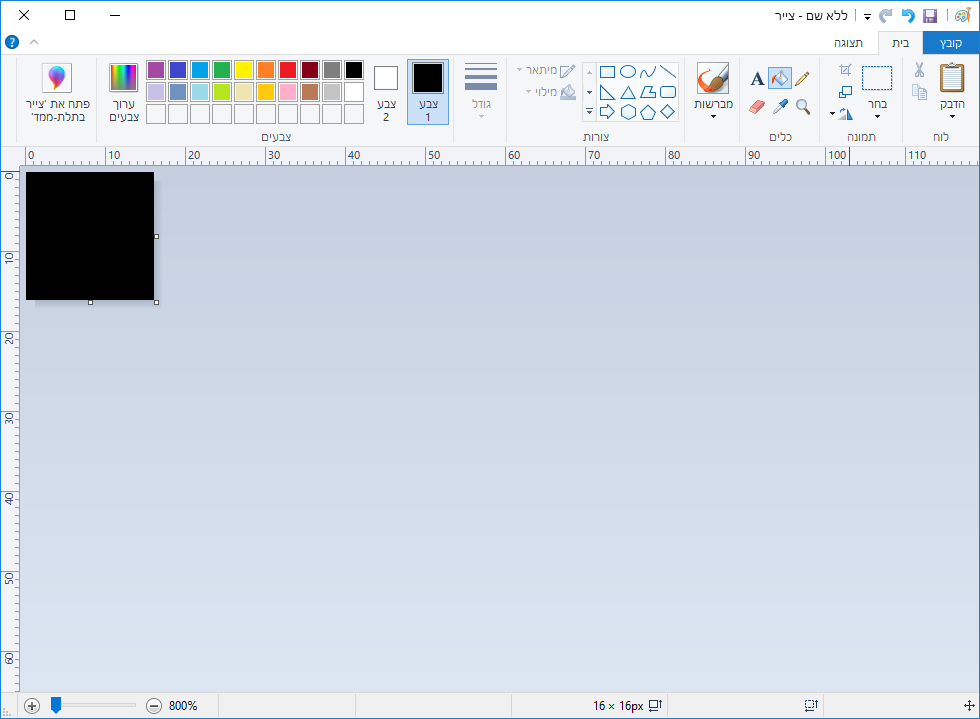
אנימציה - מורה

בדף זה נתנסה ביצירת תמונה גרפית בצייר, ניבא תמונה זו לאסמבלי, נזיז את התמונה על פני המסך עם החיצים וננסה לפגוע בה עם .

# ציור בתוכנת צייר.



נצייר בצייר ריבוע שחור

נקבע את גודל התמונה **לפיקסלים** (16\*16)

תפריט בית ⇦ שינוי גודל תמונה ⇦ נבחר בפיקסלים

נקבע את הגודל במישור האופקי (רחב התמונה) למספר המתחלק ב – 8.

נשמור את התמונה בפורמט של - **bitmap 256** ⇦ br.bmp

כדי שנוכל להשתמש בתמונה באסמבלי היא צריכה להיות שמורה במחיצה של tasm / bin

נצייר בצייר בתוך הריבוע השחור עיגול צבעוני. ⇦ pc.bmp 

ונשמור אותו לתיקיה של tasm / bin.

# הוספת תמונה

פסיקה להוספת תמונה באסמבלי היא ארוכה, אנחנו נתמקד כאן בהסבר החלקים החיוניים עבורנו.

הקובץ הכולל את הפסיקות להעלאת התמונה נקרא: **pic.asm**

המשתנה המכיל את שם התמונה – filename

שמרו את הקובץ בשם חדש ועדכנו את הפרטים של שם הקובץ, רוחב התמונה בפקיסלים, אורך התמונה בפיקסלים ומקמו את התמונה על גבי המסך (ציר x וציר y)

DATASEG

filename db 'pc.bmp',0 ; <<-- שם התמונה

BmpLeft dw 10 ; x cordinates of image left down corner

BmpTop dw 50 ; y cordinates of image left down corner

image\_height dw 16

image\_width dw 16 ; must be an 8 multiplication

CODESEG

start:

mov ax, @data

mov ds, ax

call graphicMode ;Enter graphic mode 320 \* 200

; Process BMP file

call OpenFile

call ReadHeader

call ReadPalette

call CopyPal

**הריצו את התכנית.**

**בהקשה על מקש התמונה תמחק ונחזור למצב טקסט.**

call CopyBitmap

call CloseFile

mov ah,0 ; Wait for key press

int 16h

mov ax,2 ; Return to text mode 80 \* 20

int 10h

exit:

mov ax, 4c00h

int 21h

END start

# מעבר בין שתי תמונות

לצורך יצירת אנימציה של הזזת התמונה על פני המסך יש לצייר מעל העיגול שיצרנו ריבוע שחור,

להזיז את מיקם תחילת ציור התמונה של העיגול הצבעוני על ציר ה – x או ציר ה – y או שניהם.

