Elektrotehnički fakultet
Katedra za elektroniku

Projekat iz Računarske elektronike:

Connect4 konzolna aplikacija - x86

Autori:

Ana Jelisijević 168/2014 **Luka Gavrić 23/2014**

21.6.2017.

Uvod

Predmet projekta je simulacija igre Connect4. U igri učestvuju dva igrača koji poteze naizmenično odigravaju.

Osnovni element igre su žetoni, dugmići, kamenčići - jednom rečju, ***ploops***-i. Postoji 42 ploops-a i podeljeni su bojom u dve grupe - svi ploops-i jednog igrača su iste boje. Na početku partije, igrači imaju po 21 ploops i ubacuju ih u tablu kroz otvore na njenom vrhu. Tabla ima 7 kolona u koje se mogu ubaciti ploops-i i mesta za po 6 ploops-a u svakoj koloni.

Pobeđuje igrač koji prvi sastavi niz od 4 svoja ploops-a na tabli. Niz se može sastaviti horizontalno, vertikalno i dijagonalno. Ukoliko nakon trošenja svih ploops-a pobednika nema, igra se završava nerešeno.

Projekat je izrađen u Assembler jeziku za Intel procesore x86 arhitekture. Pored posebnih procedura napisanih u kodu, korišćene su i procedure iz **Irvine32** biblioteke.

Implementirano rešenje

Programsko rešenje se sastoji od dve velike celine koje se naizmenično smenjuju - iscrtavanje i logičko proveravanje. U fazama iscrtavanja, u konzolnom prozoru se formira tabela sa numerisanim kolonama, ukazuje se na igrača koji je trenutno na potezu, kao i na boju njegovog ploops-a, iscrtava se ubačen ploops nakon svakog odigranog poteza i, konačno, na kraju igre se ispisuje ishod partije i eventualni pobednik.

Logičko proveravanje je srž Connect4 igre. Nakon svakog poteza, unutar ploops niza, koji je definisan sa

```
ploops BYTE 42 DUP(0),
```

upisuje se, na odgovarajuće mesto u nizu, broj 1 ili 2, koji u daljem toku igre označava igrača koji je na to mesto smestio svoj ploops.

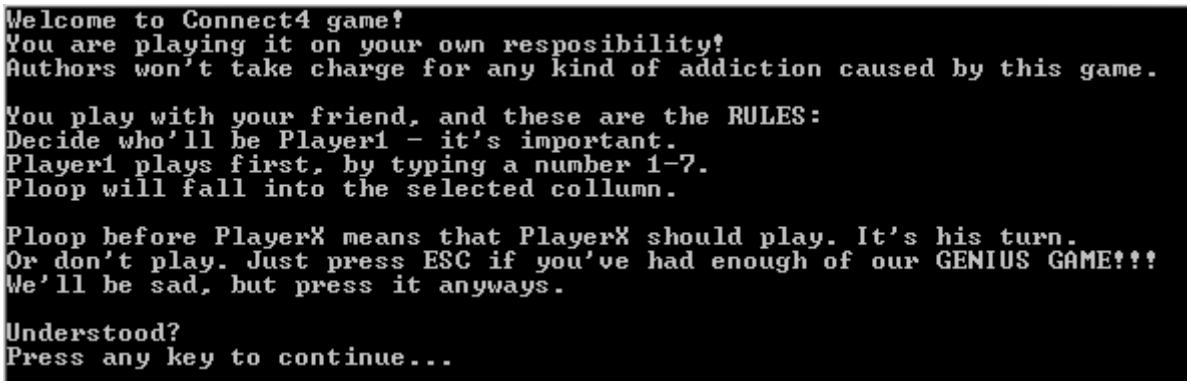
Nakon poteza, provera na 4 ploops-a u nizu se vrši u 4 etape. Provera po redu, po koloni, po jednoj i po drugoj dijagonali izvršavaju se respektivno. Ukoliko postoje 4 ploops-a istog igrača u nizu, provera poziva proceduru koja ispisuje pobednika i okončava igru.

Unutar programa uvedene su zaštite koje omogućuju da se igra izvršava po pravilima. Nije dozvoljen unos nijednog karaktera osim brojeva iz skupa {1,2,3,4,5,6,7}. Takođe, ukoliko je kolona puna, nema slaganja ploops-a van granica tabele. Posebna funkcija je dodeljenja ESC tasteru, nakon čije aktivacije se prekida igra.

