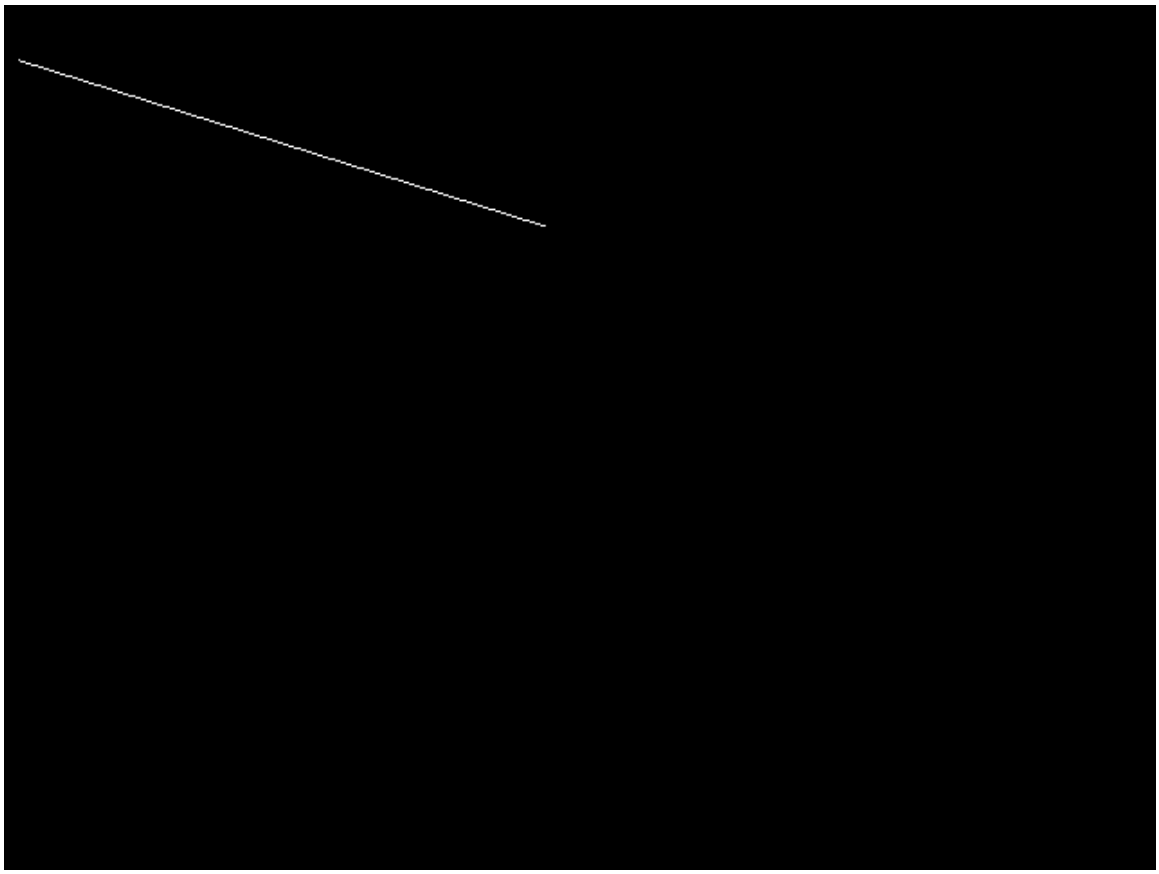
# Bressenham’s line algorithm

# בית הספר זיו ומרקס, ירושלים

מגיש: ינאי אלטר

ת"ז: 211377775

בהנחיית המורה: מאיר פנחס



הדר ב' תשעו

## תוכן עניינים:

## מבוא למשתמש:

אנו משתמשים הרבה בציור קו. במתמטיקה, במשחקי מחשב, סקיצות של גרפים, טבלאות וכו'. קו הוא למעשה אוסף של נקודות בשיפוע מסוים קבוע.
בפרויקט תוכלו להקליד 2 נקודות. נקודה (x1,y1) ונקודה (x2,y2). התוכנית תקלוט את הנקודות שביקשתם ותצייר את הקו הישר העובר ביניהן.
תוכלו ליצור ישרים בעלי שיפוע שלילי, חיובי, ישרים מקבילים לציר ה-x (שיפוע 0) וישרים מקבילים לציר הy.

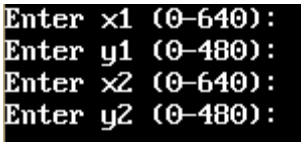
## מדריך למשתמש:

- בכניסה לפרויקט ייפתח לכם מסך שחור בו כתוב:
  "Enter X1 (0-640):"

  הקלידו את הx של הנקודה הראשונה של הקו ולחצו על מקש "Enter".

