# פסיקות - גרפיות

עבודת מחקר

בדף העבודה נתנסה בעבודה במצב גראפי.

נצייר ביחד ריבוע ונשתמש בסמן.

1. פתחו קובץ חדש ואל תשכחו להוסיף בראשו **p186**

1. **מעבר למצב גרפי**:

המעבד יכול לעבוד בשני מצבים הראשון מצב טקסט (בו עבדנו עד היום) והשני מצב גרפי בו משתמשים בגרפיקה (ציור ושרטוט על המסך) כדי לעבור למוד גראפי נשתמש בפסיקת ה bios 10h

mov ax, 13h

int 10h

1. **wait for character**

הוסיפו בסוף התוכנית wait for character , כך שהתוכנית שלכם תעצור ותוכלו לראות מה קוrה על המסך לפני היציאה

mov ah, 0h

int 16h

1. **חזרה ממצב גרפי**:

mov ax, 2

int 10h

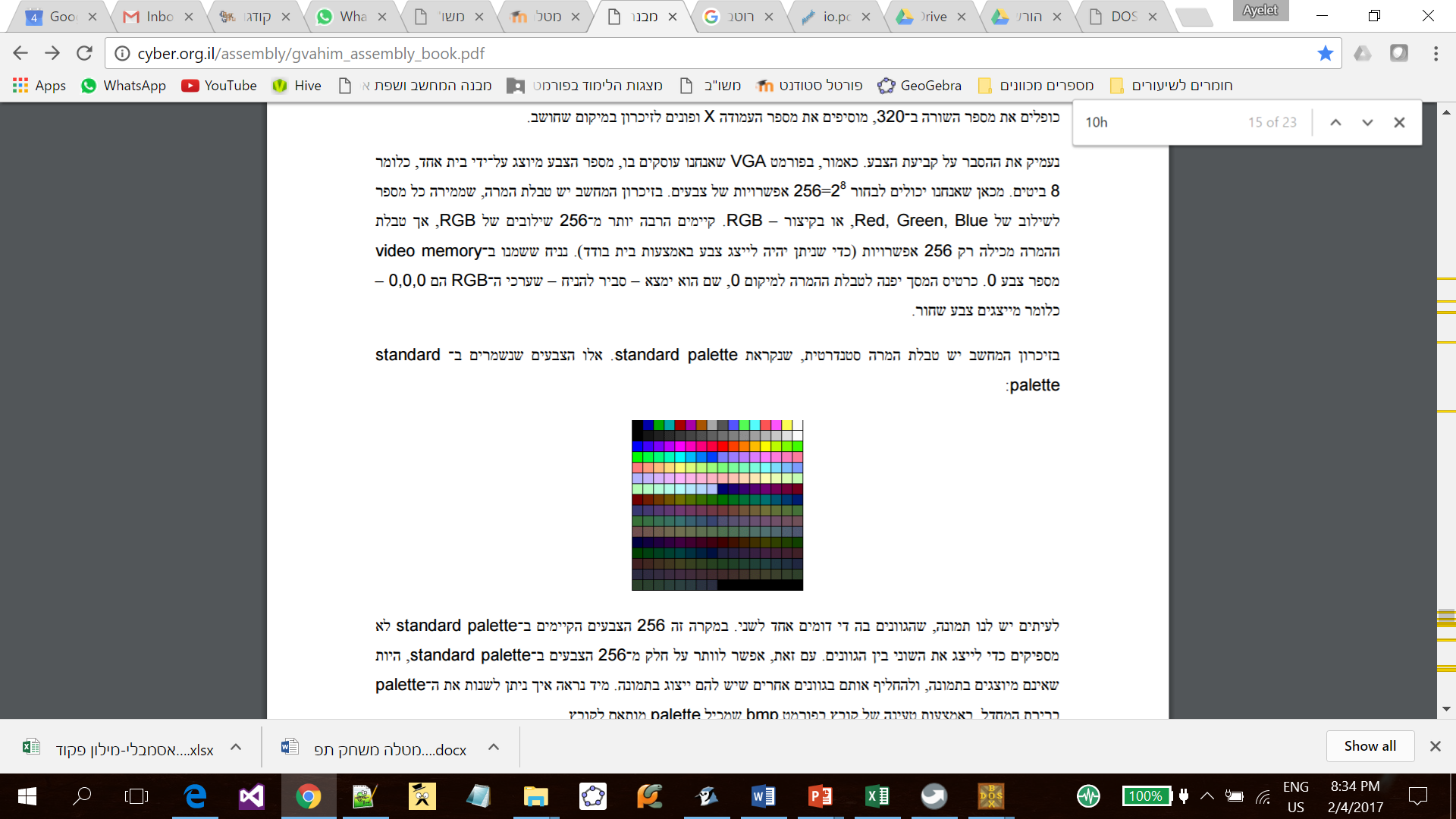
הוסיפו בסוף התוכנית שלכם את ה interrupt החוזר למצב text.

