

**פרויקט גמר בארגון המחשב ושפת סף - הערכה חלופית - מדעי המחשב** 

מגישה: הילה בלומן

כיתה: י׳3

מורה: צביקה שטרקמן

תוכן עניינים:

עמוד 0 : שער.

עמוד 1 : תוכן עניינים.

עמוד 2 : הסבר על מה זה Snake

עמוד 3 : למה בחרתי בזה.

עמוד 4 : תרשים זרימה כללי של הקוד.

עמוד 5-8: הסבר על 3 פרוצדורות מרכזיות.

עמוד 9-10 : הנחיות למפעיל.

עמוד 11 : קשיים, בעיות ופתרונות.

עמוד 12 : רפלקציה וסיכום.

מה זה Snake?

תיאור כללי:

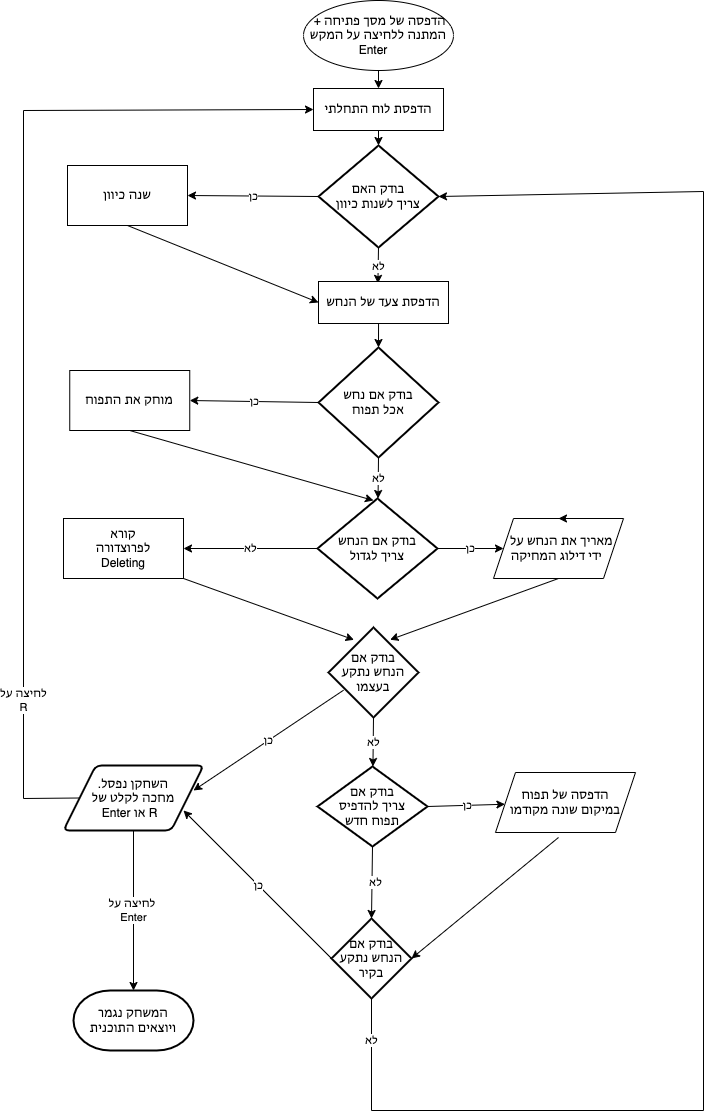
המשחק SNAKE הוא משחק לשחקן אחד, בו השחקן שולט בדמות לבנה על המסך שנראית כמו נחש. המטרה של המשחק היא להגיע לניקוד הגבוה ביותר, כל אכילת תפוח מזכה בנקודה אחת. בנוסף, כאשר הנחש אוכל תפוח הוא מתארך. אם הנחש יפגע בקיר או בעצמו הוא נפסל והמשחק נגמר. 

⬇️לאחר דקה בערך של משחק⬇️



למה בחרתי בנושא הזה?