- לאחר מכן יהיה כתוב:
  "Enter Y1 (0-480):"
  הקלידו את הy של הנקודה הראשונה של הקו ולחצו על מקש "Enter".
- לאחר מכן יהיה כתוב:
  "Enter X2 (0-640):"
  הקלידו את הx של הנקודה השנייה של הקו ולחצו על מקש "Enter".
- לאחר מכן יהיה כתוב:
  "Enter Y2 (0-480):"
  הקלידו את הy של הנקודה השנייה של הקו ולחצו על מקש "Enter".

שימו לב שהקלדתם נתונים בטווח המבוקש. אחרת, התוכנה תציג הודעה מאחר ועברתם את גבולותיה.



(תמונת המחשה)

לאחר שהקלדתם 2 נקודות בטווח המבוקש, התוכנה תצייר את הקו מהנקודה הראשונה לנקודה השנייה.

ישנה דוגמא לציור קו בעמוד הראשון של הספר.

# מבוא למתכנת + הסבר על האלגוריתם

הפרויקט בנוי בשפת מחשב (אסמבלי). באסמבלי אין פקודה בנויה שנקרא לה והיא תצייר קו (למשל כמו בC#, שם יש DrawLine), אך ישנה פקודה לציור פיקסל (נקודה על המסך) לפי x,y מסוימים.

כאשר מקלידים למחשב שיעור x,y של נקודה, כל איבר במספר נרשם כתו בזיכרון כקוד Ascii ב(16) Hexadecimal.
למשל, נניח והקלדנו למחשב שהx של הנקודה הראשונה יהיה 321, זה יהיה רשום בזיכרון כך : 31 32 33 (3 בקוד Ascii בבסיס 16 זה 33, 2 זה 32, 1 זה 31).
לכן צריך לבנות פונקציה שתיקח את המידע שהוקלד ותהפוך אותו למספר אחד (בבסיס 16 כמובן) שכתוב בזיכרון ולא לרצף של תווי Ascii, כלומר במקרה הזה, שישמר בזיכרון הערך 141 (321 בבסיס 16) ולא 31 32 33.

בנוסף לכך, אי אפשר לצייר פיקסל בנקודה המכילה בערכים שלה שבר (למשל הנקודה A(3 ,1.5) ). דבר זה יוצר בעיה משום שלא יהיה אפשר לצייר קווים בשיפועים לא שלמים.
לכן ברסנהם בנה אלגוריתם שימתח קו בין 2 נקודות גם אם השיפוע של הקו הוא לא מספר שלם.
לפי האלגוריתם :

- כאשר השיפוע שלם, ניצור קו ישר רגיל (אפשר לצייר כל נקודה בקו כפיקסל).
- כאשר השיפוע לא שלם ממש (רציונאלי לא שלם), נוצר רצף של נקודות שביחד יוצרות קו כמעט ישר בין הנקודה הראשונה לשנייה. (מכיוון שחלק מהנקודות של הקו הם שבר, הקו לא ישר לגמרי, אך קרוב להיות ישר).

התוכנה היא מימוש האלגוריתם, הסבר לעומק על האלגוריתם ראו בהמשך הספר.

## 5 אלגוריתמים מרכזיים

1) **אלגוריתם קליטת מספר -** כמו שנאמר במבוא, כאשר מקלידים מספר כלשהו באסמבלי, כל יחידה במספר נשמרת בזיכרון בנפרד כקוד Ascii בבסיס 16.
כדי להפוך את רצף תווי הasciii בבסיס 16 למספר הרגיל בבסיס 16, נבצע את האלגוריתם הבא:
0.0 שים באוגר ax את הערך A (10 בבסיס 10)
1.0 הוסף לאוגר bx את ערך הasciii של התו הראשון במספר (ערך הasciii של התו הכי שמאלי)
1.1 הורד מbx, 30h (המרה מasciii למספר רגיל).
1.2 תכפיל את ax בbx (mul bx).
1.3 תקלוט בbx את ערך ax
1.4 כל עוד התו בזיכרון הוא חלק מהקלט של המשתמש, קפוץ חזרה ל- 0.0.
1.5 תשמור את bx במשתנה שמוקצה לו מקום בזיכרון.

נבצע את האלגוריתם על המספר 321 לצורך דוגמה:

ערך בזיכרון: 31 32 33 (בסיס 16)

ax = A

bx = 33

נורד מbx 30h (30 בבסיס 16)

bx = 3

נכפיל בax

bx = 1E

נוסיף 32 (בסיס 16)

bx = 50

נוריד 30 (בסיס 16)

bx = 20

נכפיל בAX

bx = 140

נוסיף 31 (בבסיס 16)

bx = 171

נוריד 30 (בבסיס 16)

bx = 141 (16) = 321(10)

(עכשיו הגענו למספר הרצוי, ואפשר להעביר אותו למשתנה כלשהו השמור בזיכרון).

