מכללת הדסה, החוג למדעי המחשב
תכנות מונחה עצמים ופיתוח משחקים
סמסטר ב', תשפ"ד

תרגיל 1

תאריך אחרון להגשה:

הנביאים – יום ד', 17/04/2024, בשעה 23:59
שטראוס גברים – יום ה', 11/04/2024, בשעה 23:59

מטרת התרגיל:

רענון וחזרה על העקרונות שנלמדו בסמסטר א', ובכללם תכנון ממשק, ירושה ופולימורפיזם וכן שימוש בכלים מהספרייה הסטנדרטית שנלמדו כולל מצביעים חכמים.

תיאור כללי:

בתרגיל זה נממש מחשבון צורות הניתן לתכנות. המשתמש יוכל לייצר צורות בסיסיות, להגדיל ולהקטין אותן ולהוסיף וליצור צורות נוספות מהצורות הקיימות על ידי הכפלה של צורה או הנחת צורות אחת מעל השנייה (ראו פירוט והדגמות בהמשך). בתרגיל זה נחזור לעבוד בטרמינל, לשם הפשטות.

פירוט הדרישות:

המחשבון מסוגל להחזיק רשימת צורות בסיסיות שעליהן הוא מסוגל לבצע מספר פעולות כפי שיוסבר בהמשך. שלוש הצורות הבסיסיות הן:

- Triangle – משולש (שווה צלעות)
- Rectangle – מלבן
- Square – ריבוע

מהלך ריצת התוכנית:

בכל פעם המחשבון מדפיס למסך את רשימת הצורות הנוכחית, מזכיר את הפקודה לקבלת עזרה, ומחכה לקריאת פקודה מהקלט. כאשר מתקבלת פקודה, המחשבון מבצע אותה ושוב מדפיס את רשימת הצורות העדכנית וכן הלאה.

לכל צורה ברשימה יש מספר סידורי (המספרים תמיד רציפים, גם אם נמחקו צורות; ראו להלן) שמודפס לידה בהדפסת רשימת הצורות, ובעזרתה ניתן להתייחס אליה בפקודות השונות.

רשימת הפקודות האפשריות:

הערה: הפקודות שיש להקליד הן בדרך כלל רק תחילת המילה, לשם הקיצור והנוחות, ובסוגריים מוצגת השלמת המילה כדי לעזור בהבנת המשמעות (יש להציג זאת כך גם בטקסט העזרה שהתוכנית צריכה להציג, כמודגם בהמשך).

cre(ate) shape x [y] – יוצרת אחת מהצורות הבסיסיות.

- הפרמטר shape יהיה t עבור משולש, r עבור מלבן או s עבור ריבוע.
- הפרמטרים x ו־y ישמשו לבניית הצורה (y אמור להופיע רק עבור המלבן, ומשמעותו הגובה של המלבן), וכמובן צריכים להיות חיוביים.

en(large) num n – מגדילה את צלעות הצורה שמספרה num, פי n פעמים (n צריך להיות מספר שלם בין 1 ל־10, כולל 10). שימו לב שההגדלה משפיעה גם על צורות אחרות המורכבות מהצורה הזו, הן תדפסנה כעת את הצורה המוגדלת!

red(uce) num n – מקטינה את צלעות הצורה שמספרה num, פי n פעמים (n צריך להיות מספר שלם בין 1 ל־10, כולל 10), למשל, red 1 3 מקטינה את צלעות הצורה שמספרה 1 בשליש (וגם כאן ההערה כמו בהגדלה לגבי ההשפעה על צורות שנוצרו מהצורה הזו).

draw num – מציירת את הצורה שמספרה num למסך.

dup(licate) num n – מוסיפה לרשימת הצורות צורה שהיא n פעמים הצורה שמספרה num (כמובן ש־n צריך להיות מספר שלם חיובי, ואם הוא 0 או שלילי תודפס שגיאה). בהדפסה ברשימת הצורות, נסמן כפל רגיל לפני הצורה (למשל, ((Triangle(3))*5, אם הכפלנו ב־5 משולש בגודל 3). בציור, צורה כזו תצוייר n פעמים אחת אחרי השנייה (במאונך).

