המשחק

# מספרים אקראיים

|  |
| --- |
| המטרה: בכל שינוי כיוון של הדמות נצייר את הדופן של המשחק מחדש כשמיקום הקצוות המשוננים נקבע בכל פעם מחדש בצורה אקראית  נחלק את המשימה לכמה חלקים:  1 – כתיבת פרוצדורה שמציירת את הדופן ללא הקצוות המשוננים  2 – כתיבת פרוצדורה המגרילה מספר אקראי בין 0 ל 15  3 – שימוש בפרוצדורה על מנת להדפיס 5 סימני קצה משונן על הקן המצויר  4 – שילוב הפרוצדורה בתוכנית |

## הפסיקות בתכנית שלנו

|  |  |
| --- | --- |
| פקודה להדפסת מחרוזת  (שם משתנה המחרוזת msg) | mov dx, offset msg  mov ah, 9h  int 21h |
| פקודה לקליטת תו מהמקלדת  לא מדפיסה את תו ולא ממשיכה בהוראות על לקבלת תו. | mov ah, 0h  int 16h  הפסיקה מחזירה בתוךal את קוד ה־ ASCII של התו שנמצא בראש הבאפר ובתוךah את ה־code scan שלו. בנוסף, הפסיקה "מנקה" את התו מהבאפר |
| פקודה לקליטת תו מהמקלדת  לא מדפיסה את תו, אם לא הוקש על מקש ממשיכה בפקודות התכנית. | mov ah, 1h  int 16h  הפסיקה מדליקה את ה zero flag– אם יש תו מוכן לקריאה (zf = 1) , ומכבה אם אין תו מוכן (zf = 0).  אם יש תו מוכן, al ו-ah יקבלו את ערכי ה־ASCII וה־code scan של התו. הפסיקה אינה "מנקה" את התו מהבאפר |
| כניסה לתצוגה גרפית  25\*40 | mov ax, 13h  int 10h |
| יציאה מתצוגה גרפית  80\*25 | mov ax, 2h  int 10h |
| מיקום הסמן על המסך | mov dh, 6 ; row (y)  mov dl, 7 ; column (x)  mov bh, 0 ; page number  mov ah, 2  int 10h |
| ציור תו על המסך במיקום הסמן | mov ah, 9  mov al, 2 ; al = character to display  mov bx, 00Eh ; bh = Background bl = Foreground  mov cx, 1 ; cx = number of times to write character  int 10h |
| השהייה  המתנה של פרק זמן קצר (חמישית השנייה) | mov cx, 03h ;High Word  mov dx, 4240h ;Low Word  mov al, 0  mov ah, 86h ;Wait  int 15h |
| קריאה של תו במיקום הסמן.  שמירת התו למשתנה  [screenChr] | mov bl, 0h ; Page=1  mov ah, 08h ; Read character function  int 10h ;return the character to al  mov [screenChr], al |

## תכנית random

תרגול

# כתיבת פרוצדורה שמציירת את הדופן ללא הקצוות המשוננים

נכתוב תחילה תכנית קטנה המכילה רק את הפרוצדורה שמציירת את דופן ימין או שמאל

פתחו את הקובץ base.asm ושימרו אותו בשם אחר. בראש הקובץ הוסיפו p186

|  |
| --- |
| IDEAL  MODEL small  STACK 100h  **p186**  DATASEG |

1. נגדיר 2 משתנים המציינים את מיקום התו אותו נרצה לצייר. נאתחל אותם לקואורדינטות של הקצה העליון של הקו

fence\_x\_coord db 2

fence\_y\_coord db 5

1. נגדיר משתנה שבו נשמור את התו אותו נרצה לצייר

tav db '|'

1. נכתוב את פרוצדורה השמה את הסמן בקואורדינטות (fence\_x\_coord, fence\_y\_coord) נקרא לה fence\_cursor\_location

תזכורת: הפסיקה המזיזה את הסמן למיקום (fence\_x\_coord, fence\_y\_coord):

mov dh, ]fence\_y\_coord[ ; row

mov dl, ]fence\_x\_coord[ ; column

mov bh, 0 ; page number

mov ah, 2

int 10h

1. נכתוב פרוצדורה שמדפיסה את המשתנה tav במיקום הסמן ⇦ draw\_tav

תזכורת: פסיקה המדפיסה תו במיקום הסמן.