1. **ציור במצב גרפי**:

המסך מורכב מנקודות. כל נקודה נקראת פיקסל. במצב גרפי המסך מורכב מ 200 שורות ו 320 עמודות של פיקסלים. לכל פיקסל מיקום וצבע.

המיקום מורכב משורה ועמודה ⇦ column, row

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 319,0 |  |  |  |  |  |  | 0, 0 |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
| 319, 199 |  |  |  |  |  |  | 0,199 |

**הצבע:**

בזיכרון המחשב יש טבלה בודל 256 ובכל תא שלה מצוי מספר המייצג צבע.

(טווח הצבים הוא מ- 0 שחור עד - 255 לבן)

כדי לצייר פיקסל ניתן מיקום וצבע.

למשל: שורה 0

עמודה 0

צבע 4

יצבע בצבע אדום את הפיקסל בפינה השמאלית העליונה

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 319,0 |  |  |  |  |  |  | 0, 0 |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
| 319, 199 |  |  |  |  |  |  | 0,199 |

האינטרפט שמצייר פיקסל הוא פסיקת bios - **int 10h**

הפרמטרים עבור אינטרפט זה:

ah = 0ch – קוד הפסיקה

bh = 0 - תמיד

cx – עמודה

dx – שורה

ax – צבע

כתבו תכנית המציירת פיקסל אדום )תא 4 בפלטת הצבעים) בשורה 10 עמודה 100.

**הדרכה:**

1. הגדירו 3 משתנים:

x\_coordinate dw 100

y\_coordinate dw 10

color dw 4

1. הוסיפו (לאחר הכניסה למוד גרפי

xor bh, bh ; bh = 0

mov cx, [x\_coordinate]

mov dx, [y\_coordinate]

mov ax, [color]

mov ah, 0ch

int 10h

1. העבירו את קטע הקוד לפרוצדורה העושה שימוש במשתנים לקואורדינטות x ו y של פיקסל ואת צבעו ומציירת אותו על המסך.

מאחר ובפרוצדורה נעשה שימוש ברגיסטרים שונים עלינו לשמור את מצב הרגיסטרים בכניסה לפרוצדורה ובסימומה למשוך אותך חזרה.

; draws one pixel

פקודת ה – pusha משמרת את מצב הרגיסטרים לפני ביצוע הפרוצדורה.

proc draw\_pixel

pusha

…………. פקודות לביצוע

popa

פקודת popa מחזירה את הרגיסטרים למצב שהיו בכניסה לפרוצדורה.

.

ret

endp draw\_pixel

קו הוא אוסף של נקודות **...............**

1. כתבו פרוצדורה המציירת קו על המסך (לרוחב). הפרוצדורה עושה שימוש בקואורדינטת תחילת השורה אורך הקו, ובצבע הפיקסל ומציירת את השורה.

**הדרכה:**

* לצורך ביצוע השרטוט הקו עלינו ליצור משתנה חדש שבו נשמור את אורך הקו,

**len dw 10**

* ומשתנה חדש בו נשמור את הנקודה בציר ה - x של ציור הפיקסל כאשר היא גדלה ב – 1 בכל פעם שניצור נקודה חדשה על הרצף של ציור הקו [x\_temp].
* נשנה את המשתנה האחראי על קואורדינטת x בפרוצדורה המציירת את הנקודה למשתנה החדש [x\_temp].

ניצור פרוצדורה חדשה המציירת קו למסך - draw\_line

* בתחילת הפרוצדורה נבצע פקודת pusha נשמור לרגיסטר cx (מונה הלולאה) את הערך של המשתנה השומר את אורך הקו.
* נעביר למשתנה החדש את ערכה של נקודה x של ציור הפיקסל. [x\_coordinate] ⇦ [x\_temp]
* ניצור לולאה שערכה הוא כאורך הקו בפיקסלים.
* ונזמן את הפעולה של ציור הפיקסל מתוך פרוצדורה זו.
* נגדיל את ערכו של [x\_temp]
* בסיום הפרוצדורה נבצע פעולת popa.

