מיקרו-מעבדים ושפת אסמבלר 83-255

תשפ"א/2021 - תרגיל מס' 5

אחראי תרגיל - רומן

מבוא – Input/Output Ports

ההנחיות IN ו- OUT מאפשרות גישה למשאבי הקלט / פלט של המחשב ישירות, ללא שימוש בשגרות BIOS או DOS. הקוד לפניכם הוא דוגמה של שימוש ישיר ביציאות הקלט / פלט על מנת לקבל את המקש שהוקלד במקלדת. פורטים 60h ו- 64h מחווטים למקלדת ומאפשרים לקרוא/לכתוב מידע אליה וממנה.

הפונקציה הבאה בודקת אם יש שינוי במצב המקלדת. אם יש שינוי, ה- LSB של ערך הסטטוס הנקרא מפורט 64h יהיה "1". הביט המדובר נדלק לאחר כל לחיצה או שחרור של מקש. קריאה מפורט 60h מחזירה מהמקלדת את ערך הscan-code של המקש האחרון שנלחץ או שוחרר. שימו לב שscan-code של מקש ששוחרר הוא בעל ערך זהה לקוד של המקש שנלחץ + סיבית MSB דולקת. את טבלאות ה SCAN CODES ניתן למצוא בקישור הבא:

https://docs.microsoft.com/en-us/previous-versions/visualstudio/visual-studio-6.0/aa299374(v=vs.60)

שימו לב: ייתכן שיש הבדלים בטבלאות הקוד בהתאם לסוג המקלדת או מערכת ההפעלה.

PollKeyboard:	IN AL, 64h TEST AL, 01 JZ PollKeyboard	We check using the register AL if the lowest bit in port 64h is equal to 1. If it is one, we must execute a check which key was pressed or released. If it is zero, we loop to check the keyboard status again
	IN AL, 60h	AL holds the scan code of the key
Cont:	...	The rest of the program

<u>הערה חשובה:</u> בכל פעם שהמשתמש לוחץ על מקש במקלדת, מתרחשת (באופן אוטומטי) פסיקת חומרה 09h (IRQ1), והיא מפעילה קוד של מערכת-ההפעלה דוס ש"לוקח" את המקש החדש מן המקלדת ושומר אותו בזיכרון המחשב באיזור ששמור למערכת ההפעלה. הפעולה הזו מתרחשת באופן אוטומטי בפסיקה, ולכן אם לא נבטל את הפסיקה, הקוד לעיל לא יעבוד – כשהוא ייגש למקלדת זה יהיה מאוחר מדי לאחר שכבר הפסיקה לקחה משם את קוד המקש ואיפסה את הסטטוס.
כדי למנוע זאת אנו צריכים למסך(=לבטל) את פסיקות המקלדת(בלבד) על ידי תכנות מחדש של בקר הפסיקות אליו מחוברת המקלדת, שמורה לו להתעלם מIRQ1. הדבר נעשה באמצעות הקוד הבא:

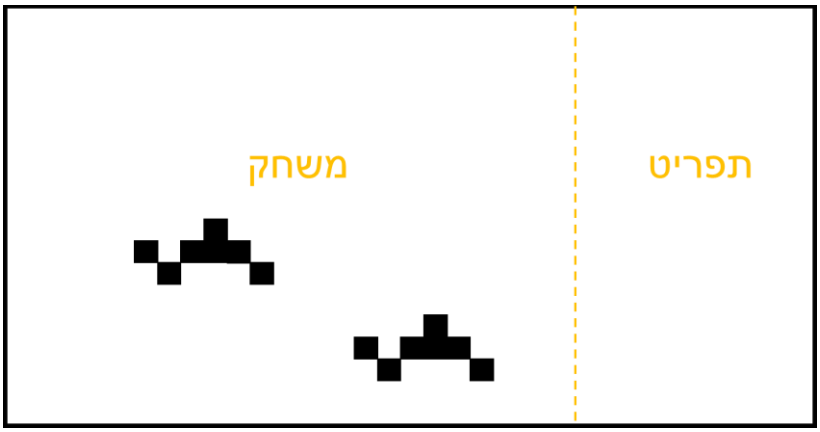
```
in      al, 21h
or      al, 02h
out     21h, al
```

