בית הספר להנדסה ומדעי המחשב ע"ש רחל וסלים בנין

מבוא למדעי המחשב 67101 – סמסטר א' 2021
תרגיל 10 – **Snake**
להגשה בתאריך **21/12/2021** בשעה 22:00

בתרגיל זה תתבקשו לממש את המשחק **Snake**.

הנחיות כלליות:

- את התרגיל **חובה להגיש בזוגות**. ניתן למצוא שותפים בפורום חיפוש השותפים באתר. כדאי להתחיל לעבוד על התרגיל מוקדם!
- עליכם לממש את החלק הראשי של פתרונכם בקובץ snake_main.py. ניתן כמובן להוסיף קבצים נוספים ובתוכם פונקציות ומחלקות כרצונכם עבור חלקים נוספים מהמימוש.
- לצורך התרגיל מימשנו עבורכם את המחלקה GameDisplay הנמצאת בקובץ **game_display.py**, מחלקה זו אחראית על ציור האובייקטים למסך ועוד פונקציות של המשחק ואתם מחויבים להשתמש בה במהלך כל התוכנית - אין לשנות מחלקה זו (**בפרט אין להגיש את הקובץ בו היא ממומשת**).
- כדי להריץ את התוכנית, עליכם להריץ את הקובץ game_display.py. הפונקציה הראשית ב-snake_main.py שאתם תממשו תופעל מתוך ההרצה הזו.
- כמו כן, עליכם להשתמש בפונקציות המאפשרות לייצר אובייקטים בצורה אקראית הממומשות עבורכם בקובץ **game_parameters.py** – אין לשנות קובץ זה **ואין להגיש אותו**.
- יצרנו עבורכם קובץ snake_main.py לדוגמא, המדגים את השימוש במחלקה GameDisplay ואת דרך העבודה עם הממשק הגרפי.

הסבר על **snake_main.py** ולולאת המשחק

הקובץ הראשון שעליכם להגיש הוא snake_main.py . מחלקה זו היא המחלקה הראשית של המשחק והיא אמורה לייצר את האובייקטים השונים של המשחק, להכיל מימוש של האינטראקציות השונות ביניהם ולדאוג לרצף פעולות תקין של המשחק. בתוך קובץ זה ישנה מתודה שנקראת main_loop שבה אתם צריכים להשתלב.

```
def main_loop(gd: GameDisplay) -> None:
    # initialize your objects here

    while True:
       # Check key presses
       # Update of objects and their state for the next round
       # Draw the board
        gd.end_round()
```

שימו לב שהלולאה כרגע תרוץ לנצח, אבל ייתכן שתרצו לעצור את הלולאה תחת תנאים אחרים.

בכל שלב בלולאה התוכנית שלכם תבצע את הפעולות הבאות:

1. קריאת הקלט (לחיצות המקש של המשתמש) במידה ויש.
2. עדכון מיקום האובייקטים ומצב הלוח.
3. ציור הלוח לפי המצב הנוכחי.
4. קריאה לפונקציה game_display.end_round() הגורמת לממשק הגרפי להמתין זמן קצר עד תחילת הסיבוב הבא.

אנו מעודדים אתכם להפעיל שיקול דעת ולכתוב פונקציות נוספות שייעזרו לכם בפתרון כל משימה כך שהקוד שלכם יהיה אלגנטי ומודולרי. המטרה של התרגיל הינה כתיבת אובייקטים וחשיבה על האינטראקציה בין הישויות השונות בתוכנית. בהצלחה!

תיאור המשחק

לוח המשחק:

לוח המשחק מורכב מתאים מרובעים. לכל תא יש קואורדינטה **(x,y)** שמציינת את מיקומו. המספרים **x,y** הם מספרים שלמים אי שליליים. גודלו של הלוח נתון ע"י קבועים **WIDTH, HEIGHT** בקובץ **game_parameters.py**. כלומר, בלוח יש סה"כ **WIDTH** תאים לרוחבו ו-**HEIGHT** תאים לגובה. הקואורדינטות של התא בפינה השמאלית התחתונה הן **(0,0)** ואילו התא בפינה הימנית העליונה נמצא התא שהקואורדינטות שלו הן **(WIDTH-1, HEIGHT-1)**.