U nastavku sledi objašnjenje i prikaz funkcionalnosti glavnog programa i svih formiranih procedura.

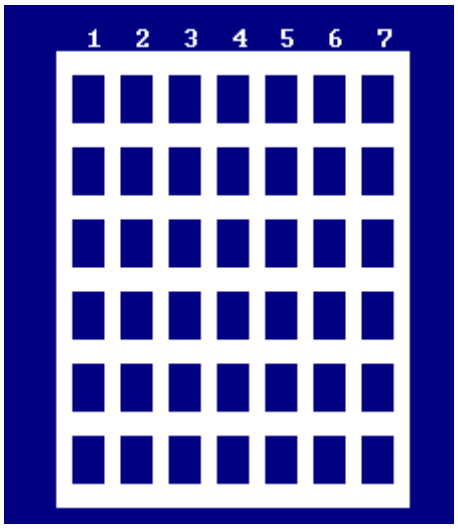
Glavni program

Na početku izvršavanja, program ispisuje poruke koje služe kao objašnjenje pravila igre i načina igranja.



Slika 1: Početni ekran sa porukama

Nakon toga se čeka na reakciju igrača, što je znak za početak iscrtavanja table.

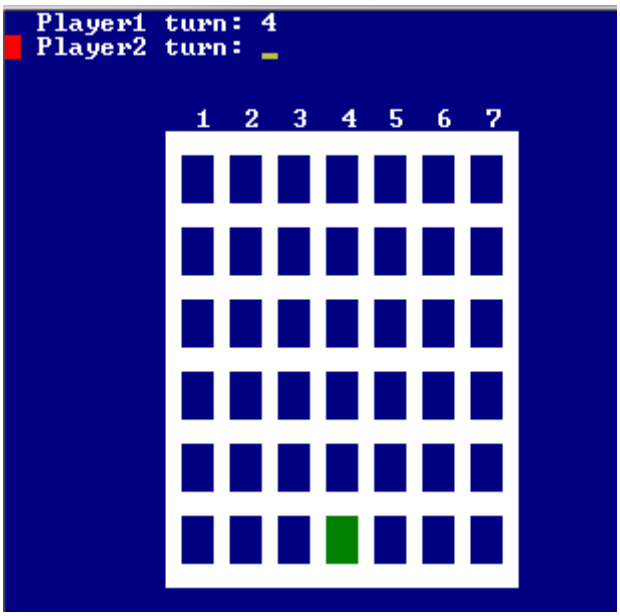


Slika 2: Tabla za igru

U narednom koraku, ispisuje se legenda. Ona označava igrača koji je trenutno na potezu i boju njegovih ploops-a.

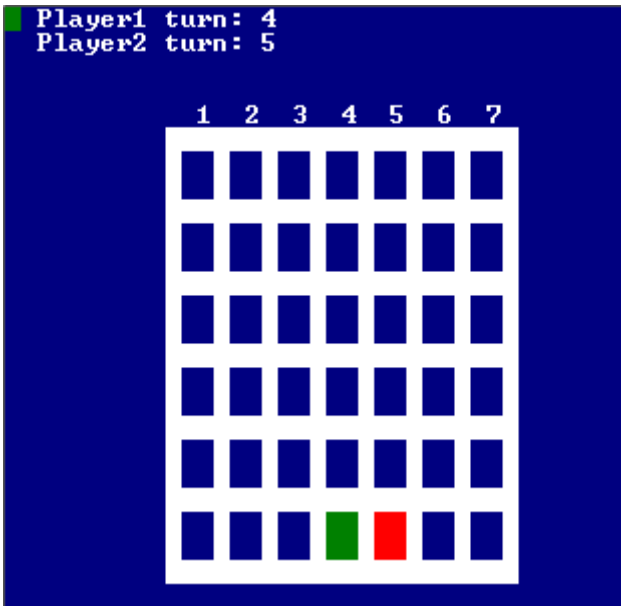
Slika 3: Legenda pri potezu igrača1 i igrača2

U ovom trenutku počinje igra i igrač1 je na potezu. On ubacuje ploop u kolonu po izboru. Odmah nakon odigravanja, crveni ploop ispred *Player2* labele govori igraču2 da je na potezu.