בחרתי בנושא זה מכיוון שרצית לשחזר בקוד את אחד מהמשחקים שהייתי משחקת בהם כילדה קטנה. המשחק snake מזכיר לי את התקופות הטובות שהיו לי עם בני דודים שלי, לכן החלטתי ללכת על זה וליצור את המשחק.

תרשים זרימה (יישום הפרויקט)

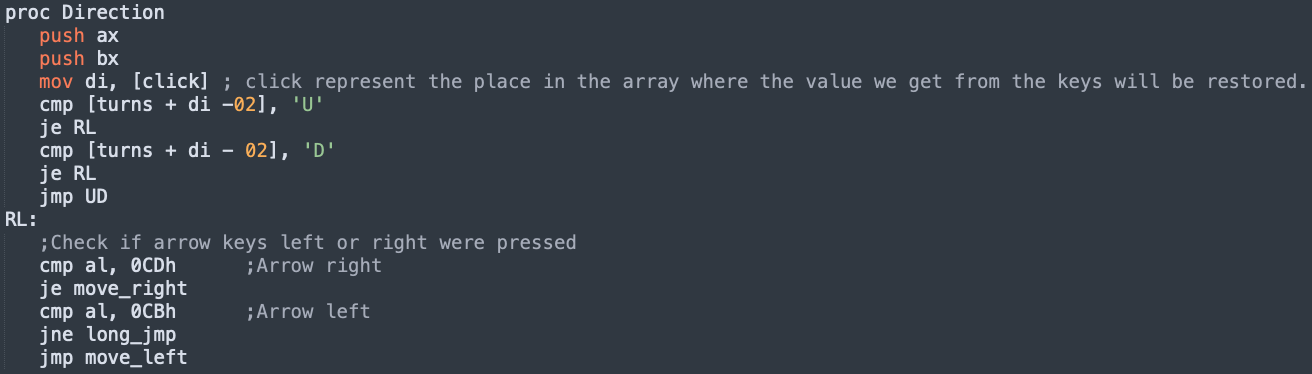
פרוצדורות עיקריות

Direction

מטרתה הפרוצדורה Direction היא לקבוע את כיוון ההתקדמות של הנחש.

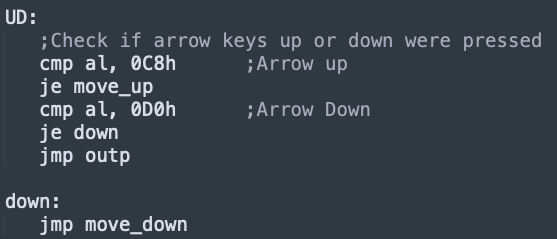
בהתחלה נבדקת תקינות הקלט מהמשתמש (כלומר אם כיוון ההתקדמות כרגע הוא למעלה הפרוצדורה תאפשר לקבל רק קלט של ימינה או שמאלה ותתעלם משאר הקלטים). לאחר מכן הפרוצדורה משנה את ערכי המשתנים בהתאם לכיוון ההתקדמות הקיים ולקלט שקיבלה מהמשתמש במידה שהתקבל. הפרוצדורה משתמשת במשתנים שונים ומערכים:

משתנים רגילים: snakeDir, currentX, currentY, click

מערכים: delCounter, deleteY, deleteX, turns 

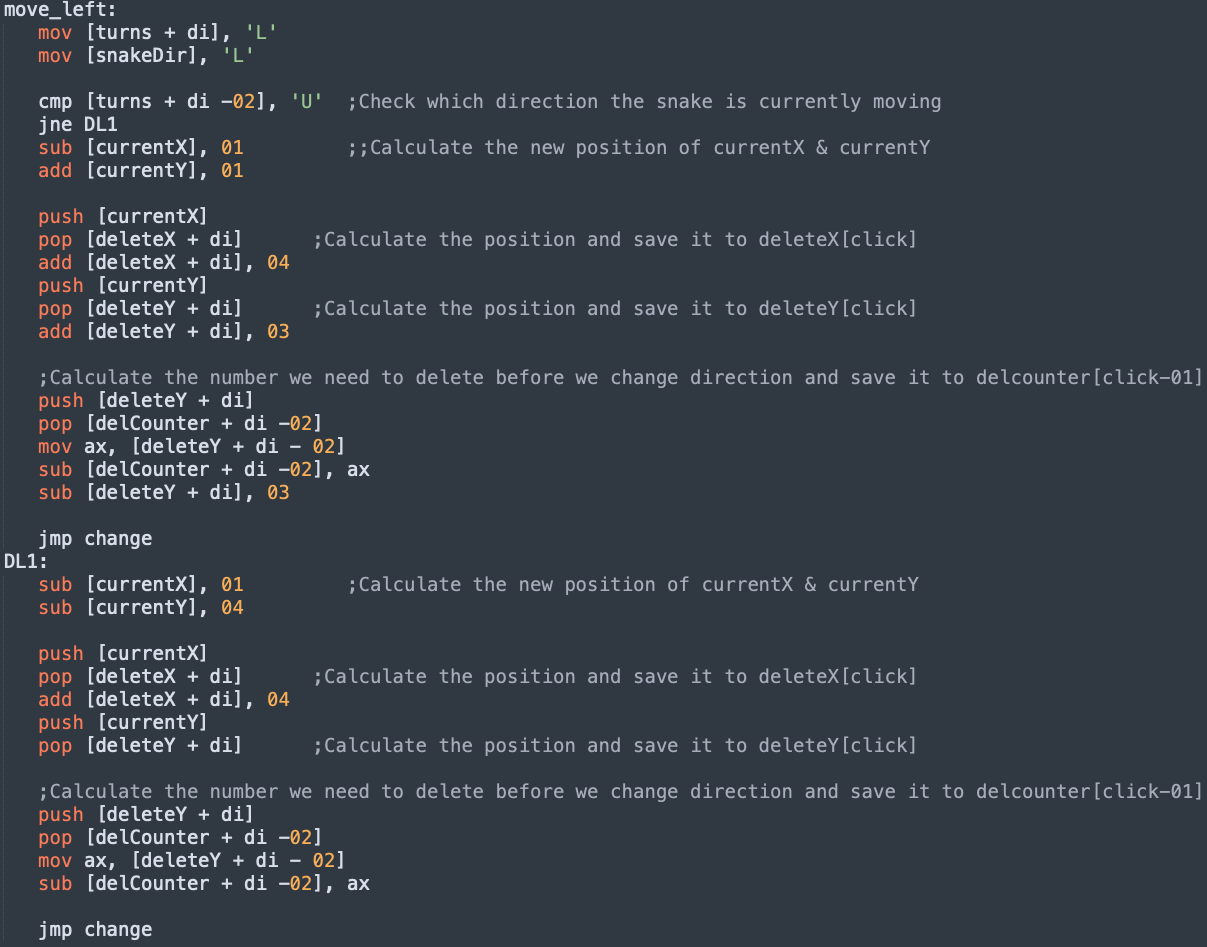
בודק מה הוא כיוון ההתקדמות הנוכחי של הנחש ולפי זה קופץ לבדיקה נכונה של הקלט.

בדיקה של הקלט במצב בוא הנחש זז למעלה או למטה

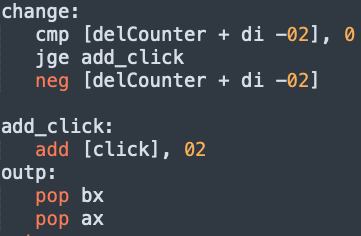


בדיקה של הקלט במצב בוא הנחש זז ימינה או שמאלה

דוגמה לשינוי ערכים:



לאחר כל שינוי כיוון חוקי וביצוע השינויים הדרושים (כמו בדוגמה למעלה) הקוד הבאה רץ:



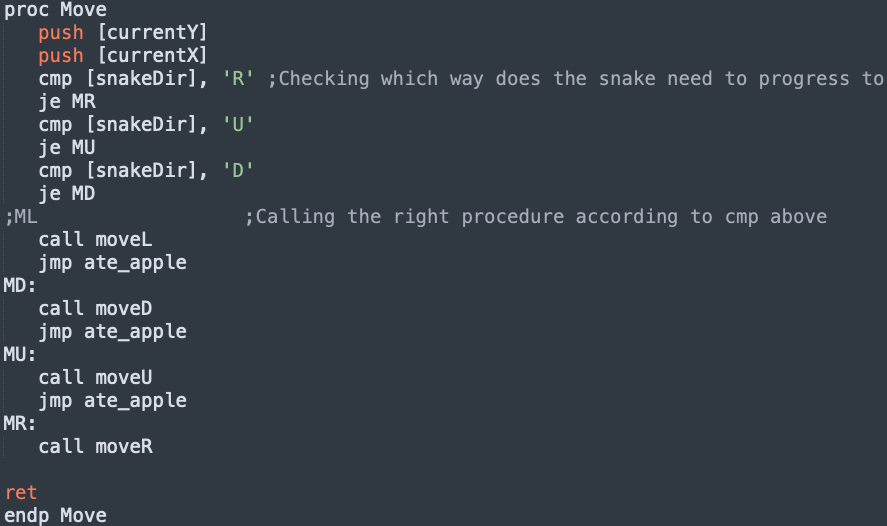
בודק אם החישוב של ה- delcounter שלילי אם כן הופך אותו לחיובי (הסבר על המערך delcounter בתיאור הפונקציה Deleting)

מקדם את click אם הקלט שנקלט היה חוקי (הסבר על המשתנה בתיאור הפונקציה Deleting)

Move

לאחר שהפרוצדורה Direction קבעה את כיוון ההתקדמות של הנחש.

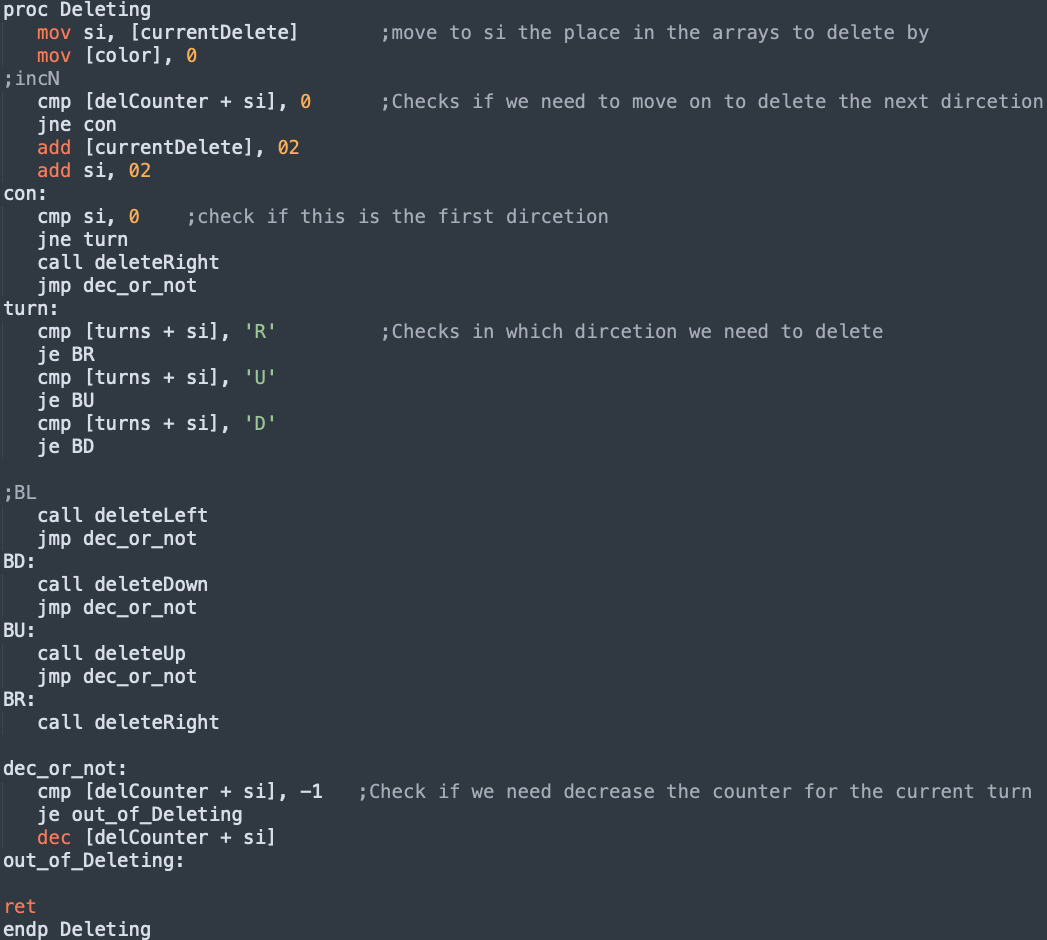
הפרוצדורה Move תקדם את הנחש לכיוון הנכון, בתחילתה מתבצעת בדיקה של הכיוון הנוכחי של הנחש (snakeDir) וקריאה לפרוצדורה המתאימה. בתחילת הפרוצדורה נכנסים למחסנית הערכים currentX ו- currentY הערכים האלה משומשים בפרוצדורת שנקראות בהמשך הפרוצדורה (moveL, moveR, moveU, moveD)



