גרפיקה ושימוש בעכבר - צייר

בדף עבודה זה נתנסה בעבודה במצב גראפי. נצייר ביחד מלבנים ונזוז על המסך עם העכבר.

**בתודה לרחל פרלמן**

# 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

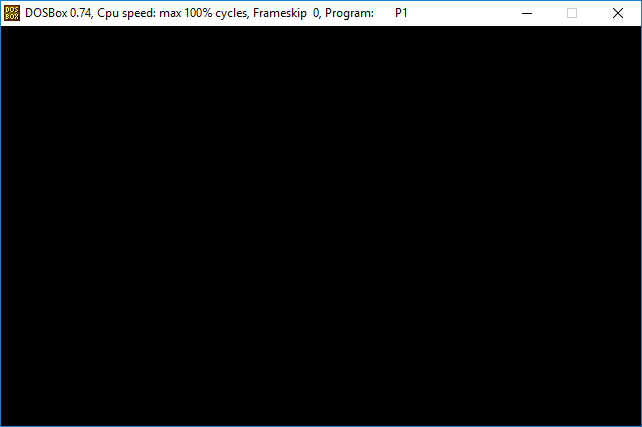
; Enter Graphic mod 320 \* 200 pixels, 256 colors

mov ax, 13h

int 10h

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 ⇦ row, column



נוסיף פקודה לתכנית שבהקשת מקש (כל שהוא) נצא מהמצב הגראפי ונחזור למוד של טקטס.

; Wait for key press

mov ah,00h

int 16h

0,0

319,199

0,199

319,0

נוסיף בסוף התכנית פסיקה המחזירה את המסך למצב טקסט 80\*20))

;Retern to text mod

mov ax, 2

int 10h

start:

IDEAL

MODEL small

STACK 100h

**p186**

כדי לשמור על מצב הרגיסטרים בכניסה לכל פרוצדורה יש צורך לשמור את מצב הרגיסטרים בתחילתה push למחסנית, ולמשוך את ממצב הרגיסטרים בסופה pop מהמחסנית.

ההוראה p186 מאפשרת לעשות פוקדת push ו – pop לכל הרגיסטרים בהוראה אחת.

**pusha**

**popa**

mov ax, @data

mov ds, ax

; --------------------------

; Enter Graphic mod 320 \* 200 pixels, 256 colors

mov ax, 13h

int 10h

; Wait for key press

mov ah,00h ; get keystroke from keyboard

int 16h

; Retern text mod

mov ax, 2

int 10h

; --------------------------

exit:

mov ax, 4c00h

int 21h

END start

# ציור במצב גרפי – ציור פיקסל (נקודה על המסך)

**הצבע:**

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

כדי לצייר פיקסל למסך, צריך לציין שלושה נתונים: מיקום על ציר x, מיקום על ציר y וצבע.

לדוגמה: שורה 0

עמודה 0

צבע 4

יצבע בצבע אדום את הפיקסל בפינה השמאלית העליונה

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

האינטרפט (פסיקה) המצייר פיקסל למסך, הוא פסיקת bois - **int 10h**

DATASEG

x\_coordinate dw 5 ; place in line

y\_coordinate dw 5 ; place in column

color dw 2

CODESEG

proc drawPixel

pusha

xor bh, bh

mov cx, [x\_coordinate] ; place in line

mov dx, [y\_coordinate] ; place in column

mov ax, [color]

mov ah, 0ch

int 10h

popa

ret

endp drawPixel

start:

**צרו פרוצדורה המציירת למסך פיקסל.**

כדי לצייר פיקסל על המסך יש לספק לאסמבלי את הנתונים הבאים:

מיקום על השורה ציר x,

מיקום בטור, ציר y

וצבע.

**הפרמטרים עבור אינטרפט זה:**

ah = 0ch – קוד הפסיקה

bh = 0 – איפוס רגיסטר bh

cx – עמודה

dx – שורה

ax – צבע

פיסקה המציירת פיקסל למסך:

xor bh, bh

mov cx, [x\_coordinate]

mov dx, [y\_coordinate]

mov ax, [color]

mov ah, 0ch

int 10h

**זמנו את הפרוצדורה מהתכנית הרשאית**

call drawPixel

**הריצו ובדקו.  
שמרו את הקובץ, לשם חדש והמשיכו בקובץ החדש.**

# ציור מרובע

קו הוא אוסף של נקודות ............... על ציר ה – x

וציור של מרובע הוא אוסף של קווים אופקיים.