Slika 4: Stanje table nakon prvog poteza igrača1

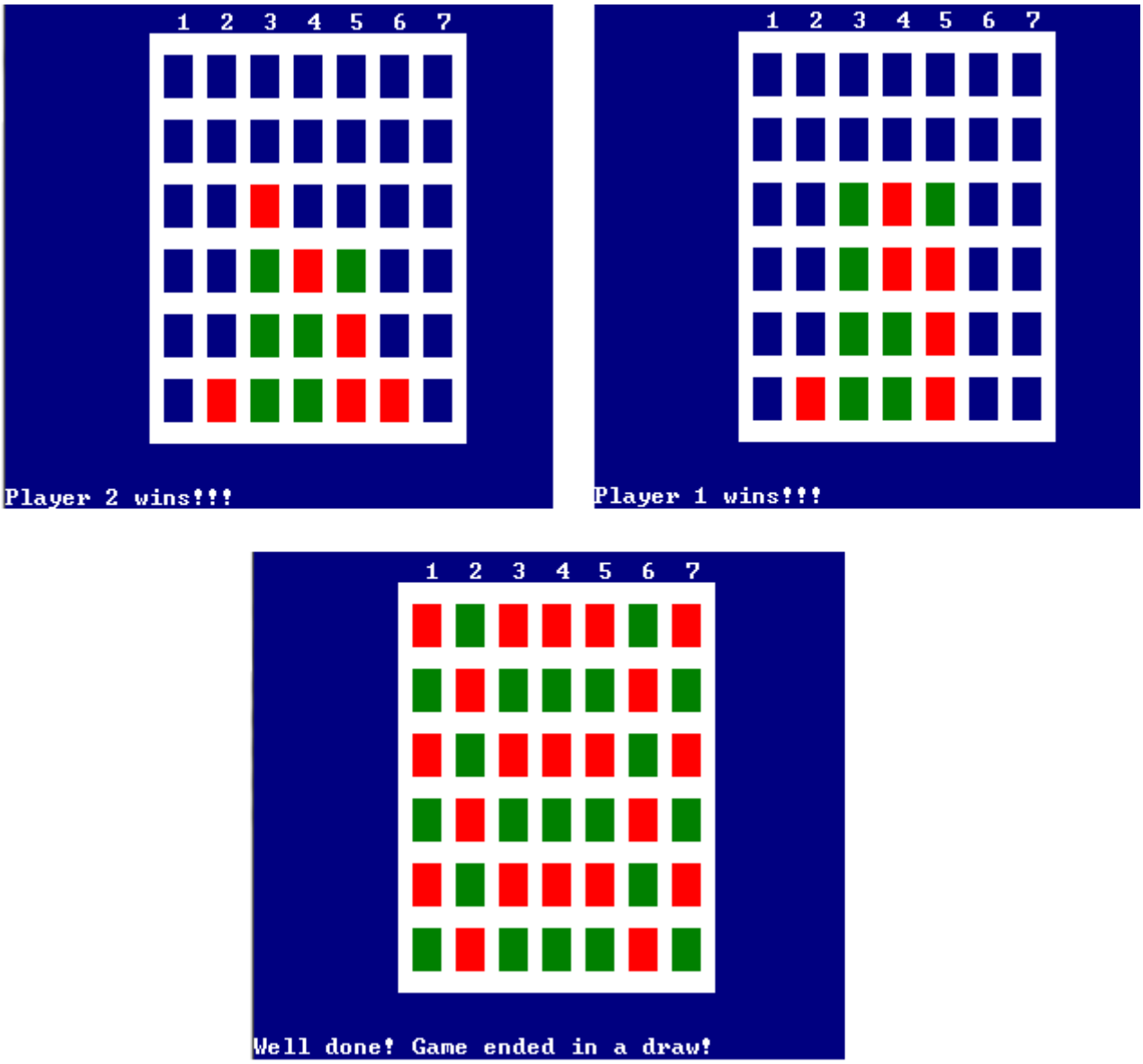
Nakon poteza igrača 2, igrač 1 ponovo igra. Ploop ispred labele *Player1* ima istu funkciju kao ploop ispred labele *Player2*.



Slika 5: Stanje table nakon prvog poteza igrača2

Igračima je uvek dostupan pregled poslednjih poteza, upisanih nakon labela *Player1 turn* i *Player2 turn*.