המחלקה **GameDisplay** מאפשרת לכם לצבוע תא בצבע מסויים ע"י קריאה למתודה

`draw_cell(self, x: int, y: int, color: str) -> None`

כאשר הצבע הוא אחד מהערכים: **"green", "red","orange", "black"**

הזמן במשחק מתקדם בסיבובים (שאורך כל אחד מהם הוא שבריר שנייה). בכל סיבוב עליכם לצייר מחדש את מיקום כל האובייקטים שבמשחק ולבסוף לקרוא לפונקציה **end_round** של מחלקת **GameDisplay** שתסיים את הסיבוב ותכין את הלוח לציור הסיבוב הבא.

תפוחים:

בתחילת כל סיבוב ישנם 3 תפוחים על הלוח. כל תפוח ממוקם במיקום אקראי על הלוח (x,y), גודלו בדיוק תא אחד, וצבעו ירוק. התפוח נשאר במקומו עד שהוא נאכל על ידי הנחש או מושמד ע"י פצצה. לכל תפוח מאותחל ניקוד שהנחש יכול לזכות בו. ברגע שגופו של הנחש נמצא באותו תא בו ממוקם התפוח, הנחש למעשה אוכל את התפוח. כאשר הנחש אוכל את התפוח, הניקוד שאותו תפוח מספק מצטרף לניקוד אותו צברנו במהלך המשחק, ואורכו של הנחש גדל ב-3 חוליות. מייד לאחר אכילתו (עוד באותו סיבוב), התפוח יופיע במיקום אקראי חדש ועם ניקוד התחלתי אחר.

את הגרלת המיקום והניקוד יש לעשות אך ורק בעזרת הפונקציה **get_random_apple_data()** מהקובץ **game_parameters.py**.

<u>מקרי קצה:</u>

- תפוח מוגרל על תפוח, או על גוף הנחש,או על הפצצה או על גלי ההדף שלה: עליכם לוודא שתפוח חדש ימוקם אך ורק בתא ריק.
- אין יותר מקום פנוי בלוח על מנת למקם תפוח: המשחק מסתיים.

פצצות:

בתחילת כל סיבוב ישנה פצצה אחת על הלוח. הפצצה תמוקם במיקום אקראי על הלוח ומוגדר עליה זמן עד לפיצוצה שמאותחל אקראית, ורדיוס פיצוץ אקראי. גודל הפצצה לפני פיצוצה הוא תא יחיד. הפצצה תצויר בצבע אדום עד לרגע הפיצוץ שלה, אז יצויר גל ההדף של הפיצוץ בצבע כתום.

אם **הפצצה** לפני פיצוצה או **גל ההדף** שלה נוגעים בגוף הנחש, המשחק מסתיים.

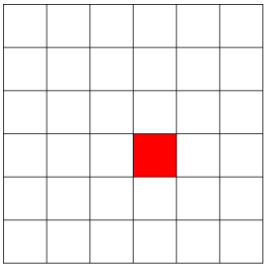
אם **גל ההדף** של הפצצה נוגע בתפוח, התפוח נעלם ונוצר מחדש במיקום אחר.

לאחר סיום הפיצוץ והתפשטות גל ההדף, הפצצה תופיע במקום אקראי חדש עם זמן לפיצוץ ורדיוס פיצוץ שהוגרלו מחדש. נתאר את סדר הפעולות: בסיבוב ה- $\boldsymbol{i}$ גל ההדף הגיע לערך המקסימלי של רדיוס הפצצה ובסיומו ציירנו את גל ההדף המקסימלי. אז בסיבוב ה $\boldsymbol{i+1}$ עלינו למקם על הלוח פצצה חדשה במיקום אקראי חדש,עם זמן לפיצוץ חדש ורדיוס חדש, ולאחר מכן לצייר את הפצצה החדשה על הלוח.