; draws a line of pixels

proc draw\_line

pusha

*; move x\_coordinate to x\_temp*

mov ax, [x\_coordinate]

mov [x\_temp], ax

mov cx, [len]

draw:

call draw\_pixel

inc [x\_temp]

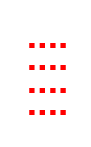
loop draw

popa

ret

endp draw\_line

* **זמנו את הפרוצדורה draw\_line במקום draw\_pixel**

ריבוע הוא אוסף של קווים 

1. השתמשו בפרוצדורה שכתבנו בשאלות 5 ו 4 לציור ריבוע 10x10 שהפינה השמאלית העליונה שלו בשורה 100 ועמודה 150.

**הדרכה:**

ריבוע הוא אוסף של קווים, ניצור כל פעם קו חדש כאשר הערך של תחילת הריבוע על ציר ה – y גדל ב – 1.

* נוסיף משתנה חדש בו נשמור את הנקודה בציר ה - y של ציור הקו כאשר ערכה גדל ב – 1 בכל פעם [y\_temp].
* נשנה את המשתנה המציין את קואורדינטת ה y בפרוצדורה לציור פיקסל למשתנה החדש [y\_temp].
* נעביר למשתנה החדש את ערכה של נקודה y של ציור הפיקסל. [y\_coordinate] ⇦ [y\_temp]

|  |
| --- |
| proc draw\_rect  pusha  mov cx, [len]  mov ax, [y\_coordinate]  mov [y\_temp], ax  rect:  call draw\_line  inc [y\_temp ;column  loop rect  popa  ret  endp draw\_rect |

* נעביר לרגיסטר cx את הגודל הריבוע [len].
* ניצור לולאה לציור הריבוע:
  + נצייר את קו – נזמן את הפרוצדורה המציירת את הקו.
  + נגדיל את הערך של [y\_temp]

העבירו את התכנית של ייצור לריבוע לפרוצדורה.

שמרו תכנית זו היא תסייע לכם בהמשך ביצירת הפרויקט שלכם אם תחבור לעבוד עם פיקסלים.

תזוזה של הריבוע היא מחיקת הריבוע הקודם וציורו במקום חדש. מחיקת הריבוע מתבצעת ע"י ציור של ריבוע שחור על גבי הריבוע האדום.

1. ניצור תכנית חשה שכאשר נקיש על מקש כל שהוא במקלדת הריבוע שלנו יזוז ב 5 פיקסלים.   
   כדי ליצור אפקט של תזוזת הריבוע יש לצייר ריבוע שחור במקום בו היה הריבוע הקודם ולצייר את הריבוע החדש כאשר הוספנו לקואורדינאטה של ציר ה - x(מיקום תחילת ציור הריבוע בשורה) 5 פיקסלים.

* נאתחל את ציור הריבוע על ציר ה - x ל – 10 x\_coordinate dw 10
* בתכנית הראשית נוסף את קטעי הקוד הבאים:

**move:**

הפסיקה הבודקת אם הקשנו על מקש היא:

mov ah, 0h

int 16h

לאחר הקשה על מקש נצייר ריבוע שחור על המיקום של הריבוע הצבעוני (צבע = 0).

נצייר ריבוע חדש כאשר הזזנו את נקודת תחילת הציור של הריבוע [x\_coordinate] בשורה, ב – 5 פיקסלים.

כל קטע הקוד הזה בתוך לולאה כדי שכל הקשה על מקש תגרום לתזוזה חדשה של הריבוע.

mov [color], 70

call draw\_rect

; wait for char

mov ah, 0h

int 16h

; draw black rectangle

mov [color], 0

call draw\_rect

;mov right 5 pixels

add [x\_coordinate], 5

**jmp move**

שפרו את התכנית והוסיפו תנאי שאם הקישו על מקש Esc נצא מהתכנית.

הוסיפו לפני החזרה ל text mode תגית, end\_game , קיפצו אליה כשהמשתמש לוחץ Esc

; wait for char

mov ah, 0h

int 16h

cmp al, 27

je end\_game

1. **העכבר** –

**פתחו קובץ חדש ואל תשכחו להוסיף p186 בתחילתו**

כדי לעבוד עם העכבר במצב גראפי נשתמש בפסיקה **int 33h**.

לפסיקה זאת מספר שימושים:

|  |  |  |
| --- | --- | --- |
| פקודה | פרמטר | אסמבלי |
| איתחול העבודה עם הסמן | ax = 0h | xor ax, ax  int 33h |
| הופעת הסמן | ax = 1h | mov ax, 1h int 33h |
| הסתרת הסמן | ax = 2h | mov ax, 2h int 33h |
| קריאת מיקום הסמן ומצב העכבר | ax = 3h | mov ax, 3h int 33h |

**נשתמש ב 3 בפסיקות הראשונות.**

כתבו תכנית:

המאתחלת את הסמן

מציגה אותו

ומסתירה אותו.