משחק החיים

בתרגיל זה אנו נתחיל לעסוק במשחק החיים. משחק החיים הידוע גם כ"אוטומט תאי". המשחק מתנהל על רשת משבצות, וכל משבצת היא באחד משני המצבים "חי" או "מת". בתחילת המשחק השחקן בוחר אילו משבצות לצבוע, ולאחר מכן המשחק מתנהל בעצמו. המשבצות נקראות "תאים" משום שההתנהגות שלהן מזכירה תאים ביולוגיים שנולדים, חיים ומתים לפי כללים מסוימים, התלויים רק במצבם ובמצב השכנים הקרובים אליהם.

https://en.wikipedia.org/wiki/Conway%27s_Game_of_Life

לוח המשחק שלנו יהיה המסך 25X80 של הדוסבוקס במצב טקסט. מתוך הלוח הזה נקצה את 20 העמודות הימניות לתפריט המשחק, ושאר המסך יציג את המחשק.

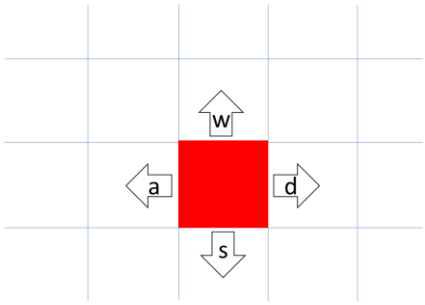


כיוון שבניית משחק החיים בשפת האסמבלי היא משימה לא פשוטה כלל, אנו לא נממש את כולו בבת אחת, אלא נחלק את בנייתו למספר שלבים, שבמהלכם ניישם את הידע שצברנו עד כה.

המשחק מתחיל מזה שהשחקן בוחר אילו משבצות לצבוע. על מנת לבצע שלב זה, אנו צריכים ממשק בו השחקן יכול לבחור את המשבצות ולצבוע אותן או לבטל את הצביעה שלהן. את המשבצות הצבועות נבחר ע"י זה שהשחקן יוכל "לנוע" על מפת המשחק בעזרת הכפתורים 'w', 'a', 's', 'd' (עבור: למעלה, שמאלה, למטה, ימינה, בהתאמה), וילחץ על הלחצן 't' כאשר הוא רוצה לשנות את הצביעה של המשבצת עליה הוא "עומד".

חלק 1 (Input/Output Ports) – שאלה 1

בשלב הראשון נבנה את הממשק להזזת סימבול על גבי המסך וסימון התאים אותם נרצה לצבוע בשחור. הסימבול שמציין את המיקום שלנו במסך הוא ריבוע אדום. בתחילת המשחק הסימבול ממוקם במרכז המסך. לחיצה על אחד



ממקשי 'wasd' גורמת להזזת הסימבול תא אחד בכיוון מעלה עם w נלחץ, מטה אם s נלחץ, שמאלה אם a נלחץ וימינה אם d נלחץ. במידה והשחקן לוחץ על המקש 't' כאשר הסמן נמצא במיקום מסוים, מיקום זה מסומן כ-"חי" בצבע שחור (או, אם כבר היה שחור, הצבע יעלם). נסיים את שלב הבחירה בלחיצה על הלחצן 'e'.

הנחיות ורמזים:

a. לפני תחילת המשחק יש להפוך את שטח לוח המשחק ללבן.
b. לאחר הלחיצה על מקש t כל עוד הסמן נמצא באותו המיקום על המשבצת יש להישאר אדומה, וכאשר הוא עוזב את המיקום, המשבצת צריכה להפוך לשחורה (כלומר, כל עוד הסמן נמצא על משבצת רואים "ריבוע אדום" ולא את הצבע שלה שמסתתר מאחורי הסמן. אתם מוזמנים להוסיף שינוי בסמן שמבדיל בין סמן מעל משבצת חיה או מתה, אבל זה לא חלק מהדרישות)
c. בתרגיל זה אתם **נדרשים** להפעיל את המקלדת ע"י שימוש ביציאות (פורטים) **60h/64h** כדי לקבל קלט מהמקלדת. נדגיש: אין להפעיל את המקלדת ע"י פסיקות.
d. על מנת להדפיס את תווים על המסך כדאי להשתמש בגישה ישירה לזיכרון המסך (כלומר, על-ידי .... mov ax,b800h).
e. כדי ליצור תזוזה של הסימבול על המסך שמרו במשתני עזר את מיקום וערך התו האחרון שהודפס. כאשר מתקבלת לחיצה על אחד מלחצני wasd במקלדת, הדפיסו את הסימבול במקום החדש ואז מחקו את הסימבול החליפו את הערך במיקום הישן בערך השמור.
f. כאשר המשתמש לוחץ על המקלדת יש SCAN CODE אחד עבור לחיצה על המקש, וקוד אחד עבור השחרור של אותו מקש (הסתכלו בדפי המידע של SCAN CODE).
כדי לזהות לחיצות של מקשים <u>בודדים</u>, מומלץ להשתמש בקוד הסריקה שמשמעותו שהלחצן המקלדת <u>שוחרר</u> במקום קוד הסריקה שמשמעותו לחיצה על הלחצן.
g. כאשר הסימבול מגיע לגבול המסך ומתקבלת לחיצה בכיוון זה, על הסימבול להישאר באותו המיקום.
h. לחיצה על הכפתורים שהם לא wasd, t או e לא צריכה להשפיע על התכנית.
i. אין צורך להתחשב באתיות הגדולות, אך ניתן אם אתם רוצים.
j. בחרנו לעבוד עם מקשי המקלדת של האותיות כי הם יותר פשוטים. לחיצה על מקש מקלדת "מיוחד" כגון החצים, או רווח, או ESC, גורם למקלדת לשלוח 2 SCAN CODE שמתארים את המקש המיוחד.

