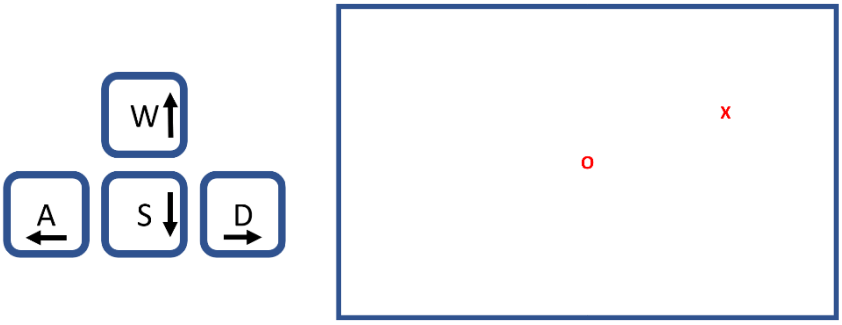
מיקרו מעבדים ושפת אסמבלר תרגיל 4 – תשפ״ב

אחראי תרגיל: אלמוג שוב

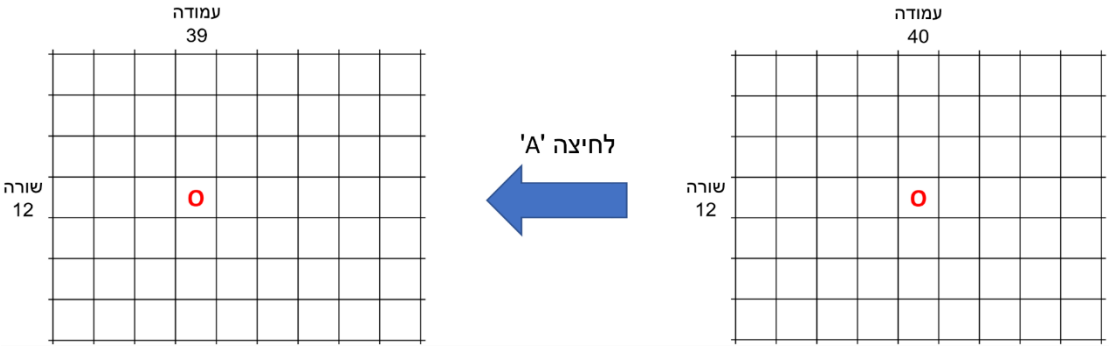
משחק הניווט

בתרגיל זה נבנה בשלבים משחק ניווט. מטרת המשחק היא לאסוף עם השחקן (הסימבול 'O') את הנקודות על המסך (הסימבול 'X'). בתחילת המשחק הסימבול ממוקם במרכז המסך, וזז באמצעות מקשי הניווט WASD. בכל פעם שהשחקן אוסף נקודה מופיעה נקודה חדשה על המסך במיקום אקראי. המשחק מסתיים כאשר המשתמש לוחץ על המקש Q, ואז תודפס הודעה עם הניקוד שצבר השחקן.



חלק 1 – הזזת השחקן

בשלב הראשון נבנה את הממשק להזזת הסימבול על גבי המסך. בתחילת המשחק הסימבול ממוקם במרכז המסך. לחיצה על אחד ממקשי הניווט גורמת להזזת הסימבול תא אחד בכיוון המתאים. למשל אם השחקן נמצא בשורה 12 עמודה 40, לאחר לחיצה על מקש A הסימבול יזוז למיקום השכן משמאל – שורה 12 עמודה 39:



הוראות ורמזים:

- תחילה קראו את נספח 1. אתם **נדרשים** להשתמש ביציאות (פורטים) **60h/64h** כדי לקבל קלט מהמקלדת.
- עליכם לנטר את מקשי הניווט WASD, וכן את המקש Q שלחיצה עליו תסיים את המשחק.
- בתחילת המשחק, צבעו את המסך בשחור (למשל, הדפיסו את התו "רווח" עם רקע שחור), והדפיסו את סימבול השחקן ('O' אדום) במרכז המסך.
- על מנת להדפיס את הסימבול השתמשו בגישה ישירה לזיכרון המסך (mov ax,b800h ...).
- הסימבול יודפס בצבע אדום (ללא רקע, כלומר הרקע צריך להישאר כפי שהיה לפני הגעת הסימבול).
- כדי ליצור תזוזה של הסימבול על המסך שמרו במשתנה את מיקום הסימבול. כאשר מתבצעת תזוזה, הדפיסו את הסימבול במקום החדש ואז מחקו אותו מהמיקום הישן (ע"י הדפסת רווח), ועדכנו את משתנה המיקום.
- כדי לזהות לחיצות בודדות של מקשים (כלומר לחיצה ארוכה = לחיצה בודדת), השתמשו בקוד הסריקה שמשמעותו שלחצן המקלדת שוחרר במקום קוד הסריקה שמשמעותו שהלחצן נלחץ.
- כאשר הסימבול מגיע לגבול (מסגרת) המסך ומתקבלת לחיצה בכיוון זה, על הסימבול להישאר באותו המיקום.

חלק 2 – איסוף נקודות

בשלב השני, נוסיף למשחק את איסוף הנקודות. על מנת ליצור אקראיות במיקום הנקודות, נשתמש בשעון.

שעון בזמן אמת (RTC) הוא התקן חומרה שניתן לגשת אליה דרך יציאות (פורטים) 70h/71h. השימוש בשעון מתבצע ע"י כתיבת 'פקודה' ליציאה (פורט) 70h, ולאחר מכן קריאה/כתיבה של נתונים דרך פורט 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 לתוכנית שגרה שיוצרת נקודה חדשה במיקום אקראי במסך, קראו לשגרה זו בתחילת המשחק על מנת ליצור את הנקודה הראשונה, וכן לאחר כל איסוף נקודה קראו לה על מנת ליצור נקודה חדשה.

כדי לייצר מיקום אקראי על המסך, קראו מהשעון את ערך השניות ואת ערך הדקות. כל אחד מערכים אלו מתקבל בגודל בית, צרפו את הערכים למילה אחת כך שערך השניות הוא הבית הגבוה (ערך השניות הוא 'אקראי' יותר מבחינתנו, מכיוון שמשתנה יותר מהר בין נקודות. לכן נרצה שישפיע יותר על המיקום, ונציב אותו בבית הגבוה). כעת השתמשו בערך האקראי שקיבלתם כהיסט בזיכרון המסך (יכול להיות שידרשו שינויים מסוימים בערך זה על מנת לשמש כהיסט).

בסיום התוכנית, עליכם להדפיס (לא משנה באיזה מיקום) הודעה כמה נקודות צבר השחקן, לדוגמה: "Score: 05".

הוראות ורמזים:

- שימו לב שההיסט האקראי שקיבלתם אינו חורג מטווח זיכרון המסך.
- הנקודה החדשה צריכה להיות במיקום שונה מהנקודה הקודמת.
- על מנת להדפיס את המחרוזת למסך, היעזרו בנספח 2.
- הדפיסו את כמות הנקודות כערך עשרוני, הניחו כי הוא לכל היותר דו ספרתי.
- המלצה: שמרו ב DS את הודעת הסיום עם מקומות ריקים, כך שתוכלו להכניס בהם מאוחר יותר את כמות הנקודות. למשל בדוגמה נשמור: "Score: _ _".

חלק 3 – לפי הקצב

בשלב השלישי, נגרום לשחקן לזוז בקצב קבוע בכיוון הנוכחי (בדומה למשחק סנייק). לשם כך נערוך את פסיקת השעון. כזכור, למעבד מחובר רכיב חיצוני בשם PIT (Programmable Interval Timer) שיוצר פסיקת חומרה (פסיקת שעון, מס' 08h) כל 55 מילי שניות.

קראו על תפקוד הפסיקה הקיימת: http://vitaly_filatov.tripod.com/ng/asm/asm_001.7.html

לאור האמור בהסבר לעיל, נרצה לעדכן את פסיקה **1Ch** כך שבכל 0.165 שניות (כלומר, כל פעם שלישית שהפסיקה 1Ch נקראת = הזמן של 3 פסיקות שעון) השחקן מתקדם צעד אחד בכיוון הנוכחי (כיוון המקש האחרון שהוקש). עליכם לכתוב קוד שגרה עבור פסיקה 1Ch, ולעדכן את טבלת ה IVT בהתאם.