Nastavak igre se izvršava analogno - naizmeničnim potezima igrača 1 i 2 popunjava se tabela. Partija se može završiti pobedom jednog igrača ili nerešenim rezultatom.



Slika 6: Mogući ishodi igre

Glavni program menja promenljivu definisanu sa

```
currentPlayer BYTE 1
```

tako da ona ima vrednost 1 ako je na potezu igrač1, i 2, ako je na potezu igrač2. Koristeći jedinstvenu vrednost te promenljive, glavni program poziva procedure koje su zadužene za prihvatanje odgovarajućeg unosa sa tastature, iscrtavanje ploop-a odgovarajućeg igrača na dno kolone koja je odabrana unosom sa tastature i proveru na 4 ploop-a u nizu, te procedure koje osvežavaju stanje legende i pokazivača na igrača koji je trenutno na potezu.

U nastavku je dat funkcionalni prikaz svih procedura realizovanih unutar koda programa, pregled svih promenljivih čije vrednosti koriste i modifikuju, kao i moguće posledice njihovog izvršavanja.

Takođe, na kraju izveštaja priložen je i celokupan kod implementiranog programskog rešenja igre **Connect4**.

Procedure implementirane u kodu

DrawTable

Nakon što igrači pročitaju poruku na otvaranju igre, potvrđuju pritiskom na bilo koji taster na tastaturi da su spremni za igru. Glavni program reaguje na pritisak tastera tako što poziva **DrawTable** proceduru. Ona iscrtava brojeve koji označavaju kolone u koje se, pritiskom na prikazani broj na tastaturi, ubacuje ploop; takođe, ona koristi statičke promenljive:

```
verticalTop = 5
verticalSize = 19
verticalBottom = 24
horizontalTop = 10
horizontalSize = 22
```

za ograničavanje iscrtavanja linija pri crtanju tabele. Prvo se iscrtavaju vertikalne linije, pa potom horizontalne.

WriteBlueDot
WriteRedDot
WriteGreenDot

Nakon što se kursor pozicionira na odgovarajuće mesto, ove procedure iscrtavaju **dot** na mesto kursora. Jedan ploop se sastoji od četiri dot-a. Dot predstavlja *solid block* karakter sa ASCII kodom 0DBh. Ime procedure sadrži boju kojom će se iscrtati dot.

Ove procedure se pozivaju direktno iz glavnog programa kada se iscrtava legenda koja označava igrača trenutno na potezu, ali i pri iscrtavanju ploops-a unutar procedure **DrawPloop**.

PlayGame

Osnovna procedura koja kontroliše tok igre. Na samom početku prihvata unet karakter i vrši nekoliko provera. Ukoliko je pritisnut ESC taster, izvršenje celog programa se prekida. Ukoliko je unet bilo koji karakter osim brojeva iz skupa {1,2,3,4,5,6}, procedura će ponovo zatražiti unos broja.

Kada uneti karakter prođe obe provere, vrednost kolone koja je uneta upisuje se u *collumnOffset* promenljivu, a vrh kolone - mesto na koje pada ploops ubačen u određenu kolonu, računa se logikom *SearchForRowOffset* i upisuje u *rowOffset* promenljivu. Potom se vrednosti promenljive *currentPlayer* upisuje na mesto unutar *ploops* niza određeno vrednošću promenljivih *collumnOffset* i *rowOffset*.

Ukoliko se pri unosu broja kolone upiše broj već napunjene kolone, procedura će zatražiti ponovni unos validne kolone. Konačno, pre povratka iz ove procedure, poziva se **DrawPloop** procedura.

DrawPloop

Pre iscrtavanja ploop-a, ova procedura pozicionira kursor u skladu sa vrednostima promenljivih *collumnOffset* i *rowOffset*. Potom, dot po dot, iscrtava ploop boje odgovarajuće igraču koji je trenutno na potezu.