על מנת שהתוכנית תעצור ותוכל לראות את הסמן הגרפי הוסף את 2 השורות הבאות:

mov ah, 1h ; wait for character

int 21h

**אחרי הצגת הסמן ואחרי הסתרתו**

**תזכורת ☺:** זהו ה - interrupt הקורא תו לתוך הרגיסטר al. באופן הזה התוכנית תעצור ותחכה שתקישו תו כלשהו.

זכרו להיכנס למצב גרפי בתחילת התכנית ולחזור למצב טקסט בסופה.

|  |
| --- |
| ; enter graphic mode  mov ax, 13h  int 10h  ; initiate cursor  xor ax, ax  int 33h    ;show cursor  mov ax, 1h  int 33h  ; wait for character  mov ah, 1  int 21h    ; hide cursor  mov ax, 2h  int 33h  ; wait for character  mov ah, 1  int 21h    ; back to text mode  mov ax, 2  int 10h |

כלומר סדר הפעולות:

* כניסה למצב גרפי
* אתחול הסמן
* הצגת הסמן
* עצירה עד להקשת תו
* הסתרת הסמן
* עצירה עד להקשת תו
* יציאה ממצב גרפי

1. השלב הבא הוא לקלוט את מיקום העכבר ואת סטטוס הלחיצה עליו. נשתמש בפסיקה:

mov ax,3h

int 33h

הפסיקה מחזירה את הערכים הבאים:

**bx** - מצב הלחיצה על כפתורי העכבר

bx **=** 0 – אין כפתור לחוץ

bx = 1 – כפתור **שמאלי לחוץ**

bx = 2 – כפתור ימני לחוץ

bx = 3 - 2 הכפתורים לחוצים

**cx** - מיקום העכבר, עמודה בין 0 - 639 (קואורדינטת x)

**dx** - מיקום העכבר, שורה בין 0-199 (קואורדינטת y)

הלולאה הבאה "מחכה" ללחיצה על הכפתור השמאלי של העכבר.

MouseLoop:

mov ax,3h ; read mouse status and position

int 33h

**cmp bx, 01h ; check left mouse click**

jne MouseLoop ; if left click not pressed….

1. שלבו לולאה זו בתכונית שכתבתכם:

* הציגו את העכבר למסך,
* בדקו בלולאה עם נלחץ הכפתור השמאלי שלו – החליפו את הפסיקה שמחכה לתו אחרי הצגת העכבר בלולאה.
* אם כן הסתירו את העכבר

1. מזגו את 2 התוכניות שכתבתם (ציור ריבוע וקליטת מיקום העכבר) לתוכנית אחת:

שלבי ביצוע התכנית:

* 1. עוברת למצב גרפי
  2. מאתחלת את הסמן
  3. מציגה את הסמן
  4. מחכה ללחיצה על הכפתור השמאלי של העכבר
  5. קוראת את מיקומו לתוך משתנים
  6. מסתירה את הסמן
  7. מציירת ריבוע שפינתו השמאלית העליונה במיקום שמצאת

|  |
| --- |
| **שימו לב ♥, כשאנחנו עובדים במצב גרפי כמות השורות שיש לנו היא 320 בלבד, לכן צריך לבצע התאמה ולחלק את cx בשתיים כדי להגיע למיקום הנכון** |

**הדרכה:**

* העבירו את המשתנים והפרוצדורות מהקובץ הקודם
* לאחר הלולאה:

- הכניסו ל x\_coordinate את cx וחלקו ב 2 (השתמשו ב shr ).

- הכניסו ל y\_coordinate את dx

* לאחר הסתרת העכבר זמנו את draw\_rect

; enter graphic mode

mov ax, 13h

int 10h

; initiate cursor

xor ax, ax

int 33h

;show cursor

mov ax, 1h

int 33h

MouseLoop:

mov ax,3h ; read mouse status and position

int 33h

cmp bx, 01h ; check left mouse click

jne MouseLoop ; if left click not pressed….

; read cursor location

**mov [x\_coordinate], cx**

**shr [x\_coordinate], 1**

**mov [y\_coordinate], dx**

; hide cursor

mov ax, 2h

int 33h

**call draw\_rect**

; wait for character

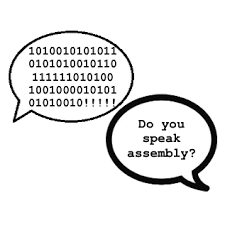
mov ah, 1

int 21h

; back to text mode

mov ax, 2

int 10h





סוף