המשחק

# קראית תו במיקום הסמן

## 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, [y\_cord] ; row  mov dl, [x\_cord] ; column  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, [color] ; 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.  שמירה התו למשתנה  [screenChr] | mov bh, 0h ; Page=1  mov ah, 08h ; Read character function  int 10h ;return the character to al  mov [screenChr], al |

תרגול

שמרו את התכנית שכתבתם בשיעור קודם בשם חדש והמשיכו לעבוד בקובץ החדש.

במשחק שלנו אם הדמות נוגעת בקצוות המחודדים שבמסגרת המשחק השחקן נפסל.

לשם כך יש לבדוק האם במיקום הבא אליו השחקן נמצא אחד מהסימנים: '<' ,'>', 'v' , '^'.

אם שחקן נוגע באחד מהתווים האלו יש פסילה אחרת הוא יכול להתקדם ולהמשיך לשחק.

ראשית ניצור משתנה אשר ישמור את התו במיקום אליו השחקן רוצה להגיע.

screenChr db 0

נוסיף את הפסיקה שלנו שקוראת את התו במיקום הסמן:

mov bh, 0h ; Page=1

mov ah, 08h ; Read character function

int 10h ;return the character to al

mov [screenChr], al

ניצור פעולה readScreenChr ובתוכה נזמן את הפסיקה.

נזמן את הפעולה לאחר מיקום הסמן במקום החדש בו נרצה לצייר את השחקן בלולאה הראשית של המשחק.

בנוסף נגדיל את הערך של רגיסטר cx ל – 20 כדי שנספיק "לשחק"

## בדיקת המשתנה screenChr

**נריץ את התוכנית ב - dt**

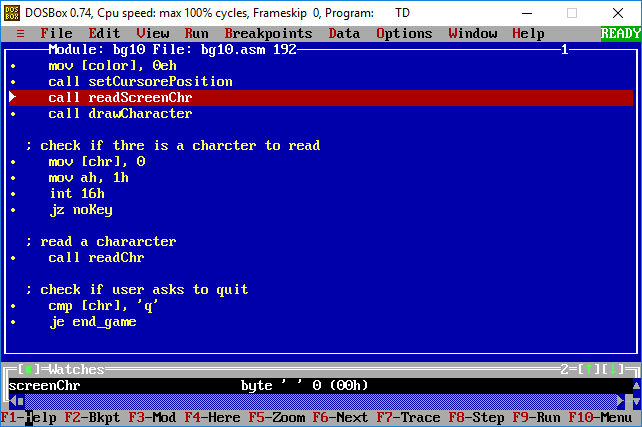
tasm /zi <file\_name>

tlink /v <file\_name>

td <file\_name>

**נעקוב אחרי השינויים בערכו של screenChr בעזרת ה Turbo Debugger**

* שים "נקודת עצירה” בזימון של הפעולה readScreenChr הקראת את התו במיקום הסמן – נעמוד על השורה בה נרצה שהתוכנית תעצור ונלחץ **F2** - השורה תיצבע באדום והתוכנית תעצור כשתגיע אליה
* בחלון נרד לאזור שבו ניתן לראות את המשתנים ונבקש לראות את המשתנה screenChar



* נריץ את התכנית עם F9 – כלומר הרצה רגילה שאינה בצעדים.
* התכנית תיעצר שנגיע לנקודת העצירה. נקבל שוב את מסך ה – td בדקו מהו ערך המשתנה.
* הריצו שוב את התכנית עם F9 עד לנקודת העצירה כמה פעמים עד שתראו   
  שהתו במשתנה הוא **'^'**.
* הרצה עם F9 תריץ את התכנית עד שנגיע שוב לנקודת העצירה.

## תנאי האם נגענו ב"שפיץ"

* נוסף תנאי אשר יבדוק האם נגענו באחד מתווי המסגרת המחודדים:
* > או < אן V אן ^
* מאחר ואנו רוצים לשים את הבדיקה בתוך פעולה נוסיף משתנה שישמור לנו האם יש הפסד - [lost]
* נאתחל את המשתנה לאפס
* אם יש נגיעה נשנה אותו ל – 1.

lost db 0

**אלגוריתם הפעולה:**

בתחילת הבדיקה נאפס את המשתנה – lost

נבדוק האם התו האו אחד מארבעת האפשרויות < או > או V או ^

אם כן נקפוץ לתווית **youLost**

קפיצה לתווית – **endCheckScrnChr** (סוף הבדיקה)

תווית – **youLost**

שינוי ערך משתנה [lost] ל - 1

תווית - **endCheckScrnChr**

סוף פעולה

; check the content of screenChr, if it is one of <>^v ends game

proc checkScrnChr

pusha

mov [lost], 0

cmp [screenChr], '<'

je youLost

cmp [screenChr], '>'

je youLost

cmp [screenChr], 86 ; V

je youLost

cmp [screenChr], 94 ; ^

je youLost

jmp endCheckScrnChr

youLost:

mov [lost], 1

endCheckScrnChr:

popa

ret

endp checkScrnChr

## נוסיף את הבדיקה האם נגענו בתו מחודד לתכנית הראשית?

**לולאה ראשית - mainGameLoop**

נזמן את פעולת הבדיקה – checkScrnChr

לאחר הפעולה הקראת מהו התו במיקום הסמן

נבדוק בסיום הפעולה האם [lost] = 1

אם כן נקפוץ לסוף המשחק תווית - end\_game

mov cx, 20

mov [chr], 0

mainGameLoop:

; delete character => draw blank - character at cursor position

mov [color], 0

call drawCharacter

; increase x\_cord by 1

inc [x\_cord]

cmp [chr], ' '

je jmpCar

add [y\_cord], 2

jmp continue

jmpCar:

sub [y\_cord], 2

continue:

; set cursor location & draw smiley - ascii 2 at cursor position

call setCursorePosition

call readScreenChr

**call checkScrnChr**

**cmp [lost], 1**

**je end\_game**

mov [color], 0Eh

call drawCharacter

; check if there is a character to read

mov [chr], 0

mov ah, 1h

int 16h

jz noKey

; read a chararcter

call readChr

; check if user asks to quit

cmp [chr], 'q'

je end\_game

noKey:

call delay

loop mainGameLoop

end\_game:

; text mode

mov ax, 2h

int 10h