נצייר את הריבוע כאוסף של קווים בעזרת שתי לולאות:

* לולאה ראשונה לציור קו ולולאה שנייה לציור קווים
* נשנה את הערכים של המיקום על ציר x לערך לא יודע ?
* ואת הערכים של המיקום על ציר y לערך לא יודע?
* נשמור את המיקום של **תחילת ציור** המרובע שלנו על ציר ה – x ועל ציר ה – y.
* ונוסיף שני מונים, מונה ללולאה המונה את הנקודות (המרכיבות קו),
* ומונה ללולאה המונה את הקווים (היוצרים את המרובע).

DATASEG

x\_coordinate dw ? ; place in line

y\_coordinate dw ? ; place in column

color dw 2

x\_begin dw 0 ; Starting point on line

y\_begin dw 30 ; Starting point on column

x\_count db 35 ; loop count line draw

Y\_count db 25 ; loop count column draw

ניצור פרוצדורה לציור מרובע

CODESEG

Proc DrawRectangle

; Draw square at x\_begin, y\_begin position, size 25\*35

mov [x\_count],35 ; Square width

mov [Y\_count],25 ; Square height

mov ax,[x\_begin] ; Save begin point on X

mov [x\_coordinate],ax

mov ax,[y\_begin] ; Save begin point on Y

mov [y\_coordinate],ax

line\_loop:

; Print one dot on screen

call drawPixel

inc [x\_coordinate]

dec [x\_count]

cmp [x\_count], 0

jnz line\_loop

mov ax,[x\_begin] ; Reset line counters

mov [x\_coordinate], ax

mov [x\_count],35 ; Reset column counters

inc [y\_coordinate]

dec [y\_count]

cmp [y\_count], 0

jnz line\_loop

ret

endp DrawRectangle

קראו לפרוצדורה מהתכנית הראשית

מקמו אותה מתחת לפסיקה המעבירה את המסך למצב גרפי ולפני הפסיקה הממתינה לתו.

call DrawRectangle

הריצו את התכנית ובדקו את התוצאה.

**זכרו 🖰,** כדי לצאת מהתכנית ולחזור למצב טקסט עליכם להקיש על מקש במקלדת.

# ציור פלטת הצבעים של הצייר למסך

בתכנת צייר בתפריט מופיעים כמה ריבועים בצבעים שונים, המהווים את פלטת הצבעים מהם בוחר המשתמש את הצבע לציור על המסך.

נוסיף לתכנית שלנו מרובעים בצבעים שונים.

**נשנה** את המשתנה של הצבעים - **[color]** למערך בו נשמור את הערכים של הצבעים שנציג למסך ונוסיף לו ערכים.

color dw 2, 3, 4, 14, 15

ציור המרובעים שלנו מתחיל תמיד באותה עמודה על ציר ה – x (נקודה 0)

אך מיקום השורה (ציר y) משתנה ממרובע למרובע, המרובעים מצוירים למסך זה מתחת לזה.

**נשנה** את המשתנה הקובע את תחילת ציור המרובע על ציר ה – y **[y\_begin]** למערך ונוסיף לו ערכים.

גובה על מרובע שלנו הוא 25 פקסלים לכן ההפרשים בין הערכים הם 25.

y\_begin dw 25, 50, 75, 100, 125

**רגיסטר si** הוא רגיסטר שניתן להוסיף לשם מערך כדי להצביע על התא במערך אליו נרצה לפנות.

בפעולה של ציור פיקסל נשנה את המשתנה של הצבע **[color]** לשם המערך + רגיסטר si

פעולה זו תאפשר לנו לפנות בכל פעם לפנות לתא אחר במערך.

נשתנה **בפעולה של ציור פיקסל** את המשתנה של הצבע לשם המערך + רגיסטר si

**mov ax, [color+si]**

נשנה **בפעולה של ציור מרובע** את המשתנה האחראי על מיקום תחילת מרובע הצבעים שלנו על ציר ה – y (השורה בה מתחילים לצייר את המרובע)

**mov ax,[y\_begin + si]**

## לולאה המציירת את פלטת הצבעים על המסך

בתוכנית הראשית ניצור לולאה המציירת למסך את פלטת הצבעים (המרובעים הצבעונים בצדו הימני של המסך, תפריט הצבעים)

בצייר שלנו 5 צבעים ולכן מונה הלולאה הוא 5.

נאפס את רגיסטר si.

; Drawing color squares

mov cx, 5

mov si, 0

colorLoop:

call DrawRectangle

add si, 2

loop colorLoop

הריצו את התכנית ובדקו את התוצאה.