Deleting

מטרת הפרוצדורה Deleting היא למחוק את הזנב של הנחש, הפרוצדורה עושה זאת בעזרת כמה משתנים:

* currentDelete שומר את המיקום במערכים לפיו יש למחוק,
* המערכים deleteX ו- deleteY שומרים את המיקום המדויק של הפיקסלים על הנחש אותם יש למחוק.
* המערך delCounter שבו שמורים כמות הפעמים שיש למחוק מאותו כיוון עד שיש שינוי בכיוון המחיקה.



הנחיות למפעיל

איך מפעילים:

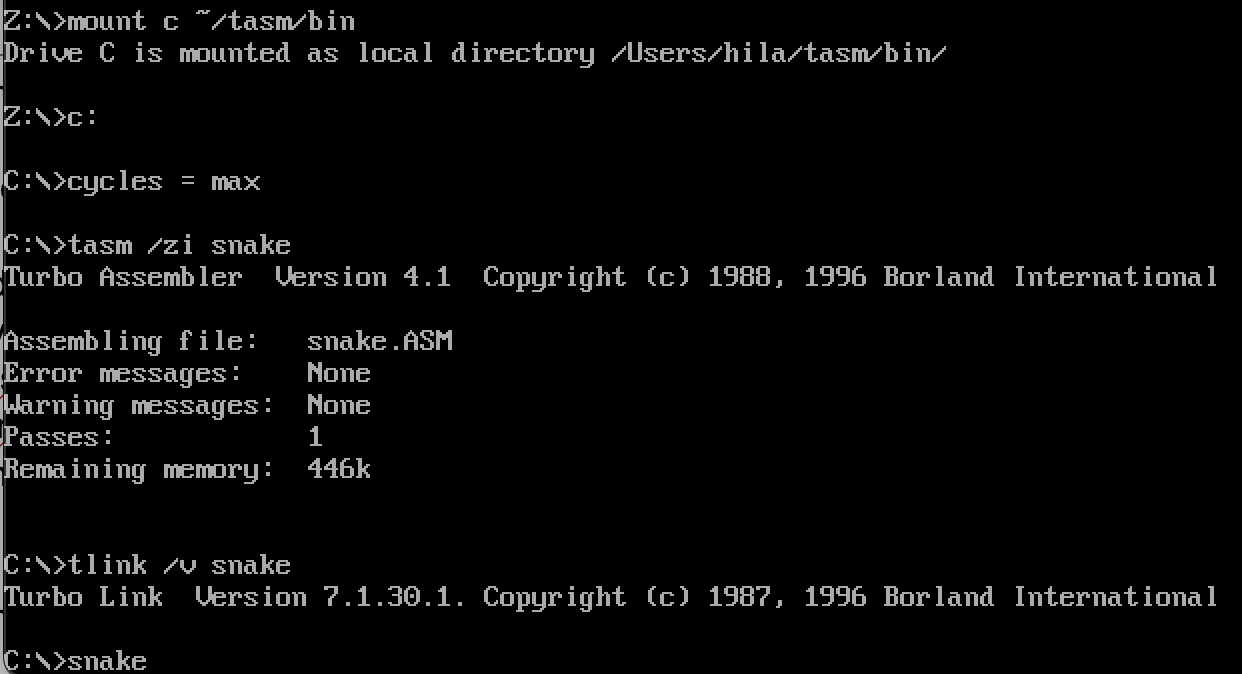
ל - Windows

צריך לפתוח את ה - Dosbox ולאחר מכן להכניס את הפקודות הבאות:

* mount c: c:/
* c:
* cd tasm/bin
* cycles = max
* tasm /zi snake
* tlink /v snake
* snake

ל - macOS

צריך לפתוח את ה - Dosbox ולאחר מכן להכניס את הפקודות הבאות:

* mount c ~/tasm/bin
* c:
* cycles = max
* tasm /zi snake
* tlink /v snake
* snake