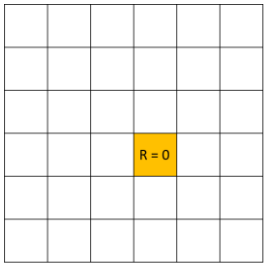
נראה דוגמה מפורטת להתקדמות הפצצה:

להלן פצצה שנוצרה עם 3 יחידות זמן עד לפיצוץ, והיא תצויר למסך במשך 3 סיבובים. בסיבוב הרביעי הפצצה לא תצויר יותר, אבל גל ההדף שלה יצויר ויתפשט מהנקודה ההתחלתית ברדיוס שהולך וגדל ב-1 בכל סיבוב. גל ההדף מתחיל ברדיוס 0 ומתקדם עד רדיוס הפיצוץ של הפצצה (כולל). בגל ההדף נמצאים התאים ברדיוס **R** מהפצצה ש"מרחק מנהטן" שלהם מתא המוצא שלה הוא בדיוק **R**. ובנוסחה מדויקת, גל ההדף מוגדר על ידי כל התאים בקואורדינטות $(\boldsymbol{u},\boldsymbol{v})$ כך ש מתקיים: $|\boldsymbol{x}-\boldsymbol{u}|+|\boldsymbol{y}-\boldsymbol{v}|=\boldsymbol{R}$.

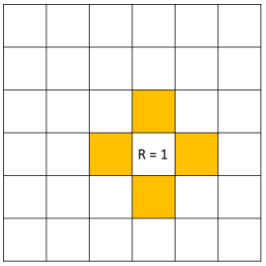
נראה דוגמה להתקדמות הפצצה בסיבובי המשחק (i) כאשר מספר יחידות הזמן לפיצוץ הוא 3 והרדיוס הוא 2:



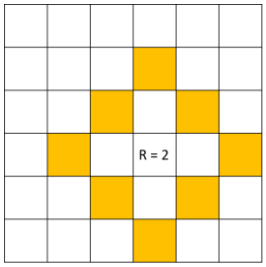
$i=\{1,2,3\}$



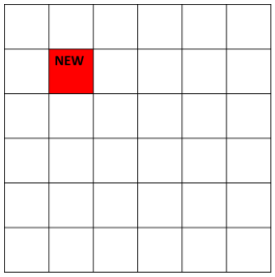
$i=4$



$i=5$



$i=6$



$i=7$

את הגרלת המיקום, זמן הפיצוץ והרדיוס של הפצצה יש לעשות אך ורק באמצעות **get_random_bomb_data()** מהקובץ **game_parameters.py**

<u>מקרי קצה:</u>

- פצצה או גל ההדף שלה מעלימים תפוח: עבור כל תפוח שנעלם, עליכם למקם אותו מחדש על הלוח באותו סיבוב
- התקדמות גל ההדף של הפצצה מעבר לגבולות הלוח:
  - אם גל ההדף של הפצצה **יוצא אל מחוץ** לגבולות הלוח ועדיין לא הגענו לערך **הרדיוס** של הפצצה, הפצצה מסיימת את חייה ותופיע במקום אקראי חדש עם זמן לפיצוץ ורדיוס פיצוץ שהוגרלו מחדש.
  - אין לצייר את הפצצה או גל ההדף שלה מחוץ לגבולות הלוח.
- הגרלות הפצצה:
  - פצצה מוגרלת על תפוח או נחש: עליכם לוודא שהפצצה תמוקם בתא ריק.

הנחש:

תיאור ומיקום התחלתי:

הנחש הוא בצבע שחור והוא בעל גודל התחלתי של **שלוש חוליות** (שלושה תאים), וכיוון אליו הוא מתקדם בכל שלב ("Right","Left","Down","Up"). בתחילת המשחק ראשו של הנחש ממוקמם בתא (10,10) והוא מתקדם בכיוון מעלה ("Up").