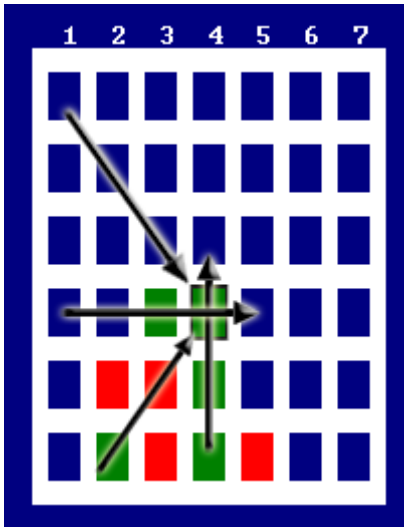
Check4inaCollumn
Check4inaRow
Check4inaMajorDiagonal
Check4inaMinorDiagonal

Ove četiri procedure se bave pretragom niza **ploops** i otkrivanjem 4 ploops-a u nizu istog igrača.

Prvi korak koji ove procedure izvršavaju su pozicioniranje na početak kolone/ reda/ Major dijagonale/ Minor dijagonale unutar kojih je unet poslednji ploop.

Nakon toga, svaka od četiri procedure pokreće svojstvenu logiku pomeranja kroz **ploops** niz, i tako se kreće kroz kolonu/ red/ Major dijagonalu/ Minor dijagonalu unutar koje je unet poslednji ploop i proverava da li je tim unosom napravljen niz od 4 ploops-a istog igrača.

Sledi ilustracija putanje kroz niz koju vrši logika pomeranja **Check4ina...** procedura na primeru odigravanja zelenog ploop-a (Player1) na treće mesto u koloni 4.



Slika 7: Ilustracija provere koju vrše procedure

U slučaju da je u bilo kojoj od ove četiri procedure otkriven niz od 4 ista ploops-a, poziva se **EndingResultWrite** procedura, a u suprotnom se program nastavlja povratkom iz procedure.

EndingResultWrite

Poziv ovoj proceduri upućuje samo neka od četiri procedure za proveru na 4 ploops-a u nizu. Nakon ispisivanja određene poruke o pobedniku, prekida se izvršenje programa.

Kod programa Connect4

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;                        CONNECT 4 - game of wisdom and strategy                         ;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

INCLUDE Irvine32.inc

verticalTop = 5                ;gornja granica pri iscrtavanju vertikalnih linija
verticalSize = 19              ;duzina vertikalnih linija
verticalBottom = 24            ;donja granica pri iscrtavanju vertikalnih linija
horizontalTop = 10             ;leva granica pri iscravanju horizontalnih linija
horizontalSize = 22            ;duzina horizontalnih linija

.386
.model flat, stdcall
.stack 4096
ExitProcess PROTO, dwExitCode:DWORD

.data
;promenljive potrebne za pracenje poteza
currentPlayer BYTE 1
currentResult BYTE 0
turnCount BYTE 42

;promenljive potrebne za iscrtavanje tabele
cnt DWORD 8
vertical_X BYTE 10             ;x position (column number)
horizontal_Y BYTE 10           ;y position (row number)

;promenljiva potrebna za smestanje unetog karaktera
char BYTE 0

;promenljive potrebne za proveru na 4 ploop-a u redu/koloni/dijagonali
xStraight BYTE 0
rowOffsetForChecking BYTE 0
collumnOffsetForChecking BYTE 0
diagonalOffset BYTE 0
currentPloopForChecking BYTE 0

;promenljive potrebne za proveru pobede i iscrtavanje ploops-a
ploops BYTE 42 DUP(0)
ploopsPointer BYTE 0
collumnOffset BYTE 0
rowOffset BYTE 0

;poruke koje se ispisuju na ekranu
introduction1 BYTE
"Welcome to Connect4 game!",0dh,0ah,
"You are playing it on your own resposibility!",0dh,0ah,
"Authors won't take charge for any kind of addiction caused by this game.",0dh,0ah,0dh,0ah,0
introduction2 BYTE
"You play with your friend, and these are the RULES:",0dh,0ah,
"Decide who'll be Player1 - it's important.",0dh,0ah,
"Player1 plays first, by typing a number 1-7.",0dh,0ah,
"Ploop will fall into the selected collumn.",0dh,0ah,0dh,0ah,0
introduction3 BYTE
"Ploop before PlayerX means that PlayerX should play. It's his turn.",0dh,0ah,
"Or don't play. Just press ESC if you've had enough of our GENIUS GAME!!!",0dh,0ah,
```