ולצייר מחדש את התמוה של העיגול.

נוסיף משתנה לשם התמונה ונשנה את שם המשתנה הקיים:

DATASEG

filename1 db 'pc.bmp',0 ; <<-- name of image file

filename2 db 'br.bmp',0 ; <<-- name of image file

הפרוצדורה הקראת לקובץ להדפסה על המסך היא:

היא הפרוצדורה הראשונה בקובץ.

**proc OpenFile**

; Open image file file

שכפלו את הפרוצדורה ושנו את שמות הפרוצדורות ל – OpenFile1 ו - OpenFile2

שנו את שמות המשתנים המצביעים על שם הקובץ ל - [filename1] ובפרוצדורה השנייה ל – [filename2].

וכמו כן את שם הלולאה למקרה של שגיאה ל - openerror1 ו - openerror2.

mov ah, 3Dh

xor al, al

**mov dx, offset filename**

int 21h

jc openerror

mov [filehandle], ax

ret

**openerror:**

mov dx, offset ErrorMsg

mov ah, 9h

int 21h

ret

endp OpenFile

בתכנית הראשית

נקרא לקובץ הראשון (ציור העיגול)

ונבדוק שהקובץ תקין.

start:

mov ax, @data

mov ds, ax

call graphicMode

; Process pc.btm file -->> Purple circle

**call OpenFile1**

call ReadHeader

call ReadPalette

call CopyPal

call CopyBitmap

call CloseFile

## הוספת השהייה

המחשב מאוד מהיר ואם אנו רוצים שנוכל לראות את המונה, עלינו להוסיף השהייה.

הוסיפו פרוצדורה של השהייה:

proc delay

mov al, 0

mov ah, 86h

mov cx, 1

mov dx, 1

int 15h

ret

endp delay

נוסיף בתכנית הראשית זימון (קריאה) לפרוצדורה של ההשהיה.  
וזימון לפרוצדורת המציירות את הריבוע השחור. [filename2]

call delay

; Process br.btm file -->> Black rectangle

**call OpenFile2**

call ReadHeader

call ReadPalette

call CopyPal

call CopyBitmap

call CloseFile

הריצו ובדקו את התכנית.   
אם עובדת טוב שמרו אותה בשם חדש והמשיכו לפקודות הבאות.

# הזזת השחקן על ציר x, - העיגול השחור מתקדם ימינה

המשתנה אשר קובע את המיקום שת התמונה על ציר ה – x הוא [BmpLeft].

אם נגדיל את המשתנה בשני פיקסלים הציור שלנו יזוז ימינה.

נוסיף בסוף הזימונים של הפקודות המציירות את התמונות את הפקודה

inc [BmpLeft]

נשלב את כל הפקודות שלנו בתוך לולאה בה העיגול הסגול ינוע עד שנלחץ מקש esc.

ראשית נקבע תווית ליציאה מהמשחק, **TheEnd:**

מתחת לתווית זו נשים את הפקודות למעבר למוד של טקסט.

TheEnd:

; Back to text mode

mov ah, 0

mov al, 2

int 10h

נוסיף לפסיקה הבודקת אם הקשנו על מקש את הבדיקה האם הקשנו על מקש Esc?

; Check Type Ahead Buffer Status

mov ah, 1h

נוסיף תווית **Move**:

בתחילת התכנית. לפני הזימון (קריאה לציור הראשון), אחרי הפקודה למעבר למוד גרפי של המסך.

בסוף הלולאה נוסיף קפיצה לתווית Move כדי שהשחקן שלנו ימשיך לזוז על פני הבמה.

int 16h

**jz Move**

; Read key pressed

mov ah, 0h

int 16h

; Was Esc pressed?

cmp al, 27

je theEnd

**jmp Move**

# התכנית הראשית

DATASEG

filename1 db 'pc.bmp',0 ; <<-- name of image file

filename2 db 'br.bmp',0 ; <<-- name of image file

BmpLeft dw 10 ; x ccordinates of image left down corner

BmpTop dw 50 ; y ccordinates of image left down corner

image\_height dw 16

image\_width dw 16 ; must be an 8 multiplication

**. . .**

CODESEG

; Procedures

**. . .**

start:

mov ax, @data

mov ds, ax

call graphicMode

Move:

; Process pc.btm file -->> Purple circle

call OpenFile1

call ReadHeader

call ReadPalette

call CopyPal

call CopyBitmap

call CloseFile

call delay

; Process br.btm file -->> Black rectangle

call OpenFile2

call ReadHeader

call ReadPalette

call CopyPal

call CopyBitmap

call CloseFile

inc [BmpLeft]

; Cheack type ahade buffer status

mov ah, 1h

int 16h

jz Move

; Read key pressed

mov ah, 0

int 16h

; Was Esc Pressed?

cmp al, 27

je TheEnd

jmp Move

TheEnd:

; Back to text mode 80\* 20

mov ah, 0

mov al, 2

int 10h

exit:

mov ax, 4c00h

int 21h

END start