# משתנים ופקודת mov

### מערכים

הצהרה על משתנים נעשית בסגמנט הנתונים – Data Segment.

name type value

num db 5

### הצרה על מערך

מערך הוא אוסף של איברים בעלי גודל זהה. לכל איבר במערך שי אינדקס המצביע על מיקומו בערך.

**הצהרה על מערך**

name type value1, value2, value3

array1 db 2, 4, 6

array2 db 5 dup (0)

יוצר מערך של 5 איברים בגודל byte המאותחלים בערך 0

array3 dw 4 dup (?)

יוצר מערך של 4 איברים בגודל word שאין להם ערכים התחלתיים

array4 db 6 dup ('X')

יוצר מערך של 6 איברים בגודל byte המכילים את האות X.

לכל איבר במערך יש אינדקס המצביע על מיקומו במערך והכתובת שלו:

array1 db 2, 4, 6

|  |  |  |  |
| --- | --- | --- | --- |
| כתובת בזיכרון | array1 | array1 + 1 | array1 + 2 |
| אינדקס במערך | 0 | 1 | 2 |
| ערך | 2 | 4 | 6 |

### מצביע bx

bx הוא רגיסטר מיוחד. הוא משמש בעיקר כמצביע לכתובת בזיכרון.

### ****פקודת offset****

פקודת ה – offset מעתיקה לרגיסטר bx את הכתובת של תחילת המערך,   
האיבר הראשון במערך ⇦ אינדקס 0.

דרך נפוצה לטיפול במערכים היא להעתיק ל-bx את כתובת תחילת המערך:

mov bx, offset array

לרגיסטר bx תועתק כתובת תחילת המערך,   
כתובת האיבר הראשון במערך ⇦ האיבר שנמצא באינדקס 0.

mov al, [bx+2]

העתקת האיבר באינדקס 2 של המערך לתוך al:

**רשימת פסיקות - מחסן פקודות**

|  |  |
| --- | --- |
| פקודה להדפסת מחרוזת  (שם משתנה המחרוזת msg) | mov dx, offset msg  mov ah, 9h  int 21h |
| פקודה להדפסת תו על המסך | mov dl, 'x' ; print x  mov ah, 2  int 21h |
| פקודה למעבר שורה | mov dl, 0ah  mov ah, 2h  int 21h |
| פקודה לקליטת מספר מהמשתמש ושמירתו לרגיסטר al | mov ah, 1h  int 21h  sub al, '0' |
| פקודה להדפסת ערך השמור ברגיסטר **al** בתרגום הערך למספר בבסיס 10 על המסך  Ten הוא שם משתנה המכיל את הערך 10  ten db 10 | mov ah, 0  div [ten]  add ax, '00'  mov dx, ax  mov ah, 2h  int 21h  mov dl, dh  int 21h |

**לולאה:**

mov cx, 5

SomeLabel:

; פקודות לביצוע

loop SomeLabel

# תרגילים

1. הגדירו מערך של 8 בתים שמאותחלים לערך 5.

הגדירו מערך של 8 מילים שמאותחלות לערך 5.

הגדירו מערך של 8 בתים שמאותחל לערך 'e'

השוו את תמונת הזיכרון בשלושת המערכים!   
  
**הריצו את התכנית ב – TD**

tasm /zi file\_name

tlink /v file\_name

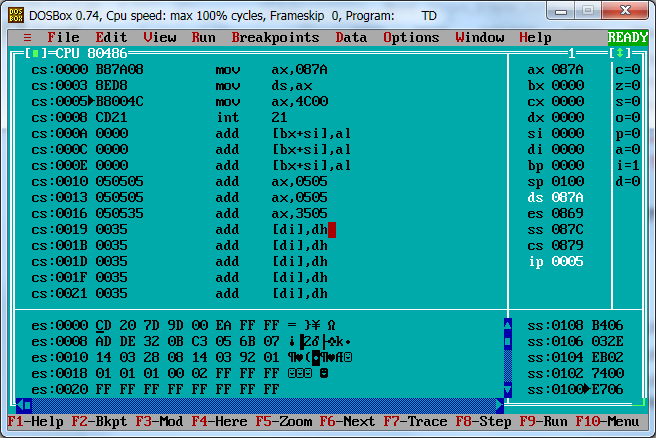
td file\_name

**בדקו ב – td בתצוגה של ה – cpu את הזיכרון**

הריצו את שתי השורות הראשונות של הקוד.

בעזרת המקש F8.