2) **<u>אלגוריתם מציאת cases:</u>**

לקו ישר יש 8 מקרים:

1) X1<x2 && y1<y2
2) X1<x2 && y1>y2
3) X1>x2 && y1<y2
4) X1 > x2 && y1 >y2
5) X1 == x2 && y1 < y2
6) X1 == x2 && y1 > y2
7) Y1 == y2 && x1<x2
8) Y1 == y2 && x1 > x2

על מנת למתוח קו, המחשב מקבל 2 נקודות שהמשתמש מקליד (x1,y1),(x2,y1) והוא צריך לדעת איזה מקרה הן מקיימות על מנת להמשיך את החישובים של הקו. לכן מקצים בזיכרון משתנה case, ובודקים את כל המצבים הללו, אם אחד מהם מתקיים שמים בcase את הערך של המצב.

האלגוריתם:

0.0 הקצה משתנה case בזיכרון

1.0 השווה את x1 לx2 אם התוצאה שלילית (x1<x2) קפוץ ל2.0, אם התוצאה חיובית (x2<x1), קפוץ ל3.0, אם התוצאה 0 (x1==x2), קפוץ ל4.0

2.0 השווה את y1 לy2, אם התוצאה שלילית (x1<x2 && y1<y2) ,case=1
2.1 אם התוצאה חיובית, (x1<x2 && y1>y2), case = 2
2.2 אם התוצאה 0 (x1<x2 && y1==y2), case = 7

3.0 השווה את y1 ל y2, אם התוצאה שלילית (x1>x2 && y1<y2), case = 3
3.1 אם התוצאה חיובית (x1>x2 && y1>y2) , case = 4
3.2 אם התוצאה 0 (x1>x2 && y1==y2), case = 8

4.0 השווה את y1 לy2, אם התוצאה שלילית (x1==x2 && y1<y2), case = 5
4.1 אם התוצאה חיובית (x1==x2 && y1>y2), case = 6

3) **<u>אלגוריתם העיגול (כלפי מעלה או מטה)</u>**

כמו שנאמר במבוא, מכיוון שאי אפשר לצייר נקודה בy לא שלם (שבר), למשל, אי אפשר לצייר נקודה ב (1,3/4), צריך לעגל למספר שלם הכי קרוב. כלומר, במקרה הזה, המחשב יעגל או ל 1 או ל-0.

**אז איך המחשב ידע לאיפה לעגל**? ניקח לדוגמה את הנקודות (4,3), (8,4). השיפוע של הישר בין הנקודות יהיה רבע:

m = (y2-y1)/(x2-x1) = (4-3)/(8-4) = ¼.

מכאן כי לכל x שעולה באחד, y יעלה ב1/4, ולכן הנקודה ( ¼ 5,3) היא חלק מהישר הנ"ל.

כעת, המחשב צריך לבחור איזה נקודה הוא יצייר, (5,3) או (5,4). לשם כך נבנה אלגוריתם שיעגל:

**<u>האלגוריתם:</u>**

הכפל את המונה של השארית ב2 ולאחר מכן תחלק אותו במכנה, אם תוצאת החילוק יצאה 1 (לא כולל השארית), צייר פיקסל במקום ה(x,y+1), אחרת צייר פיקסל ב(x,y). כמובן שהy מספר שלם מאחר ולא מתייחסים לשארית. סוף האלגוריתם.

<u>נממש את האלגוריתם על הנקודה למעלה:</u>

נכפיל את המונה של 1/4 ב2 ונחלק במכנה. נקבל ½ . כלומר 0 ושארית ½ . אם לא נתייחס לשארית, התוצאה לא יצאה 1. ולכן זה יצייר נקודה ב(x,y). כלומר ב(5,3).

ניקח את הנקודה הבאה בישר למעלה, ( ½ 6,3) (העלנו את x ב-1 ואת y בשיפוע). ניקח את השארית (1/2) ונכפיל את המונה ב2 ולאחר מכן נחלק במכנה. יצא 2/2 = 1. כלומר, 1 ושארית 0. תוצאת החילוק יצאה 1 ולכן נבחר ב(x,y+1) כלומר, (6,4). כך הלאה עד שמגיעים לנקודה שהאיקס שלה לא בין גבולות האיקסים בין הנקודות.

4) **<u>אלגוריתם ניקוי מסך</u>**

לאחר שהמחשב מקבל מהמשתמש כקלט 2 נקודות ושומר אותם במשתנים, הוא צריך לנקות את המסך על מנת להציג את הקו. לכן צריך לכתוב פונקציה שתנקה את המסך.