חלק 1, שאלה 2 – בדיקת זמן שלב ההכנה ע"י שימוש ב RTC

שעון בזמן אמת (RTC) הוא התקן חומרה שניתן לגשת אליו דרך יציאות (פורטים) 70h/71h.
כדי לגשת ל- RTC, השתמשו בתהליך הבא בן שני השלבים:

1. כתיבת "פקודה" ליציאה (פורט) 70h.
2. קריאה / כתיבה של נתונים לתוך / החוצה, מהפורט 71h.

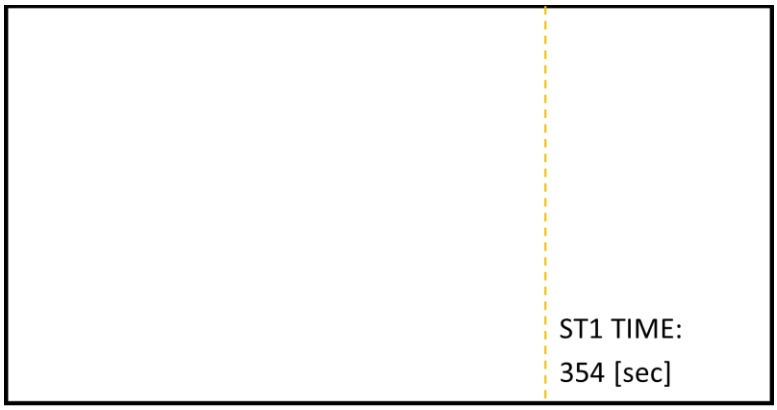
לדוגמה, הפקודה עבור בקשת "שעה" היא 04h. לכן, כדי לקרוא את השעה, נכתוב 04h לפורט 70h ולאחר מכן נקרא מהפורט 71h. הערך שקוראים הוא השדה "שעה" של השעון. צריך לגשת לפורט 71h לאחר כל כתיבה לפורט 70h.

ראו http://stanislavs.org/helppc/cmos_ram.html לתיאור מפורט של פקודות אפשריות.

הוסיפו לתכנית קוד שבודק את השעון בתחילת שלב בחירת המשבצות הדולקות ובסיום בחירת המשבצות (לאחר הלחיצה על מקש 'e').
הדפיסו בתחתית התפריט את מספר הדקות שלקח שלב הבחירה מתחת למחרוזת "ST1 TIME:" כפי שמצויר באיור.

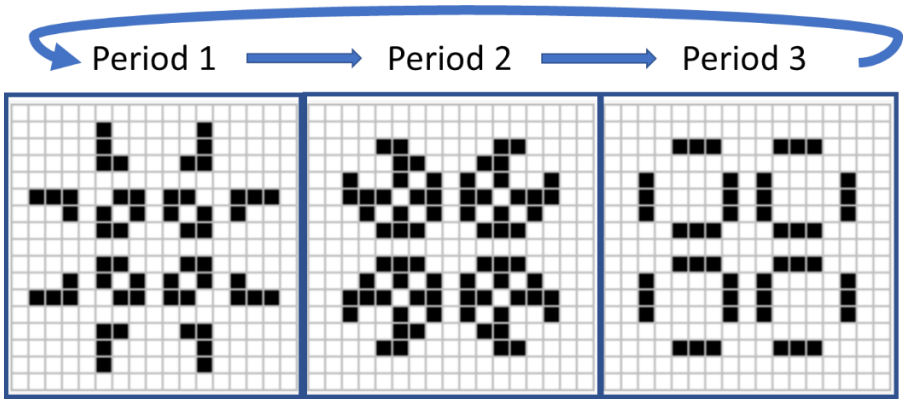
הוראות ורמזים:

- הדפסת המחרוזת על המסך יכולה להיעשות בכל דרך שהיא (היעזרו בנספח לשימוש בפסיקות dos).
- שימו לב שערכי השעה, הדקות והשניות יכולים להשתנות בזמן שלב הבחירה
- הניחו שהתאריך לא משתנה בזמן שלב הבחירה
- ניתן להדפיס את מספר השניות בהקסה-דצימלי או דצימלי לבחירתכם.
- אין צורך להדפיס את הקו המקוטע המופיע באיור (אך ניתן אם אתם רוצים. אפשר להבדיל בצבע שונה את האיזורים אם אתם מעדיפים)



חוקי משחק החיים

המשחק מדמה דורות של תאים שנולדים ומתים. תמונת מצב מסוימת של לוח המשחק נקראת "הדור הנוכחי". לפי התאים שחיים ומתים בדור הנוכחי, קובעים מי יהיו התאים שיחיו או ימותו בדור הבא. למשל, הלוח הבא הוא מחזורי, כאשר יש שלושה דורות קבועים שמחליפים זה את זה.



* שימו לב כי יכול להיות מחזור של יותר מ-3 דורות, או לא להיות מחזור כלל.

כל משבצת במסך המשחק שלנו מוקפת ב-8 משבצות אחרות, לכן לכל תא יכולים להיות עד 8 שכנים (שימו לב לקצוות). המשחק מתנהל לפי הכללים הבאים. לכל תא מסתכלים על השכנים שלו בדור הנוכחי:

- אם הוא **חי**, ויש לו שכן אחד או שאין לו שכנים כלל, הוא מת מבדידות.
- אם הוא **חי** ויש לו יותר משלושה שכנים, הוא מת מצפיפות.
- אם הוא **חי** ויש לו או שניים או שלושה שכנים, מצבו לא משתנה – נשאר חי.
- אם התא **מת**, ויש לו <u>בדיוק</u> שלושה שכנים, הוא הופך להיות חי ("נולד").

בנוסף יש לספק מצב התחלה כלשהו. על פי כללים פשוטים אלה משתנה סידור התאים על הרשת מדור לדור, ומתוך פיזור אקראי של תאים נוצרים מבנים מורכבים ואוכלוסיות מאורגנות.

חלק 2 – פסיקות חומרה, שאלה 3

רכיב ה (Programmable Interval Timer) PIT הינו שעון הניתן לתכנות. לשעון 3 ערוצים. ערוץ 0 של שעון זה מחובר חומרתית למעבד דרך בקר הפסיקות, כך שכל 55 מילי-שניות מתקבלת במעבד פסיקת חומרה מספר 08h.

במעבדה אנו נרצה לבצע קטע קוד שירוץ כל פעם שפסיקה זו מתרחשת. לשם כך אנו נכתוב ISR חדשה עבור פסיקת השעון. מכיוון שיתכן שה-ISR הקודמת עשתה דברים חשובים ואנחנו לא רוצים לפגוע בפעילות התקינה, ב-ISR החדשה אנו **קודם** נקרא לקטע הקוד של הפסיקה המקורית **ולאחר מכן** נבצע את הקטע קוד שלנו.

בסיום ביצוע שגרת פסיקה יש לדווח לבקר הפסיקות שנגמר הטיפול בפסיקה זו על מנת לאפשר טיפול בפסיקות נוספות. דיווח זה נעשה בעזרת הקוד הבא בסיום השגרה ולפני פקודת IRET.

```
mov al,20h ;report the end of the interrupt to the pic
out 20h,al
```

בתרגיל זה אנחנו נממש את משחק החיים לפי החוקים שהוגדרו, על גבי לוח המשחק שלנו, שהוא המסך. נגדיר שהדורות של התאים צריכים להתחלף כל שניה (או כמה שיותר קרוב לפרק זמן זה). כל פעם שמתרחשת פסיקת טיימר אנחנו נבדוק אם הגיע הזמן לעבור לדור חדש. נדע שעברה שניה ע"י זה שנספור את כמות פסיקות הטיימר שהתרחשו. במידה והגיע הזמן לעבור לדור החדש, פסיקת הטיימר צריכה לעדכן את תוכן המסך לדור הבא.