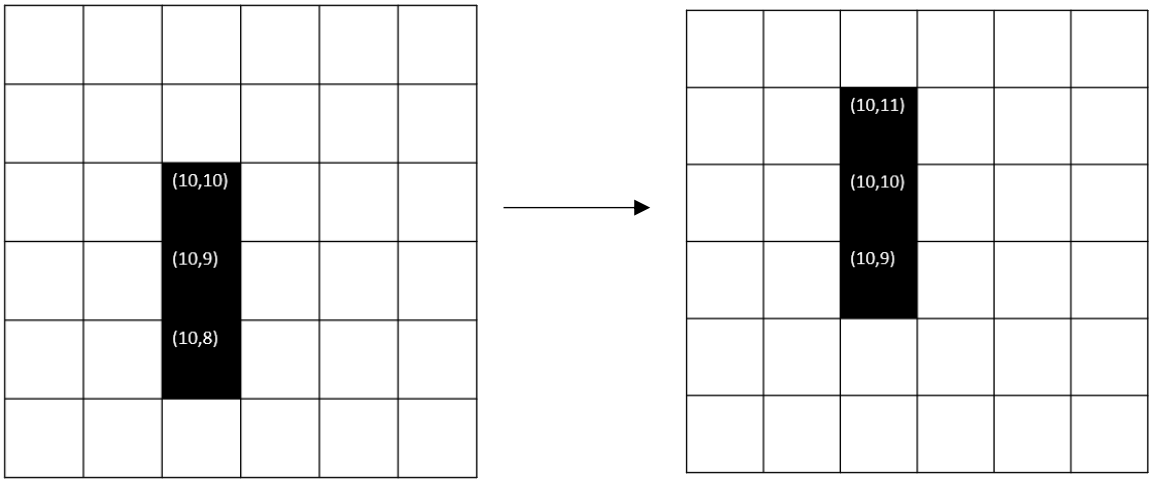
התקדמות הנחש:

בכל איטרציה של המשחק, נזיז את הנחש משבצת אחת **בכיוון** ההתקדמות שלו. נבצע זאת בשני צעדים:

1. נוסיף ראש חדש לנחש במיקום שנקבע על פי הכיוון הנוכחי של הנחש.
2. נסיר את החוליה האחרונה בגוף הנחש.

נראה דוגמה להתקדמות הנחש בשני כיוונים:

מקרה א: נחש בעל שלוש חוליות שמיקומו ההתחלתי (10,10) והכיוון שלו הוא **"Up"**. אזי הגוף של הנחש מוגדר על ידי המיקומים הבאים בסיבוב ה – i (בכל חוליה ישנו מיקום (x,y)):

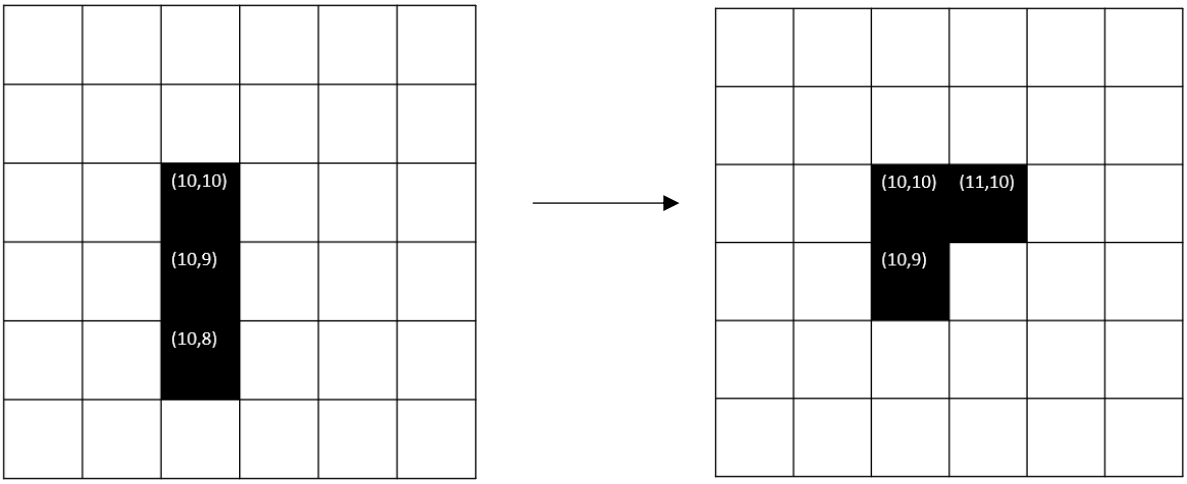


<u>תנועת הנחש מתעדכנת על ידי:</u>

א) הוספת ראש חדש במיקום (10,11) – מתקדמים בציר ה - y כי הכיוון הוא "Up".

ב) הסרה של החוליה האחרונה בגוף הנחש - (10,8).