בעזרת המקש Tab עברו לאזור של ה – DataSeg



ראו שמסגרת מקטע הנתונים נצבעה בכחול

הקישו על מקש

**Ctrl + g**

בחלון שנפתח

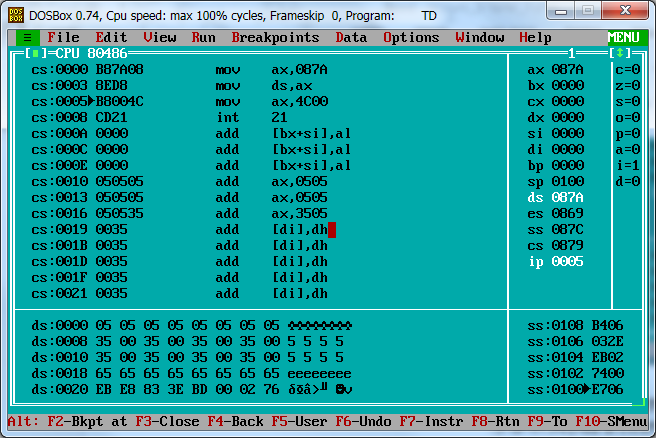
הקישו ds:00 והקישו Enter.

תוכלו לראות את הנתונים במקטע הזיכרון.

ראו את השורה הראשונה - 8 תאי זיכרון המכילים את הערך 5

שתי השורות הבאות- 4 מילים בכל שורה מכלים את התו – 5

ובשורה הרביעית – 8 תאי זיכרון בהם התו e.



1. הצהירו על מערך בן 4 איברים.

השמו בו ערכים בסדר עולה מ 1 עד 4.

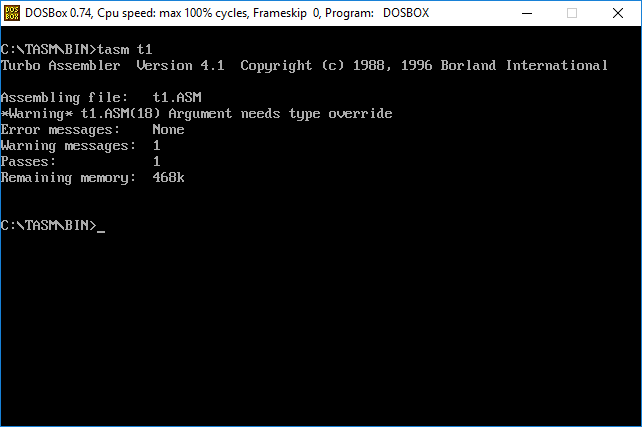
הגדילו כל תא ב – 10h. הפקודה לחיבור שני אופרנדים היא: add destination, source

כאשר רוצים להוסיף למשתנה במערך, שפנינו לכתובת שלו בעזרת רגיסטר [bx], מספר קבוע.

add [bx], 10h

בהרצת התוכנית נקבל הודעת שגיאה.

Argument needs type override



הודעה זו היא מאחר והאסמבלי לא יודע האם התא שהפנינו אליו את הוא בגודל מילה, בית או מילה כפולה. לכן יש להודיע לו לאיזה גודל משתנה אנו רוצים לפנות וזאת נעשה ע"י פקודת:

byte ptr / word ptr

add [byte ptr bx], 10h

בדקו את התכנית ב – TD

הוסיפו לתכנית פסיקה להדפסת ערך השמור ברגיסטר al ופסיקה המדפיסה למסך רווח והציגו את המערך החדש למסך.

1. הצהירו על מערך של 5 אותיות.

השמו בו את האותיות: 'A',B','C','D', 'E',.

הפרש בייצוג של אותיות "גדולות" ו"קטנות" הוא 32. הגדילו כל אות ב - 32 והדפיסו את המערך.

שימו לב 🎔, הפסיקה של הדפסת תו **שונה** מהדפסה של מספר השמור ברגיסטר al.



1. כתבו תכנית המחשבת את סכום האיברים במערך.  
   צרו מערך של חמישה בתים וקבעו להם ערכים, סכמו את האיברים במערך והציגו את התוצאה.   
   **הערה**: שימו לב שסכום האיברים במערך לא יעלה על 255.
2. כתבו תכנית בה מערך של מספרים, בעל 8 תאים.

החליפו את המיקום של כל זוג מספרים:  
הערך של אינדקס 0 יוחלף עם הערך של אינדקס 1 וכך הלאה.  
הציגו את המערך למסך.

לדוגמה במערך: 1 2 3 4 5 6 7 8

נקבל : 2 1 4 3 5 6 5 8 7