```
"We'll be sad, but press it anyways.",0dh,0ah,0dh,0ah,
"Understood?",0dh,0ah,0
inputMessageP1 BYTE " Player1 turn: ",0
inputMessageP2 BYTE " Player2 turn: ",0
numbers BYTE "  1  2  3  4  5  6  7",0
player BYTE "Player ",0
wins BYTE " wins!!!",0dh,0ah,0
drawMessage BYTE "Well done! Game ended in a draw!",0dh,0ah,0
outMessage BYTE "Game aborted/ended :( ",0dh,0ah,0

.code
;procedura za ispisivanje tabele
DrawTable PROC
        mov  dl,vertical_X
        mov  dh,verticalTop-1
        call Gotoxy
        mov edx,offset numbers
        call WriteString
        mov  dl,vertical_X
        mov  dh,verticalTop
        call Gotoxy
        mov  ecx,verticalSize
DrawVertical:
        mov  al,0DBh
        call Gotoxy
        call WriteChar
        inc  dh
        loop Drawvertical

        add vertical_X, 3
        mov ecx,verticalSize
        mov dl,vertical_X
        mov dh,verticalTop
        dec cnt
        mov eax, cnt
        sub eax, 0
        jnz DrawVertical

        mov dl,horizontal_Y
        mov dh,verticalTop
        mov ecx,horizontalSize
        mov cnt,7
DrawHorizontal:
        mov  al,0DBh
        call Gotoxy
        call WriteChar
        inc  dl
        loop DrawHorizontal

        add dh,3
        mov dl,horizontal_Y
        mov ecx,horizontalSize
        dec cnt
        mov eax, cnt
        sub eax, 0
        jnz DrawHorizontal

        ret
DrawTable ENDP
```

```
;procedura za prihvatanje kolone i smestanje ploops-a na njen vrh
PlayGame PROC
CharInput:
        mov rowOffset,0
        call ReadChar
        mov char,al      ;provera da li je unet esc karakter,
        sub al,27
        jnz ContinuePlay         ;ako nije, nastavlja se sa proverama i igrom
        mov dl,0                 ;ako jeste, izlazi se i ispisuje se abort message
        mov dh,verticalBottom
        add dh,2
        call Gotoxy
        mov edx,offset outMessage
        call WriteString
        INVOKE ExitProcess,0

        ContinuePlay:
        sub al,22                ;provera da li je unet broj izmedju 1 i 7
        js CharInput             ;ako je manji od 1
        sub al,7                 ;ili veci od 7
        jns CharInput    ;neophodno je ponovo uneti broj!!!
        mov al,char
        sub al,48
        mov collumnOffset,al
        movzx eax,collumnOffset
        imul eax,6
        mov ploopsPointer,al

SearchForRowOffset:
        movzx eax,ploopsPointer
        dec al
        mov ploopsPointer, al
        mov esi, OFFSET ploops
        add esi, eax
        add rowOffset,1
        mov al,rowOffset
        sub al,7
        jz CharInput
        mov al, [esi]
        sub al,0
        jnz SearchForRowOffset
        mov al,char
        call WriteChar
        mov esi, OFFSET ploops
        movzx eax,ploopsPointer
        add esi, eax
        mov al, currentPlayer
        mov [esi], al
        call drawPloop
        ret
PlayGame ENDP

;procedura za proveru na 4 ploop-a u koloni
Check4inaCollumn PROC
        mov cnt,7
        movzx eax,collumnOffset
        imul eax,6
        mov rowOffsetForChecking, al
NotStreak:
        mov xStraight,0
```

```
Streak:
        mov eax,cnt
        dec eax
        jz Returning
        mov cnt,eax
        movzx eax, rowOffsetForChecking
        dec eax
        mov rowOffsetForChecking,al
        add eax, OFFSET ploops
        mov esi, eax
        mov al, [esi]
        sub al, currentPlayer
        jnz NotStreak
        movzx eax,xStraight
        inc al
        mov xStraight,al
        sub al,4
        jnz Streak
        call EndingResultWrite

Returning:
        ret
Check4inaCollumn ENDP

;procedura za proveru na 4 ploop-a u redu
Check4inaRow PROC
        mov cnt,-1
        mov eax,6
        sub al,rowOffset
        mov collumnOffsetForChecking, al
NotStreak:
        mov xStraight,0
Streak:
        mov eax,cnt
        inc eax
        mov cnt,eax
        sub eax,7
        jz Returning
        add eax,7
        imul eax, 6
        add al, collumnOffsetForChecking
        add eax, OFFSET ploops
        mov esi, eax
        mov al, [esi]
        sub al, currentPlayer
        jnz NotStreak
        movzx eax,xStraight
        inc al
        mov xStraight,al
        sub al,4
        jnz Streak
        call EndingResultWrite