שמרו את הקובץ שלם חדש, המשיכו לעבוד על הקובץ החדש.

# פסיקת עכבר

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

לפסיקה זאת מספר שימושים:

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

; Boot mouse

מקמו פקודות אלו לאחר ציור פלטת הצבעים.

הזיזו את העבר ובדקו שהסמן שלו זז.

mov ax, 0h

int 33h

; The mouse is visible

mov ax,1h

int 33h

שמרו את התבנית בשם חדש והמשיכו.

## העכבר

**מצב הרגיסטרים בהתאם ללחיצת העכבר:**

הפסיקה הקוראת את מיקום העכבר ואת סטטוס הלחיצה עליו:

mov ax,3h

int 33h

הפסיקה מחזירה את הערכים הבאים: -

**bx**מצב הלחיצה על כפתורי העכבר

0 – לא בוצעה לחיצת עכבר

1 – כפתור שמאלי נלחץ

2 – כפתור ימני לחוץ

**dx**מיקום העכבר, עמודה בין 0– 200 [y]

**cx**מיקום העכבר, שורה בין 0– [x] 640

(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)

בדיקה של מיקום העכבר על המסך והאם לחצו על הכפתור השמאלי שלו.

בלחיצה על הכפתור השמאלי נצייר פיקסל על המסך.

נחלק את רגיסטר cx ב – 2 כדי למקם אותו נכון במסך

ונפחית מרגיסטר dx 2 כדי שלא יצייר בדיוק על מיקום העכבר ונוכל לראות את הציור.

; Wait for Mouse

waitMS:

mov ax,3h

int 33h

cmp bx, 01h ; if Left click

jne waitMS

; Draw one pixel in Mouse place

shr cx,1 ; התאמת ערך לגודל המסך – חלוקה ב-2

sub dx, 2 ; חיסור 2 מערך העמודה

mov bh,0h

mov al,[color]

mov ah,0Ch

int 10h

מקמו פקודות אלו אחרי הפקודות של אתחול העכבר ולפני הפסיקה הממתינה למקש (ליציאה מהתכנית).

הריצו ובדוק שבהקלדה על הכפתור הימני של המסך תקבלו ציור של פיקסל אחד למסך במיקום של העכבר.

**שמרו את הקובץ בשם חדש המשיכו לעבוד על הקובץ החדש.**

## ציור עם העכבר

נשנה את הפעולה של העכבר:

לחיצה על הכפתור הימני של העכבר תקרא את הצבע עליו הוא נמצא, ולחיצה על כפתור שמאלי של העכבר תצייר על המסך עם הצבע שקראנו.

**bx**מצביע על מצב העכבר:

00b – לא בוצעה לחיצת עכבר

01b – כפתור שמאלי נלחץ

10b – כפתור ימני לחוץ

נשנה את הלולאה הממתינה למצב העכבר:

נפעיל את הפסיקה הקראת את המצב העכבר

נמקם אותו על המסך בהתאם למיקום שנוח לנו

נבדוק את רגיסטר bx

אם הוקש מקש ימני (2) של העכבר נקפוץ לתווית של קריאת הצבע

אם הוקש מקש שמאלי (1) של העכבר נקפוץ לתווי של ציור על המסך

אם לא הוקש על העכבר (0) נחזור לתווית של המתנה לעכבר.

waitMS:

mov ax,3h

int 33h

**cmp bx, 10b ; if Right click**

**je readColor**

**cmp bx, 01b ; if Left click**

**je draw**

**jz waitMS**

נוסיף למערך הצבעים ערך נוסף, 0

ערך זה ישמור את ערך הצבע שנקרא מהמסך ונוכל לקרא אותו כאשר נצייר את הפיקסל למסך.

color dw 2, 3, 4, 14, 15, **0**

את הנקודות על ציר ה-x וציר ה-y, יש לנו מהקריאה של העכבר.

נפעיל את הפסיקה של ציור פיקסל למסך כרגע בצבע שחור ערך 0.

readColor:

draw:

shr cx, 1 ; to position on x

sub dx, 1 ; to position on y

mov ax, [color+5]

xor bh, bh

**mov ah,0Ch**

**int 10h**

jmp waitMS

הריצו את התכנית, כל הזזה של העכבר מציירת על המסך בצבע שחור.

בנוסף איבדנו את היכולת לצאת בהקשה על מקש.