מה צריך לקרות:

לאחר ביצוע הוראות ההפעלה למעלה, יופיע מסך הפתיחה עם הוראות המשחק, מטרת המשחק וחוקי המשחק. לאחר מכן פשוט לוחצים על Enter ומתחילים לשחק.

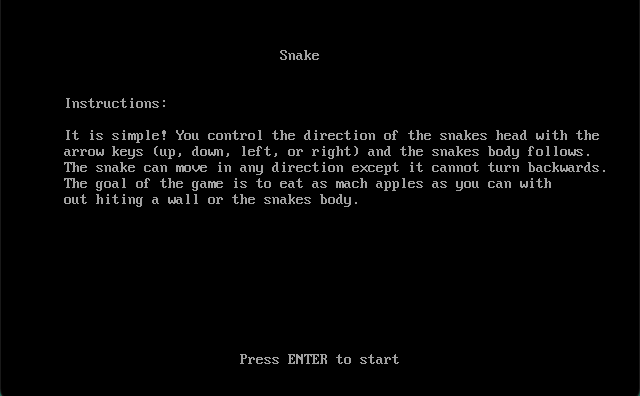
הוראות המשחק:

אתה שולט בכיוונו של ראש הנחש עם מקשי החצים (למעלה, למטה, שמאלה או ימינה), וגוף הנחשים עוקב.

הנחש יכול לנוע לכל כיוון, אך אינו יכול להסתובב לאחור(כלומר עם הוא זז ימינה הוא לא יזוז שמאלה גם אם נלחץ על המקש שמאלה).

בנוסף אם הנחש מתקדם ימינה אין צורך להמשיך ללחוץ על המקש ימינה הוא מתקדם אוטומטית אם אין שינוי, לחיצה על המקש ימינה לא תשנה כלום.

מטרת המשחק היא לאכול כמה שיותר תפוחים בלי לפגוע בקיר או בגוף הנחש.



מגבלות ידועות:

1. קיימת מגבלה של מקסימום 300 פניות למשחק.
2. יש מגבלה של ניקוד - הניקוד יכול להגיע רק עד 99.