Returning:
        ret
Check4inaRow ENDP

;procedura za proveru na 4 ploop-a u glavnoj dijagonali
Check4inaMajorDiagonal PROC
        mov collumnOffsetForChecking, 0
```

```
        mov rowOffsetForChecking, 7
        movzx eax, collumnOffset
        dec eax
        imul eax,7
        mov diagonalOffset,al
        mov al, ploopsPointer
        inc al
        sub al, diagonalOffset
        mov diagonalOffset, al
        sub al,4
        jns Returning
        add al,6
        js Returning
        mov al,diagonalOffset
        dec al
        js HighDiagonals
LowDiagonals:
        mov collumnOffsetForChecking,0
        mov al,8
        sub al,diagonalOffset
        mov rowOffsetForChecking, al
        mov al,diagonalOffset
        mov currentPloopForChecking,al
        sub currentPloopForChecking,7
        jmp NotStreak
HighDiagonals:
        mov rowOffsetForChecking, 7
        movzx eax,diagonalOffset
        imul eax,-1
        mov collumnOffsetForChecking,al
        mov al,diagonalOffset
        imul eax,-6
        add     al,7
        mov currentPloopForChecking,al
        sub currentPloopForChecking,7
NotStreak:
        mov xStraight,0
Streak:
        mov al,collumnOffsetForChecking
        inc eax
        mov collumnOffsetForChecking,al
        sub eax,8
        jz Returning
        movzx eax,rowOffsetForChecking
        dec eax
        mov rowOffsetForChecking,al
        sub eax,-1
        jz Returning

        ;logika pomeranja
        movsx eax,currentPloopForChecking
        add eax,7
        mov currentPloopForChecking,al
        dec eax
        add eax, OFFSET ploops
        mov esi, eax
        mov al, [esi]
        sub al, currentPlayer
        jnz NotStreak
        movzx eax,xStraight
```

```
        inc al
        mov xStraight,al
        sub al,4
        jnz Streak
        call EndingResultWrite

Returning:
        ret
Check4inaMajorDiagonal ENDP

;;procedura za proveru na 4 ploop-a u sporednoj dijagonali
Check4inaMinorDiagonal PROC
        mov collumnOffsetForChecking, 0
        mov rowOffsetForChecking, 0
        movzx eax, collumnOffset
        dec eax
        imul eax,5
        mov diagonalOffset,al
        mov al, ploopsPointer
        inc al
        sub al, diagonalOffset
        mov diagonalOffset, al
        sub al,10
        jns Returning
        add al,6
        js Returning
        mov al,diagonalOffset
        sub al,7
        js HighDiagonals
LowDiagonals:
        mov rowOffsetForChecking,0
        mov al,diagonalOffset
        sub al,6
        mov collumnOffsetForChecking, al
        movzx eax,diagonalOffset
        imul eax,6
        sub eax,30
        mov currentPloopForChecking,al
        sub currentPloopForChecking,5
        jmp NotStreak
HighDiagonals:
        mov collumnOffsetForChecking,0
        mov al,8
        sub al,diagonalOffset
        mov rowOffsetForChecking,al
        mov al,diagonalOffset
        mov currentPloopForChecking,al
        sub currentPloopForChecking,5
NotStreak:
        mov xStraight,0
Streak:
        mov al,collumnOffsetForChecking
        inc eax
        mov collumnOffsetForChecking,al
        sub eax,8
        jz Returning
        movzx eax,rowOffsetForChecking
        inc eax
        mov rowOffsetForChecking,al
        sub eax,7
```

```
        jz Returning

        ;logika pomeranja
        movsx eax,currentPloopForChecking
        add eax,5
        mov currentPloopForChecking,al
        dec eax
        add eax, OFFSET ploops
        mov esi, eax
        mov al, [esi]
        sub al, currentPlayer
        jnz NotStreak
        movzx eax,xStraight
        inc al
        mov xStraight,al
        sub al,4
        jnz Streak
        call EndingResultWrite