mov ah, 9

mov al, [tav] ; al = character to display

mov bx, 00Fh ; bh = Background bl = Foreground

mov cx, 1 ; cx = number of times to write character

int 10h

1. נכתוב פרוצדורה draw\_line שמציירת קו שמתחיל בנקודה (fence\_x\_coord, fence\_y\_coord) ואורכו 17 תווים

proc draw\_line

pusha

mov cx, 17 ; loop

mov [tav], ‘|’ ; draw line of 17 🡪 |

; draws a line of length 17 starting at fence\_x\_coord fence\_y\_coord

draw\_fence:

call fence\_cursor\_location

call draw\_tav

לולאה בת 17 סיבובים

בכל סיבוב

* נזמן את fence\_cursor\_location
* נזמן את draw\_tav
* נגדיל את fence\_y\_coord ב 1

inc [fence\_y\_coord]

loop draw\_fence

popa

ret

endp draw\_line

1. בתוכנית הראשית: היכנסו למוד גראפי וזמנו את draw\_line

תזכורת: כניסה למוד גראפי

mov ax, 13h

int 10h

1. הוסיפו בסוף התוכנית פסיקה שמחכה לתו על מנת שהתוכנית תעצור לפני הסוף

תזכורת: המתנה לתו וקריאתו

mov ah, 0h

int 16h

וחזרו לתצוגת טקסט 80\*25

mov ax, 2h

int 10h

## -------------------------------------------

## קוד התכנית:

IDEAL

MODEL small

STACK 100h

**p186**

DATASEG

fence\_x\_coord db 2

fence\_y\_coord db 5

tav db '|'

; --------------------------

CODESEG

proc fence\_cursor\_location

pusha

mov dh, [fence\_y\_coord] ; row

mov dl, [fence\_x\_coord] ; column

mov bh, 0 ; page number

mov ah, 2

int 10h

popa

ret

endp fence\_cursor\_location

proc draw\_tav

pusha

mov ah, 9

mov al, [tav] ; al = character to display

mov bx, 00Fh ; bh = Background bl = Foreground

mov cx, 1 ; cx = number of times to write character

int 10h

popa

ret

endp draw\_tav

proc draw\_line

pusha

mov cx, 17

mov [tav], '|'

; draws a line of length 17 starting at fence\_x\_coord fence\_y\_coord

draw\_fence\_loop:

; Enter Grafic mode 40 \* 25

mov ax, 13h

int 10h

**call draw\_line**

; Waite for charcter

mov ah, 0h

int 16h

; return to text mode 80\*25

mov ax, 2h

int 10h

exit:

mov ax, 4c00h

int 21h

END start

call fence\_cursor\_location

call draw\_tav

inc [fence\_y\_coord]

loop draw\_fence\_loop:

popa

ret

endp draw\_line

start:

mov ax, @data

mov ds, ax

# כתיבת פרוצדורה המגרילה מספר אקראי בין 0 ל 15

|  |
| --- |
|  |
|  |
|  |
|  |

1. נגדיר משתנה rnd בגודל byte אליו נשמור את המספר המוגרל בפרוצדורה
2. בכל הגרלה של מספר אקראי נעשה xor עם תא זיכרון אליו מפנה bx על מנת ליצור שונות בין המספרים המוגרלים.

נגדיר משתנה bx\_saver בגודל 2 בתים. למשתנה זה נשמור בכל קריאה לפרוצדורה את הכתובת בזיכרון אליה נרצה ש bx יפנה אותנו. נאתחל את bx\_saver ל 0. כלומר נרצה שבסיבוב הראשון נשתמש בתא זיכרון 0.

1. יצירת מספר אקראי

proc random

; generates a random number and keeps it in rnd

pusha

mov bx, [bx\_saver]

הגרלת המספר האקראי

* נעביר את הכתובת של תא כלשהו בזיכרון מ bx\_saver ל bx על מנת שנוכל לגשת אליו.
* נשלוף מספר מהזיכרון של השעון [[40h:6ch ל ax
* נבצע xor אם התא בזיכרון עליו מצביע bx.
* נגדיל את bx\_save ב ax.
* נבצע and עם 0fh (15 בבסיס 16) כדי לקבל מספר קטן מ 15
* נשמור את התוצאה ב rnd

; put segment number in register es

mov ax, 40h

mov es, ax

; move random number to ax

mov ax, [es:6Ch]

xor ax, [bx]

add [bx\_saver], ax

and al, 0Fh ; check if number is < 15

mov [rnd], al

popa

ret

endp random

## ------------------------------

## משתנים

rnd db ?

bx\_server dw 0

## הפעולה

proc random

pusha

mov bx, [bx\_server]