מקרה ב: נחש בעל שלוש חוליות שמיקומו ההתחלתי (10,10) והכיוון שלו הוא **"Right"**. אזי הגוף של הנחש מוגדר על ידי המיקומים הבאים בסיבוב ה – i (בכל חוליה ישנו מיקום (x,y)) :



<u>תנועת הנחש מתעדכנת על ידי:</u>

א) הוספת ראש חדש במיקום (11,10) – מתקדמים בציר ה - x כי הכיוון הוא "Right".

ב) הסרה של החוליה האחרונה בזנב (10,8).

באופן סימטרי, אם הכיוון הוא "left", אז הראש זז מיקום אחד שמאלה בציר ה - x, ואם הכיוון הוא "down" אז הראש זז מיקום אחד מטה בציר ה – y. בשני המקרים הסימטריים אנו מסירים את החוליה האחרונה בגוף הנחש.

שינוי כיוון הנחש:

אם המשתמש לחץ על מקשי החיצים (שמאלה, ימינה, למעלה, למטה) עלינו לשנות את הכיוון של הנחש בהתאם. השתמשו במתודה **get_key_clicked** מהמחלקה **GameDisplay** על מנת לבדוק איזה מקש הוקלד. עליכם לקרוא לה **פעם אחת בסיבוב בלבד**.

נשים לב שאי אפשר לשנות את הכיוון של הנחש לכיוון ההופכי שלו בעדכון יחיד. למשל, אם הכיוון הנוכחי של הנחש הוא *"למעלה"*, והמשתמש לחץ *"חץ מטה"* אז שינוי הכיוון לא יתבצע. שינויי הכיוון היחידים שמותרים במקרה זה הם *"שמאלה או ימינה"*. באופן סימטרי, אם הנחש כרגע מתקדם *ימינה*, אין אפשרות לשנות את כיוונו *לשמאלה* בעדכון יחיד.

התנגשות עצמית, בקצוות המסך:

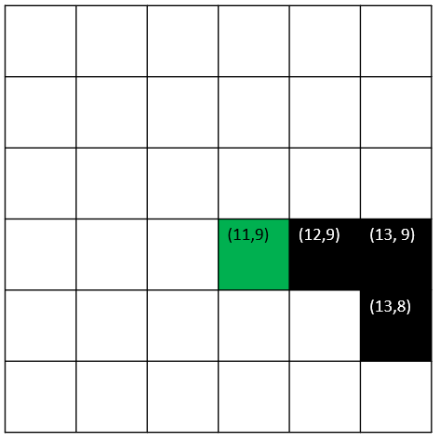
אם ראש הנחש מתנגש עם אחת מהחוליות שלו, או עם אחד מקצוות המסך, או פוגע בפצצה, או בגל ההדף שלה, המשחק מסתיים. כמו כן, במקרה של התנגשות עם הפצצה/גל ההדף, עליכם לצייר את הפצצה ולא את הנחש.

אכילת התפוח:

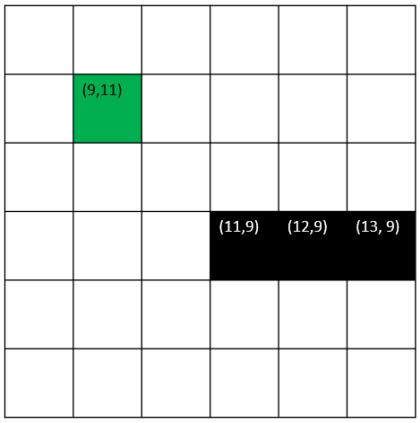
כשגוף הנחש **ממוקם על התפוח**, הנחש למעשה אוכל את התפוח. כתוצאה מכך, מספר החוליות בגוף שלו גדל ב-3. את הוספת החוליות החדשות נבצע על ידי הוספה של חוליית הראש במיקום חדש על פי כיוון התנועה של הנחש, אך

בשונה מהתיאור שתיארנו בהתקדמות הנחש, לא נסיר את החוליה האחרונה בגופו. התוצאה תהיה התארכות הגוף של הנחש ביחידה אחת בכל סיבוב. נראה דוגמה לאכילת התפוח והתקדמות הנחש.