Returning:
        ret
Check4inaMinorDiagonal ENDP

;procedura za ispis rezultata igre - ako je neko pobedio
EndingResultWrite PROC
        mov dl,0
        mov dh,verticalBottom
        add dh,2
        call Gotoxy
        mov edx,offset player
        call WriteString
        mov al,currentPlayer
        add al,48
        call WriteChar
        mov edx,offset wins
        call WriteString
        INVOKE ExitProcess,0
EndingResultWrite ENDP

;procedura za iscrtavanje ploop-a na mestu odredjenom
                        ;collumnOffset i rowOffset promenljivama
drawPloop PROC
        mov dl,0
        mov dh,0
        call Gotoxy

        mov al,collumnOffset
        dec eax
        imul eax,3
        add eax,horizontalTop
        inc eax
        mov dl,al

        mov al,rowOffset
        dec eax
        imul eax,3
        imul eax,-1
        add eax,verticalBottom
        sub eax,2
        mov dh,al
```

```
        call Gotoxy

        movzx eax, currentPlayer
        sub eax,1
        jz casePlayer1

        call WriteRedDot
        add dl,1
        call Gotoxy
        call WriteRedDot
        sub dh,1
        call Gotoxy
        call WriteRedDot
        call Gotoxy
        sub dl,1
        call Gotoxy
        call WriteRedDot
        ret
caseplayer1:
        call WriteGreenDot
        add dl,1
        call Gotoxy
        call WriteGreenDot
        sub dh,1
        call Gotoxy
        call WriteGreenDot
        sub dl,1
        call Gotoxy
        call WriteGreenDot
        ret
drawPloop ENDP

;tri procedure za iscrtavanje obojenih kvadratica
WriteRedDot PROC
        mov eax,lightRed + (blue * 16)
        call SetTextColor
        mov  al,0DBh
        call WriteChar
        mov eax,white + (blue * 16)
        call SetTextColor
        ret
WriteRedDot ENDP

WriteBlueDot PROC
        mov eax,blue + (blue * 16)
        call SetTextColor
        mov  al,0DBh
        call WriteChar
        mov eax,white + (blue * 16)
        call SetTextColor
        ret
WriteBlueDot ENDP

WriteGreenDot PROC
        mov eax,green + (blue * 16)
        call SetTextColor
        mov  al,0DBh
        call WriteChar
        mov eax,white + (blue * 16)
```

```
        call SetTextColor
        ret
WriteGreenDot ENDP

;glavni program pocinje ovde
main PROC
        mov  edx,offset introduction1
        call WriteString
        mov  edx,offset introduction2
        call WriteString
        mov  edx,offset introduction3
        call WriteString
        call WaitMsg
        mov eax,white + (blue * 16)
        call SetTextColor
        call Clrscr
        call DrawTable

player1:
        mov currentPlayer,1
        mov dl,0
        mov dh,0
        call Gotoxy
        call WriteGreenDot
        mov dl,0
        mov dh,1
        call Gotoxy
        call WriteBlueDot

        mov dl,1
        mov dh,0
        call Gotoxy
        mov edx,offset inputMessageP1
        call WriteString
        call PlayGame
        call Check4inaCollumn
        call Check4inaRow
        call Check4inaMajorDiagonal
        call Check4inaMinorDiagonal

        mov al, turnCount
        sub al,1
        mov turnCount,al
        sub al,0
        jz endingProcess

player2:
        mov currentPlayer,2
        mov dl,0
        mov dh,0
        call Gotoxy
        call WriteBlueDot
        mov dl,0
        mov dh,1
        call Gotoxy
        call WriteRedDot

        mov dl,1
        mov dh,1
        call Gotoxy
```

```
        mov edx, offset inputMessageP2
        call WriteString
        call PlayGame
        call Check4inaCollumn
        call Check4inaRow
        call Check4inaMajorDiagonal
        call Check4inaMinorDiagonal

        mov al, turnCount
        sub al,1
        mov turnCount, al
        sub al,0
        jnz player1

;deo koda koji se izvrsava kada nema pobednika nakon 42 poteza
endingProcess:
        mov dl,0
        mov dh, verticalBottom
        add dh,2
        call Gotoxy
        mov edx, offset drawMessage
        call WriteString
        INVOKE ExitProcess,0
main ENDP

END main

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;                end of   CONNECT 4 - game of wisdom and strategy                     ;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```