הוראות ורמזים:

- זכרו למסך פסיקות בעת שינוי טבלת ה IVT, כפי שנלמד בכיתה.
- זכרו בסיום התוכנית לשחזר את טבלת ה IVT למצבה המקורי.
- המלצה: בשגרה החדשה עדיף לבצע כמה שפחות פעולות (משום שהיא מתרחשת כל 55ms). לכן עדיף שהשגרה החדשה לא תקרא בעצמה לפונקציה שמבצעת הזזה, אלא למשל תשנה משתנה בזיכרון שלאחר מכן בקוד ה'ראשי' יגרום להפעלת פונקציית ההזזה.
- הדיבגר משתמש בפסיקות כדי לבצע את פעולתו, ולכן עשוי לא לעבוד כהלכה כאשר מכבים פסיקות ו/או עוצרים את פעולת התוכנה בתוך שגרת פסיקה כאשר הפסיקות ממוסכות.
- **שימו לב: פסיקות DOS (int 21h) הן לא re-entrant !**

הערות כלליות

- על הקוד להיות מתועד היטב!
- לפני כל שגרה, הוסיפו הערה מה מבצעת השגרה, מה הקלט ומה הפלט. שמרו על פורמט קבוע.
- כתבו קוד מודולרי. כלומר חלקו את הקוד לשגרות בעלי תפקיד משמעותי. בסופו של דבר צריכה להיות היכולת לקרוא את הקוד הראשי או השגרה הראשית בצורה קצרה שנותנת הבנה על מהלך המשחק.
- בצעו debug בצורה מודולרית על מנת לעקוב אחרי בעיות בקוד בצורה מיטבית.

הגשה

- הגשה במודל עד תאריך 9.6.22. ההגשה בזוגות, אחד מכל זוג מגיש.
- נא להוסיף בתחילת כל קובץ הערה עם שמות המגישים ותעודות זהות.
- הגישו תיקיית zip בשם Name1_ID1_Name2_ID2.zip הכוללת את הקובץ ex4.asm.
- במידה ולא סיימתם את התרגיל, ניתן להגיש רק את החלק הראשון בקובץ בשם ex4p1.asm, או רק את 2 החלקים הראשונים בקובץ בשם ex4p2.asm.

נספח 1: אמצעי קלט פלט

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 ממנה / אליה.

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 ה-scan 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 בקישורים הבאים:

https://docs.microsoft.com/en-us/previous-versions/visualstudio/visual-studio-6.0/aa299374(v=vs.60)
http://staff.ustc.edu.cn/~xyfeng/research/cos/resources/BIOS/Resources/assembly/makecodes.html
שימו לב: ייתכן שיש הבדלים בטבלאות הקוד בהתאם לסוג המקלדת או מערכת ההפעלה.

PollKeyboard:	IN TEST	AL, 64h AL, 01	We check using the register AL if the lowest bit in port 64h is equal to 1.
	JZ	PollKeyboard	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.

הערה חשובה: בכל פעם שנלחץ מקש במקלדת, מתרחשת (באופן אוטומטי) פסיקת חומרה 09h (IRQ1) שמפעילה קוד של מערכת ההפעלה דוס ש"לוקח" את המקש החדש מן המקלדת ושומר אותו בזיכרון המחשב באזור ששמור למערכת ההפעלה. פעולה זו מתרחשת באופן אוטומטי, לכן אם לא נבטל את הפסיקה, הקוד לעיל לא יעבוד – הוא ייגש למקלדת מאוחר מדי לאחר שהפסיקה כבר 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. הדבר נעשה באמצעות הקוד הבא:

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

נספח 2: פסיקות תוכנה/DOS

פסיקות DOS הן פסיקות תוכנה שנכתבו ע"י חברת Microsoft כחלק מהקוד של 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 לMS-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 (חייב להסתיים ב-'$'), נכניס את כתובת המערך ל-DX.	mov dx, offset name mov ah, 9h int 21h

ניתן לקרוא בהרחבה על פסיקות DOS (ואחרות) בקישור הבא:
http://www.gabrielececchetti.it/Teaching/CalcolatoriElettronici/Docs/i8086_and_DOS_interrupts.pdf

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' (ירידת שורה).