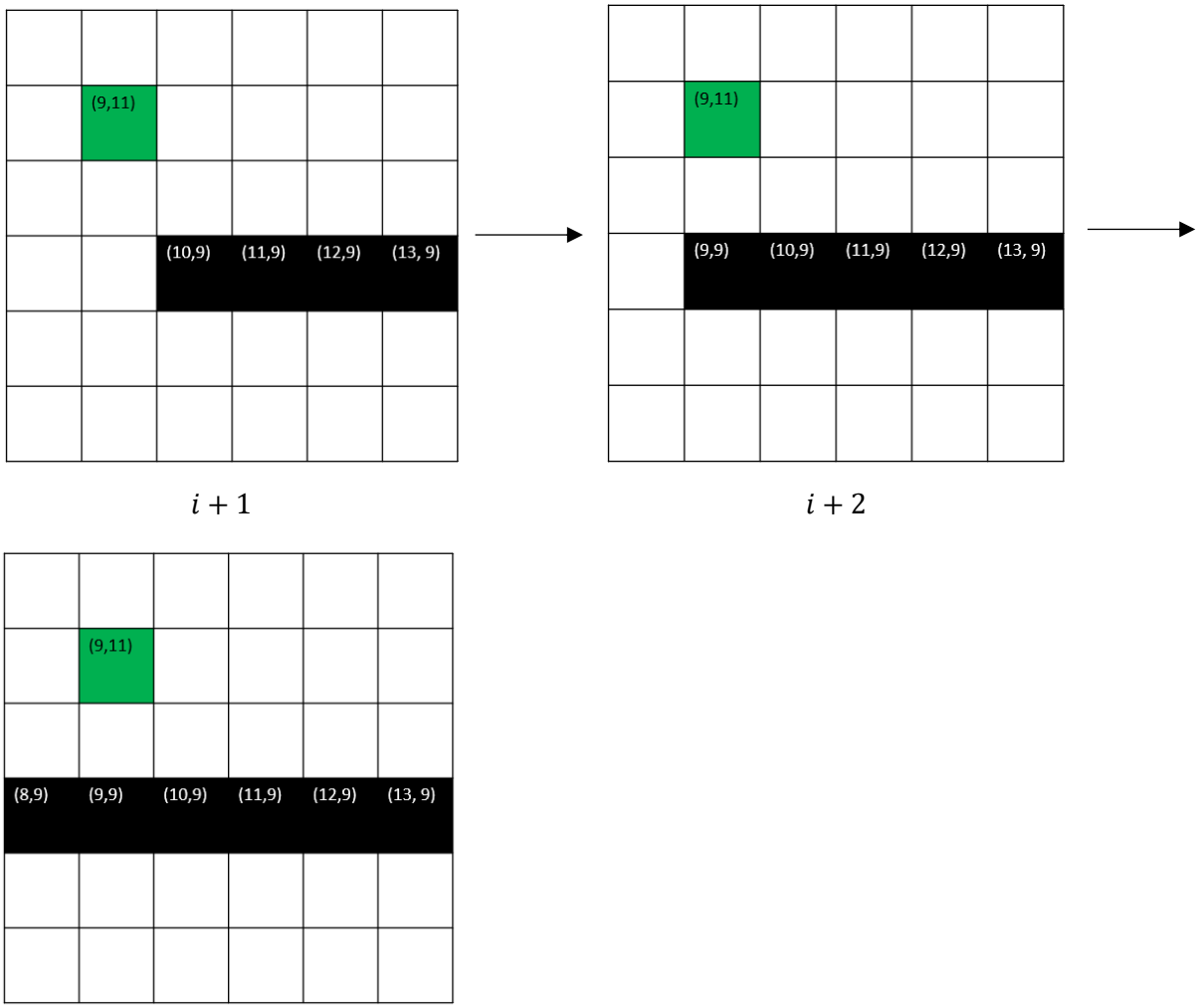
1.מצב הלוח בסיום הסיבוב ה- $(\boldsymbol{i}-\mathbf{1})$: גוף הנחש צבוע בשחור, כיוון ההתקדמות שלו הוא **"left"**, והתפוח צבוע בירוק.



2.מצב הלוח בסיבוב ה- $\boldsymbol{i}$: הנחש התקדם שמאלה, הראש שלו נמצא על מיקום התפוח באיטרציה ה - $\boldsymbol{i}-\mathbf{1}$, והתפוח נוצר מחדש וממוקם במיקום אחר.