; Put segment number in

; register es

mov ax, 40h

mov es, ax

; Mov random number to ax

mov ax, [es:6ch]

xor ax, [bx] ; to get different number

add [bx\_server], ax ; change the "xor" address

and al, 0fh ; to get random number < 15

mov [rnd], al

popa

ret

endp random

# שימוש בפרוצדורה על מנת להדפיס 5 סימני קצה משונן על הקן המצויר

1. סוף כל סוף נוכל לכתוב את הפרוצדורה draw\_fence. בפרוצדורה נקרא קודם כל ל draw\_line כדי שיצוייר קו ישר ומייד אחר כך בלולאה נגריל 5 מספרים אקראיים.   
   כל אחד מהם מהמספרים האקראיים יהיה קואורדינטת y של תו משונן. כשנצייר את התו '<' במיקום שהוגרל הוא ידרוס-ימחק את התו '|' שצויר שם קודם

proc draw\_fence

; Draws a line of length 17 starting at fenc\_x\_coord fence\_y\_coord

: The line contains 5 obstacles

pusha

|  |  |
| --- | --- |
| call draw\_line | * נזמן את draw\_line לציור הקו הישר |
| mov [tav], '<' | * נכניס ל tav את הסימן המשונן בו נבחר |
| ; loop for drawing random obstacles  mov cx, 5 | * נרצה 5 סימנים משוננים אותם נגריל בלולאה. לכן נכניס ל cx – מונה הלולאה - 5 |
| draw\_obstacles: | * נפתח את הלולאה ב label |
| call random | * נזמן את random על מנת להגריל מספר נזכור שהתוצאה ב rnd |
| mov dl, [rnd]  add dl, 5  mov [fence\_y\_coord], dl | * נגדיל את rnd ב 5 – משום שהקו מתחיל בשורה 5 ונעביר אותו ל fence\_y\_coord |
| call fence\_cursor\_location  call draw\_tav | * נצייר את התו |
| loop drw\_obstacles | * נחזור לתחילת הלולאה |

popa

ret

endp draw\_fence

**זמנו את הפרוצדורה הריצו ובדקו את התוצאה.**

# שילוב הפרוצדורה במשחק

## קובץ bg12

1. העתיקו את המשתנים והפרוצדורות מהתוכנית שכתבתם למשחק
2. הוסיפו את המשתנה obstacle שגודלו - byte.   
   המשתנה יחזיק את התו לסימן המשונן.   
   אם אנחנו בצד ימין נשים ב obstacle ⇦ '>'   
   ואם אנחנו בצד שמאל נשים ב obstacle ⇦ '<'.   
   אתחלו את obstacle ל ‘<’

|  |
| --- |
| proc **draw\_left\_fence**  ; Sets parameters for left fence  pusha  mov [fence\_x\_coord] , 6  mov [fence\_y\_coord], 5  mov [obstacle], '<'  call draw\_fence  popa  ret  endp drawLeftFence |

1. נוסיף 2 פרוצדורת drw\_right\_fence ו **draw\_left\_fence**   
   שיקבעו את הערך של המשתנים obstacle ו - fence\_y\_coordinate   
   לפני הקריאה ל draw\_fence.

שימו לב 🎔: בציור הגדר הימנית המיקום של [fence\_x\_coord] = 26

1. עלינו לשנות את התו המצייר את התווים המשוננים [obstacle] כאשר מציירים את הגדר בצד ימין ובצד שמאל בפרוצדורה המציירת את הגדר.

נשתמש ברגיסטר dl כדי להעביר את ערך התו מהמשתנה [obstacle] למשתנה [tav].

הוסיפו לפרוצדורה המציירת את הגדר את שתי השורות המעבירות ערך ממשתנה למשתנה.

proc draw\_fence

pusha

call draw\_line

**mov dl, [obstacle]**

**mov [tav], dl**

mov cx, 5

; loop for drawing random obstacles

drw\_obstacles:

call random

mov dl, [rnd]

add dl, 5 ; add the random number 5 since the fence starts at 5

mov [fence\_y\_coord], dl

call fence\_cursor\_location

call draw\_tav

loop drw\_obstacles

popa

ret

endp draw\_fence

1. זמנו מתוך הפרוצדורה changeDir המשנה את כיוון הדמות את drw\_right\_fence ו - draw\_left\_fence כך שכשהדמות תשנה כיוון הדופן יודפס מחדש כשהשיניים במיקומים אקראיים.

; change character direction from 'r' to 'l' and from 'l' to 'r'

proc changeDir

pusha

cmp [direction], 'r'

je left

mov [direction], 'r'

add [x\_coord], 2

**call draw\_left\_fence**

jmp endChangeDir

left:

**call draw\_right\_fence**

mov [direction], 'l'

sub [x\_coord], 2

endChangeDir:

; sets character position in opposite direction

call setCursorePosition

popa

ret

בהצלחה

endp changeDir