<u>האלגוריתם :</u>

0.0 עבור למצב טקסט (text mode)
1.0 תרד שורה
2.0 בצע את 1.0 24 פעמים (גודל הפעמים שצריך לרדת שורה כדי שהמסך יתנקה.
3.0 עבור חזרה למצב גרפי (graphic mode).

5) **<u>אלגוריתם מציאת נקודה הבאה:</u>**

אחרי כל ציור נקודה, אנו צריכים למצוא את הנקודה הבאה. כדי למצוא את הנקודה הבאה נצטרך לעלות הx של הנקודה הקודמת ב-1, ואת הy של הנקודה הקודמת בשיפוע. מאחר ובאסמבלי אי אפשר שy יהיה מספר לא שלם, נצטרך להגדיר 2 משתנים, האחד יהיה מונה המספרים השלמים, השני יהיה מונה השיפועים. כלומר, נניח שיש לנו ישר עם שיפוע ¼ המתחיל מנקודה (0,0). נקרא למשתנה המספרים השלמים y, ולמשתנה של מונה השארית yS (y שארית). נעקוב בעזרת טבלת מעקב אחרי הנקודות של הישר:

| x | Y | yS |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 2 | 0 | 2 |
| 3 | 0 | 3 |
| 4 | ? | ? |

אך מה קורה כאשר מונה השארית מגיע למכנה השארית? כלומר, מה קורה כשהמונה חלקי מכנה השארית הוא 1 (מספר שלם)? (מכנה השארית הוא deltax לפי נוסחת שיפוע ישר בה המכנה הוא deltax). נוצר בשארית מספר שלם ולכן נצטרך להוסיף אחד למונה המספרים השלמים ולאפס את מונה השארית. כלומר בטבלה יופיע:

| | | |
|---|---|---|
| 4 | 1 | 0 |

ולא:

| | | |
|---|---|---|
| 4 | 0 | 4 |

שכן 4/4 = 1 (מספר שלם) ולכן צריך להתייחס אליו כמספר שלם ולא כשבר ולהוסיף אותו למונה המספרים השלמים.

<u>האלגוריתם</u>:

0.0 הקצה מקום בזיכרון למשתנה y

0.1 הקצה מקום בזיכרון למשתנה yS (y שארית)

1.0 העלה את y במספר השלם של שיפוע הישר

1.1 העלה את yS במונה השבר של שיפוע הישר (במידה והשיפוע מספר לא שלם).

1.2 תחלק את yS ב-deltaX

1.2.1 אם יצא מספר שלם 1, תאפס את yS, ותוסיף לy אחד.

1.2.2 אם יצא שבר, תשאיר את yS כמו שהיה לפני החילוק (החילוק מתבצע דרך אוגרים כך שyS יישאר כמו שהוא).

**<u>טענות כניסה ויציאה</u>**:

Part 1 –ניקוי הלוח (ע"פ אלגוריתם ניקוי לוח), מעבר לGraphic Mode, הדפסת הוראות מהמשתמש, קליטה מהמשתמש x1, y1, x2, y2 אם הוזנו מספרים חוקיים (בגבולות הx,y של התוכנית), המספרים ישמרו בזיכרון, אם הוזנו מספרים לא חוקיים, תוצג הודעת שגיאה (אלגוריתם קליטת מספר).

Part 2 – בדיקה איזה מצב (case) מתארות הנקודות שהתקבלו מהמשתמש, ושמירת המצב במשתנה case המוקצה בזיכרון המחשב (אלגוריתם מציאת cases).

Part 3 – חישוב delta x, delta y לפי המצב של הישר ושמירתם בזיכרון ב2 משתנים.

Part 4 – העלאת x באחד, y במספר השלם של שיפוע הישר, yS במספר הלא שלם של שיפוע הישר (אלגוריתם מציאת נקודה הבאה על הישר), חילוק yS בdeltaX ובדיקה האם מתקבל מספר שלם 1, אם כן העלאה של y באחד ואיפוס yS אם לא, להשאיר את הערכים כמו שהם (מימוש האלגוריתם מציאת נקודה הבאה של ישר. לאחר מכן בדיקה האם לצייר את (x,y) או את (x,y+1) (אלגוריתם עיגול כלפי מעלה או מטה).

Part 5 – בדיקה אם ציור הקו הסתיים, כלומר, אם הגענו לנקודה שהיא לא בין 2 הנקודות שהוזנו. אם סיימנו לצייר את הקו התוכנית תסתיים, אם לא סיימנו התוכנית תחזור לPart4.