3.מצב הלוח בסיום הסיבובים ה- $(\boldsymbol{i}+\mathbf{1}, \boldsymbol{i}+\mathbf{2}, \boldsymbol{i}+\mathbf{3})$, גופו של הנחש מתארך בחוליה אחת בכל פעם:



זכרו לעדכן את הניקוד במשחק בהתאם לניקוד אותו מספק התפוח.

<u>מקרי קצה:</u>

- נחש מתנגש בעצמו, או בפצצה, או בגל ההדף שלה במהלך תהליך ההתארכות:
  - במקרה כזה המשחק מסתיים
- הנחש אוכל יותר מתפוח אחד בסיבוב: אם במהלך האיטרציה ה - $\boldsymbol{i}$ אכלנו תפוח נוסף, נוסיף לניקוד את הניקוד של התפוח השני ונמקם את שני התפוחים מחדש על הלוח באותו סיבוב. כמו כן, עלינו להאריך את הנחש בעוד 3 חוליות בשביל התפוח השני ועוד 3 חוליות בשביל התפוח הראשון שאכלנו. סך הכל נאריך את הנחש עד האיטרציה ה - $\boldsymbol{i}+\mathbf{6}$. באופן דומה אם נאכל אם נאכל 3 תפוחים בסיבוב, הניקוד של שלושתם יצטבר, הם ימוקמו מחדש על הלוח ונאריך את הגוף שלו עד הסיבוב ה $\boldsymbol{i}+\mathbf{9}$.

- הנחש אוכל תפוח נוסף במהלך תהליך ההתארכות:
  - הנחש יקבל את הניקוד הנוסף של התפוח, התפוח ימוקם מחדש על הלוח באותו סיבוב, ושלושת החוליות החדשות יתווספו לגוף הנחש בסיום תהליך ההתארכות הנוכחי. לדוגמה, אם במהלך האיטרציה ה - $\boldsymbol{i+1}$ אכלנו תפוח נוסף, עלינו להאריך את הנחש בעוד 2 חוליות בשביל התפוח הקודם ועוד 3 חוליות בשביל התפוח הנוכחי. סך הכל נאריך את הנחש עד האיטרציה ה - $\boldsymbol{i+6}$.

סדר פעולות מומלץ

אנו ממליצים לעקוב אחר סדר הפעולות הבאות במימוש המשחק:

אתחול:

- מיקום הנחש על הלוח
- מיקום הפצצה על הלוח
- מיקום התפוחים על הלוח
- קריאה לפונקציה end_round מהמחלקה GameDisplay

בכל סיבוב:

- קריאת הקלט מהמשתמש
- הנחש מתקדם בכיוון שמוגדר עבורו
- בדיקה האם הנחש התנגש בעצמו, התנגש בפצצה, אכל תפוח ועדכון בהתאם (ניקוד, התארכות, סיום משחק)
- אם הפצצה התפוצצה – קידום גל ההדף שלה בלוח המשחק
- בדיקה אם הפצצה/גל ההדף פוגעת בתפוח, נחש ועדכון בהתאם – (סיום המשחק, או ציור פצצה חדשה)
- השלמת תפוחים חסרים
- מציירים את הלוח
- קריאה לפונקציה end_round מהמחלקה GameDisplay

סיום המשחק

המשחק מסתיים רק אם הנחש התנגש בעצמו, התנגש בקיר, או נפגע מהפצצה, או גל ההדף שלה. **אם הנחש נפגע מהפצצה או גל ההדף שלה, עליכם לצייר בסיבוב האחרון את הפצצה, ולא את הנחש**. בכל המקרים בהם המשחק מסתיים, עליכם לקרוא פעם אחרונה ל end_round לפני היציאה מהלולאה הראשית, ולצאת מהמשחק בצורה מסודרת ללא הדפסת כל הודעה וללא כל שגיאה.

