        cmp trenutnoPolje, 0
        je C0
        jmp CX0
C0:
        call promenaNijeOtvoreno
CX0:
        cmp trenutnoPolje, 1
        je C1
        jmp CX1
C1:
        call promenaOtvoreno
CX1:
        cmp trenutnoPolje, 2
        je C2
        jmp CX2
C2:
        call promenaZastava
CX2:
```

```
        ; pomera kursor
        mov eax, kursorBoja
        call SetTextColor
        inc kursorX
        call izracunajTrenutnoPolje
        cmp trenutnoPolje, 0
        je D0
        jmp DX0
D0:
        call promenaNijeOtvoreno
DX0:
        cmp trenutnoPolje, 1
        je D1
        jmp DX1
D1:
        call promenaOtvoreno
DX1:
        cmp trenutnoPolje, 2
        je D2
        jmp DX2
D2:
        call promenaZastava
DX2:
        mov eax, pozadinaBoja
        call SetTextColor
ENDIF2:

        ; ******************************
        ; gore
        ; ******************************

_if3:
        cmp al, 'w'
        je THEN3
        jmp ENDIF3
THEN3:
        call izracunajTrenutnoPolje

        ; provera da nije kraj ekrana
        cmp kursorY, 0
        je ENDIF3
```

```
        ; vraca pozadinu
        cmp trenutnoPolje, 0
        je E0
        jmp EX0
E0:
        call promenaNijeOtvoreno
EX0:
        cmp trenutnoPolje, 1
        je E1
        jmp EX1
E1:
        call promenaOtvoreno
EX1:
        cmp trenutnoPolje, 2
        je E2
        jmp EX2
E2:
        call promenaZastava
EX2:

        ; pomera kursor
        mov eax, kursorBoja
        call SetTextColor
        dec kursorY
        call izracunajTrenutnoPolje
        cmp trenutnoPolje, 0
        je F0
        jmp FX0
F0:
        call promenaNijeOtvoreno
FX0:
        cmp trenutnoPolje, 1
        je F1
        jmp FX1
F1:
        call promenaOtvoreno
FX1:
        cmp trenutnoPolje, 2
        je F2
        jmp FX2
```

```
F2:
        call promenaZastava
FX2:
        mov eax, pozadinaBoja
        call SetTextColor
ENDIF3:

        ; ******************************
        ; dole
        ; ******************************


_if4:
        cmp al, 's'
        je THEN4
        jmp ENDIF4
THEN4:
        call izracunajTrenutnoPolje

        ; provera da nije kraj ekrana
        cmp kursorY, 8
        je ENDIF4

        ; vraca pozadinu
        cmp trenutnoPolje, 0
        je G0
        jmp GX0
G0:
        call promenaNijeOtvoreno
GX0:
        cmp trenutnoPolje, 1
        je G1
        jmp GX1
G1:
        call promenaOtvoreno
GX1:
        cmp trenutnoPolje, 2
        je G2
        jmp GX2
G2:
        call promenaZastava
```

```
GX2:

        ; pomera kursor
        mov eax, kursorBoja
        call SetTextColor
        inc kursorY
        call izracunajTrenutnoPolje
        cmp trenutnoPolje, 0
        je H0
        jmp HX0
H0:
        call promenaNijeOtvoreno
HX0:
        cmp trenutnoPolje, 1
        je H1
        jmp HX1
H1:
        call promenaOtvoreno
HX1:
        cmp trenutnoPolje, 2
        je H2
        jmp HX2
H2:
        call promenaZastava
HX2:
        mov eax, pozadinaBoja
        call SetTextColor
ENDIF4:

        ; ******************************
        ; otvaranje polja
        ; ******************************
_if5: cmp al, 32 ; spacebar
        je THEN5
        jmp ENDIF5
THEN5:
        call izracunajTrenutnoPolje

        ; provera da nije otvoreno polje ili zastava (onda nista ne radi)
        cmp trenutnoPolje, 1
        je ENDIF5
```

```
        cmp trenutnoPolje, 2
        je ENDIF5

        xor eax, eax
        mov al, kursorX
        mov dataX, al
        mov al, kursorY
        mov dataY, al
        call izracunavanje
        mov al, matricaMina[esi]
        cmp al, 1
        je lastLoop
        jmp lastLoop1
lastLoop:
        ; poruka za zavrsetak igre ***
        jmp lastLoop2
lastLoop1:
        ; otvara polje
        mov eax, kursorBoja
        call SetTextColor
        call promenaOtvoreno
        mov eax, pozadinaBoja
        call SetTextColor

        ; menja stanje polja u matrici (polje je sada otvoreno)
        xor eax, eax
        mov al, kursorX
        mov dataX, al
        mov al, kursorY
        mov dataY, al
        call izracunavanje
        xor ebx, ebx
        mov bl, 1
        mov matricaMinaPodatak[esi], bl

ENDIF5:

        jmp loopKey

lastLoop2:
```

```
        INVOKE ExitProcess,0
main ENDP
END main
```