השתמשו בקוד של שאלות 1 ו 2 על מנת להזין תנאי התחלה.

על מנת להצליח במשימה של תרגיל 3 מומלץ עד כדי דרוש לבצע את השלבים הבאים:

a. הגדירו שני משתנים מסוג מילה - Old_int_off, Old_int_seg.
b. בעזרת פסיקת ה-DOS INT 21h/35h שמרו את הכתובת של ה ISR המקורית של פסיקת הטיימר בשני המשתנים שהגדרתם. (חפשו בטבלת פסיקות ה DOS כיצד פסיקה INT 21h/35h עובדת)
c. כתבו שגרת פסיקה חדשה עבור פסיקת השעון (08h) כפי שנלמד בכיתה – הפסיקה החדשה תקרא לשגרת הפסיקה הישנה (לפי הכתובת שנשמרה בסעיף ב'), **ולאחר מכן** תבצע את הפעילות הקשורה לחישוב הדור החדש ועדכון המסך. (כדי לקרוא לקטע קוד החדש יש להשתמש בפקודת CALL DWORD PTR. הזכרו מה מבצעת פקודת IRET וקחו זאת בחשבון)
d. בכל פסיקת שעון יש לקדם מונה פנימי הסופר את כמות הפסיקות הדרושה כדי להגיע לזמן של שניה.
e. במידה ועברה שניה (או טיפה יותר- הרי אנחנו עובדים ברזולוציה של 55 מילישניות) מהפעם האחרונה שהחלפנו דור, הפסיקה צריכה לגרום למסך להתעדכן לתמונה של הדור הבא.
חשבו: איך ניתן לשמור מידע בין קריאה של פסיקה אחת לשנייה? איזה ערך יש לאוגר DS בכניסה לפסיקה? כיצד ואיפה הפסיקה שומרת את המפה הישנה ובונה את החדשה בו זמנית? מה נמצא בזיכרון המסך?
f. בעזרת פסיקת ה-DOS INT 21h/25h הטעינו את כתובת הפסיקה החדשה לתוך טבלת הפסיקות (IVT). (חפשו בטבלת פסיקות ה DOS כיצד פסיקה INT 21h/25h עובדת)
g. לאחר שינוי ה-IVT בצעו <u>לולאה אינסופית</u> (ע"מ שהקוד יישאר בזיכרון) הריצו את התוכנית ב dosbox

הנחיות והערות נוספות:

- זכרו, ה-debugger עשוי לא לעבוד כהלכה כאשר מכבים פסיקות ו/או כאשר עוצרים את פעולת התוכנה בתוך שגרת פסיקה כאשר הפסיקות ממוסכות (הסיבה לכך היא שה-debugger משתמש בפסיקות על-מנת לבצע את פעולתו ולכן אם הפסיקות ממוסכות הדיבאגר עלול לא לעבוד כהלכה)
- על-מנת להקל את ה-debug כתבו את הקוד בצורה מודולרית, חלק חלק, ובדקו כל חלק בנפרד. למשל, קודם כתבו שגרה המקדמת את המונה כל 55 מילישניות, ומדפיסה למסך הודעה כאשר מגיע הזמן לעדכן את המסך. הפעילו את השגרה **בלולאה** וודאו שהיא עובדת באופן תקין. בשלב הבא כתבו קוד ששומר את הפסיקה הישנה ומפעיל אותה בנוסף לקוד השגרה שכתבתם. רק בשלב הסופי הגדירו את השגרה שכתבתם כשגרת פסיקה, ועדכנו את ה-IVT. (מרגע שהקוד רץ בפסיקה קשה יותר לדבג אותו, לכן שלב זה יבוא אחרון)
- מומלץ לשנות את תוכן המסך על ע"י כתיבה לאזור b800:0000 ישירות (זה יהיה מועיל עבור פעולה תקינה יותר בחלק הבא). אחרת, עליכם לשנות את מיקום הסמן (למשל ע"י int 10h / service 02).
- אם ה- DOSBOX נתקע, ניתן לסגור אותו בלחיצת CTRL-F9. אם ה"עכבר" תקוע בחלון, אפשר לעבור לחלון אחר עם ALT-TAB.

- כתבו קוד מסודר ומודולרי, השתמשו בשגרות, השתמשו בקוד מתרגילי בית קודמים (חלקו לשגרות אם לא נכתב כשגרה). הקפידו לתעד בתחילת התוכנית את מטרתה, ולפני כל שגרה תעדו מה היא עושה ואיך מתבצעת העברת המידע מ/אל השגרה.
- שימו לב: פסיקות DOS (int21h) הן לא re-entrant !