<u>מתודות מהמחלקה **GameDisplay.py**:</u>

get_key_clicked(self) -> Optional[str]

מתודה זו מחזירה את אחד מהערכים **"Up", "Down", "Left", "Right"** אם נלחץ אחד ממקשי החיצים במקלדת. אם לא נלחץ אף מקש במהלך הסיבוב, ערך ההחזרה הוא **None**.

draw_cell(self, x: int, y: int, color: str) -> None

מתודה זו צובעת את התא במיקום (x,y) בצבע הנתון. ניתן להשתמש בצבעים "[**Black, green, red ,orange**]" ציור של תא בקואורדינטה הנמצאת מחוץ לגבולות הלוח תגרום **לשגיאה** ויש להימנע מכך.

end_round(self) -> None

מתודה זו מציינת את סוף הסיבוב הנוכחי. יש לקרוא לה לאחר הדפסת הלוח בסיבוב הנוכחי.

show_score(self, val: int) -> None

מתודה המעדכנת את הניקוד הנוכחי במשחק. יש לקרוא לה בכל פעם שנאכל תפוח ונרצה להציג את הניקוד המצטבר שצבר השחקן.

<u>פונקציות וקבועים מהקובץ **game_parameters.py**:</u>

במהלך בדיקת התרגיל נשנה את הקבועים ואת הערכים המוחזרים מהפונקציות בקובץ זה. אפשר להניח שהערכים המוחזרים מפונקציות אלו הינם חוקיים (הם בגבולות הלוח, אי שליליים וכו').

WIDTH, HEIGHT - הרוחב והגובה של הלוח בהתאמה (מספר התאים)

x, y, score = get_random_apple_data() -> Tuple[int, int, int]

פונקציה המגרילה את קאורדינטות התפוח, ואת הניקוד אותו יספק התפוח.

x, y, radius, time = get_random_bomb_data() -> Tuple[int, int, int, int]

פונקציה המגרילה את הקואורדינטות של הפצצה, את רדיוס הפיצוץ ואת משך הזמן שיעבור עד לפיצוץ.

שימו לב שגל ההדף של פצצה עשוי לחרוג מגבולות הלוח, ובמקרה זה אין לצייר מחוץ לגבולות הלוח.

הערות כלליות

1. על תרגיל זה יערכו ראיונות עם מדריכי המעבדה. בראיונות תתבקשו בין היתר להסביר את השיקולים שהיו לכם בעת כתיבת המחלקות השונות והממשקים ביניהן, את ההחלטות שקיבלתם באשר לקביעת המקום בקוד בו יופיעו הפעולות השונות, וכו'.
2. כחלק מהראיון ינתן ציון לסגנון התכנות שלכם (תיעוד, שמות משתמש, פונקציות ומתודות אלגנטיות וכד').

נהלי הגשה

תרגיל זה חובה להגיש בזוגות.

לפני ההגשה ותחת הלינק המיועד להגשה, עליכם לפתוח קבוצה ב-moodle. אחד השותפים ייצור את הקבוצה על ידי הזנת שם השותף השני (שימו לב להוסיף את השם בדיוק כפי שהוא מופיע ב-moodle, כולל אותיות גדולות וקטנות במקומות הנכונים). השותף (שלא יצר את הקבוצה) יוכל לראות שרשמו אותו על ידי כניסה לקישור ההגשה וצפייה בשם בן הזוג. כדאי לרשום את בן הזוג בשלב מוקדם (אין צורך להגיש את התרגיל בפועל על מנת להירשם

כזוג). כמו כן, וודאו כי הקבוצה אינה מכילה יותר משני שותפים.

עליכם להגיש קובץ נוסף בשם AUTHORS **(ללא כל סיומת).** קובץ זה יכיל שורה אחת ובו הלוגינים (CSE login) של שני הסטודנטים המגישים, מופרד ע"י פסיק. כך:

minniemouse,mickeymouse

ודאו כי הגשתם קובץ AUTHORS תקין! אי הגשה של קובץ AUTHORS תקין תגרור הורדה בציון.

עליכם להגיש את הקובץ ex10.zip (בלבד) בקישור ההגשה של תרגיל 10 דרך אתר הקורס על ידי לחיצה על "Upload file". אנו ממליצים להתחיל לעבוד על התרגיל בשלב מוקדם.

Ex10.zip צריך לכלול את הקבצים הבאים:

1. snake_main.py (קובץ ההרצה הראשי אותו סיפקנו לכם).
2. AUTHORS (כפי שתואר להלן).
3. קבצי פייתון אחרים בהם נכתב המימוש שלכם. אין להגיש את game_display.py או את game_parameters.py.

בהצלחה!