## יציאה מהתכנית בהקשה על Esc

לפני הפסיקה לחזרה למצב טקסט נוסיף תווית – textMod

textMod:

;Retern text mod

mov ax, 2

int 10h

נוסיף ללולאה הבודקת את העבר גם פסיקה הממתינה להקשה על המקלדת,

מתחת לבדיקה של האם ערך העכבר הוא 0 (אפס) jz waitMS -

; Wait for Mouse

waitMS:

mov ax,3h

int 33h

cmp bx, 10 ; if Right click

je readColor

cmp bx, 01 ; if Left click

je draw

**mov ah, 1h ; Wait for key press**

**int 16h**

jz waitMS

**mov ah, 0h ; Read key from keyboard to AL**

**int 16h**

**cmp al , 27 ; if Esc**

**je textMod**

**jmp waitMS**

## קריאת הצבע במיקום העכבר

readColor:

; read color from screen on mouse place to AL

mov bh, 0h

mov ah, 0dh ; get color of a single pixel.

int 10h

mov [color+5], ax ; AL = pixel color

jmp waitMS

**הריצו ובדקו**

בתכנית שלנו לחיצה על הכפתור השמאלי של העכבר דוגמת את הצבע עליו עומד העכבר.

לחיצה על הכפתור הימני של הכבר מציירת

מקש esc – יוצא מהתכנית.

# הרחבות

**שפרו את התכנית:**

ממתינים ללחיצת העכבר,

אם הלחיצה באזור פלטת הצבעים שמור את הצבע וחזור ללולאת המתן לעכבר.

אם לא צייר פיקסל.

הוסיפו צבעים לפלטת הצבעים

# קודה התכנית:

פתיח + משתנים

IDEAL

MODEL small

STACK 100h

**p186**

DATASEG

x\_coordinate dw ? ; place in line

y\_coordinate dw ? ; place in column

color dw 2, 3, 4, 14, 15, 0

x\_begin dw 0 ; Starting point on line

y\_begin dw 25, 50, 75, 100, 125 ; Starting point on column

x\_count db 35 ; loop count line draw

Y\_count db 25 ;loop count column draw

פעולות

CODESEG

proc drawPixel

pusha

xor bh, bh

mov cx, [x\_coordinate] ; place in line

mov dx, [y\_coordinate] ; place in column

mov ax, [color+si]

mov ah, 0ch

int 10h

popa

proc DrawRectangle

; Draw square at x\_begin, y\_begin position, size 20\*30

pusha

mov [x\_count],35

mov [Y\_count],25

mov ax,[x\_begin]

mov [x\_coordinate],ax

mov ax,[y\_begin + si]

mov [y\_coordinate],ax

line\_loop:

; Print one dot on screen

call drawPixel

inc [x\_coordinate]

dec [x\_count]

cmp [x\_count], 0

jnz line\_loop

mov ax,[x\_begin] ;reset line counters

mov [x\_coordinate], ax

mov [x\_count],35 ;reset column counters

inc [y\_coordinate]

dec [y\_count]

cmp [y\_count], 0

jnz line\_loop

popa

ret

endp DrawRectangle

ret

endp drawPixel

start:

תכנית ראשית

mov ax, @data

mov ds, ax

; --------------------------

; Enter Graphic mod 320 \* 200 pixels, 256 colors

mov ax, 13h

int 10h

; Drawing color squares

mov cx, 5

readColor:

; read color from screen on mouse place to AL

mov bh, 0h

mov ah, 0dh ; get color of a single pixel.

int 10h

mov [color+5], ax ; AL = pixel color

jmp waitMS

draw:

; Draw one pixel in Mouse place

shr cx,1 ; X ערך

sub dx, 1 ; Y ערך

mov ax,[color+5]

xor bh, bh

mov ah,0Ch

int 10h

jmp waitMS

textMod:

;Retern text mod

mov ax, 2

int 10h

; --------------------------

exit:

mov ax, 4c00h

int 21h

END start

mov si, 0

colorLoop:

call DrawRectangle

add si, 2

loop colorLoop

; Boot mouse

mov ax,0h

int 33h

; The mouse is visible

mov ax,1h

int 33h

; Wait for Mouse

waitMS:

mov ax,3h

int 33h

cmp bx, 10b ; if Right click

je readColor

cmp bx, 01b ; if Left click

je draw

mov ah, 1h ; Wait for key press

int 16h

jz waitMS

mov ah, 0h ; Read key from keyboard to AL

int 16h

cmp al , 27 ; if Esc

je textMod

jmp waitMS