קשיים, בעיות ופתרונות

לאורך הפרויקט לא היו לי הרבה בעיות וקשיים, נתקלתי בשני אתגרים מעניינים.

מחיקת הנחש לאחר יותר מפניה אחת -

הקושי היה שהגרסה הראשונית של המחיקה לא אפשרה יותר מפניה אחת בכל פעם. הגרסה לא אפשרה את זה כיוון שהיא הסתמכה על currentX, currentY ועל כמות קבועה של פעמיים שיש לחזור על המחיקה של הזנב.

הפתרון שלי לבעיה זו היה להשתמש במערכים אשר ישמרו בכל פנייה את הנתונים הנדרשים (מיקום הפיקסל שממנו מוחקים, כמות המחיקות שיש לבצע, וכיוון המחיקה) ושני משתנים, אחד אשר שומר את המיקום במערכים לפיו יש למחוק והשני ששומר המיקום במערך שבו הערכים לאחר פניה יוכנסו. בעזרת המערכים ושני המשתנים הצלחתי לקודד קוד למחיקה של זנב הנחש שעובד גם במצב של פניות מרובות לא משנה מהו אורכו של הנחש.

הדפסה רנדומלית של תפוחים - להדפיס תפוחים במקומות שונים על המסך (רנדומליים), קושי זה נוצר משתי סיבות:

1. כתיבת קוד לקבלת מספרים רנדומליים - ניסיתי לכתוב קוד שעושה זאת והתקשיתי מאוד.
2. לדאוג שהתפוח לא יודפס מחוץ לקירות או עליהם - השימוש בקוד שבודק את זה היה מאט מאוד את התוכנית ולא יעיל.

פתרון - בסוף השתמשתי במערך של מיקומי תפוח שונים מוכנים מראש ובקוד שיוצר מספרים רנדומלי (שמצאתי בספר). בעזרת המספרים הרנדומלים נבחר כל פעם מיקום אחר במערך שלפיו מודפס תפוח.

רפלקציה וסיכום

איך היה לי הפרוייקט:

התחלתי את הפרויקט עם הרבה ביטחון בידע שלי וההבנה שלי החומר, אבל לאורך הזמן הבנתי שחסר לי ידע כדי לקודד את הפרויקט הזה, למרות זאת המשכתי לעבוד, לטעות, לתקן וללמוד. נתקלתי בכמה בעיות בתכנון ובבאגים בקוד בעיקר על המיקום של הנחש (התקדמותו ומחיקתו), את רובם פתרתי תוך כמה דקות או שעות, אך היו מספר בעיות שלקחו לי כמה ימים לפתור רעיונית ולממש. ואפילו במסגרת הפרויקט תכננתי אלגוריתם לפתרון הבעיה.

לאורך הפרויקט היו לי עליות וירידות רבות, זמן רב הושקע בפרויקט ואני שמחה מאוד מהתוצר הסופי ומתגאה בעובדה שקודדתי משחק בעצמי.

מה הייתי עושה אחרת:

בתחילת הפרויקט קפצתי לתוך הפרויקט ישר התחלתי לכתוב את הקוד לפי הרעיון שהיה לי. אם הייתי צריכה לעשות את הפרויקט מחדש לא הייתי מתחילה בכתיבת הקוד לאור התכנון הראשוני, אלה הייתי כותבת תכנון יותר מסודר על דף, רושמת איך הייתי עושה כל חלק בנפרד.   
אם הייתי עושה את זה בדרך הזו הייתי חוסכת לעצמי הרבה זמן וכאבי ראש.

מטלת המשך

המטלה שקיבלתי מצביקה לאחר הצגת הפרויקט היה להפוך את הנחש למהיר יותר. עשיתי זאת על ידי הדפסה ומחיקה של שתי עמודות/שורות של פיקסלים במקום אחת. בעקבות שינוי זה הייתי צריכה לשנות עוד כמה דברים בקוד על מנת שהוא יפעל כמו שצריך.