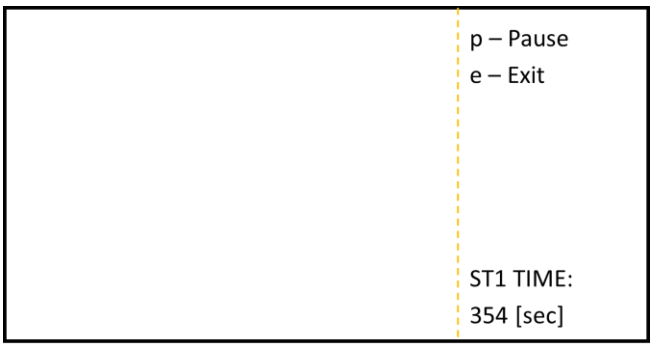
חלק 2, שאלה 4 – מקביליות

כעת נרצה להוסיף למשחק שלנו את היכולת החשובה מאוד, לצאת ממנו! ובנוסף נרצה לאפשר לשחקן לעשות pause למשחק, על מנת שיוכל להתבונן בקפידה בדור הנוכחי.

הוסיפו לאזור התפריט את המחרוזות:

p – Pause
e – Exit

כך שמסך המשחק יראה כך:



כאשר השחקן לוחץ על מקש 'p' התכנית צריכה להפסיק את העדכונים של הדורות, עד לאשר תתקבל לחיצה נוספת על המקש 'p' שמשמעותה un-pause.

כאשר מתקבלת לחיצה על המקש 'e' המשחק צריך להסתיים ולצאת למסך הדוסבוקס הרגיל.

יש לדגום את פורט המקלדת ע"י לולאת polling בקוד התכנית הראשית. אין להשתמש בפסיקה לצורך דגימת המקלדת.

שימו לב כי המשחק ודגימת המקלדת הן פעולות שקורות ***במקביל*** – המשחק פועל מתוך ISR בעוד שדגימת המקלדת נמצאת בתכנית הראשית.

בנוסף שימו לב שביציאה מהתכנית יש להחזיר את ערכי טבלת הIVT לקדמותם. לצורך כך תדאגו לשמור ערכים אלו במקום שנגיש לכם.

בהצלחה!

נספח עזר – פסיקות תוכנה/DOS

נתעסק בפסיקות תוכנה אשר מבוצעות בעזרת int N כאשר N נע בין 0 ל255, סה"כ 256 אפשרויות פעולות המחשב בעת פסיקה:

- דחיפת הכתובת IP ו-CS למחסנית
- דחיפת אוגר הדגלים למחסנית
- קפיצה וביצוע הפסיקה
- משיכה של הכתובת ושל אוגר הדגלים
- חזרה לכתובת אחרי IRET

נעסוק ספציפית בפסיקות DOS שנכתבו ע"י מערכת ההפעלה. ההפעלה של פסיקה זו היא ע"י הפקודה (int 21h) כאשר הפעולה הספציפית מבוצעת בהתאם לרגיסטר AH.

=AH	מטרת הפסיקה	פעולות שצריך לעשות	דוגמא
4cH	סיום התוכנית וחזרה ל-MS dos		mov ah, 4ch int 21h
1 או 8	קליטת תו בודד בתוך AL	נלחץ תו והוא יכניס את הערך הASCII שלו לתוך AL	mov ah, 1h int 21h

2	הצגת תו למסך	נכניס תו לDL והוא יופיע למסך	mov dl, '*'
			mov ah, 2h
			int 21h

9	הצגת מערך string למסך	• נאתחל מערך בסגמנט הDATA • המערך חייב להסתיים ב-'$' • נכניס את כתובת המערך ל-DX אפשר לעשות את זה ע"י אחת משתי האפשרויות הבאות:	mov dx, offset name mov ah, 9h int 21h

למשל, בהינתן הקוד הבא, כך תיראה תמונת המסך:

```
.model small

.stack 100h
.data

   out_msg db 'Hi', 0Ah, 0Dh, 'I', 0Ah, 0Dh, 'Am', 0Ah, 0Dh, 'Stack!', 0Ah,
0DH, '$'
.code
start:
mov ax, @data
mov ds, ax

mov dx, offset out_msg
mov ah, 9
int 21h

mov ah, 4ch
int 21h
end start
```

כאשר 0Ah הוא הערך האסקי של Carriage return (חזרה לתחילת השורה), ו0DH הוא 'Enter'