stack num1 num2 – מוסיפה לרשימת הצורות צורה המורכבת מחיבור של שתי הצורות שמספרן num1 ו־num2 בצורה **אנכית** (נערום את הצורות אחת על השנייה). למשל, כך ניתן לייצר בית מחיבור של משולש ומלבן/ריבוע (למען הסר ספק: אין צורך לבדוק בקוד האם החיבור הגיוני/מתאים, זו החלטה של המשתמשים). בהדפסה ברשימה, נסמן בין הצורות / (כמו מונה ומכנה).

del(ete) num – מוחקת את הצורה שמספרה num מרשימת הצורות. שימו לב שאם יש צורות שנסמכות על הצורה הזו (צורות שיצרנו תוך שימוש בצורה הזו כאבן בניין), הצורות הללו עדיין צריכות להמשיך לעבוד כרגיל! מצד שני, מספרי הצורות שנשארו ברשימה צריכים להישאר רציפים, כמו שהוזכר לעיל.

help – מדפיסה מסך עזרה עם רשימת הפקודות האפשריות והסבר קצר עליהן.

exit – מדפיסה למסך "Goodbye" ויוצאת מהתוכנית.

דוגמה לריצת התוכנית:

הערה: אין צורך לעקוב בצורה מדויקת אחרי הניסוח כאן, אבל כן לעקוב אחרי הקו הכללי.

הקלט מהמשתמש מסומן ברקע צהוב. ההסברים בעברית נוספו כאן כדי להבהיר מה קורה, ואינם חלק מהפלט או הקלט של התוכנית עצמה.

הפלט ההתחלתי:

```
Shape list is empty
```

ניצור משולש באורך 4:

```
Enter command ('help' for the list of available commands): cre t 4

List of the available shapes:
0.      Triangle(4)
```

ונצייר אותו:

```
Enter command ('help' for the list of available commands): draw 0
   *
  * *
 *   *
* * * *

List of the available shapes:
0.      Triangle(4)
```

ניצור ריבוע באורך 4:

```
Enter command ('help' for the list of available commands): cre s 4

List of the available shapes:
0.      Triangle(4)
1.      Square(4)
```

ונצייר אותו:

```
Enter command ('help' for the list of available commands): draw 1
* * * *
*     *
*     *
* * * *

List of the available shapes:
0.      Triangle(4)
1.      Square(4)
```

נשים אותם אחד על השני:

```
Enter command ('help' for the list of available commands): stack 0 1

List of the available shapes:
0.      Triangle(4)
1.      Square(4)
2.      (Triangle(4)) / (Square(4))
```

ונצייר את התוצאה:

```
Enter command ('help' for the list of available commands): draw 2
   *
  * *
 *   *
* * * *
* * * *
*     *
*     *
* * * *

List of the available shapes:
0.      Triangle(4)
1.      Square(4)
2.      (Triangle(4)) / (Square(4))
```

נכפיל את הבית ב־3:

```
Enter command ('help' for the list of available commands): dup 2 3
List of the available shapes:
0.      Triangle(4)
1.      Square(4)
2.      (Triangle(4)) / (Square(4))
3.      3 * ((Triangle(4)) / (Square(4)))
```

ונצייר:

```
Enter command ('help' for the list of available commands): draw 3
   *
  * *
 *   *
* * * *
* * * *
*     *
*     *
* * * *
   *
  * *
 *   *
* * * *
* * * *
*     *
*     *
* * * *
   *
  * *
 *   *
* * * *
* * * *
*     *
*     *
* * * *

List of the available shapes:
```

```
0.      Triangle(4)
1.      Square(4)
2.      (Triangle(4)) / (Square(4))
3.      3 * ((Triangle(4)) / (Square(4)))
```

נקטין את הריבוע המקורי פי 2 (שימו לב להשפעה המיידית על ההדפסות של כל הצורות הקשורות):

```
Enter command ('help' for the list of available commands): red 1 2

List of the available shapes:
0.      Triangle(4)
1.      Square(2)
2.      (Triangle(4)) / (Square(2))
3.      3 * ((Triangle(4)) / (Square(2)))
```

ונצייר שוב את הבית:

```
Enter command ('help' for the list of available commands): draw 3
   *
  * *
 *   *
* * * *
* *
* *
   *
  * *
 *   *
* * * *
* *
* *
   *
  * *
 *   *
* * * *
* *
* *

List of the available shapes:
0.      Triangle(4)
1.      Square(2)
2.      (Triangle(4)) / (Square(2))
3.      3 * ((Triangle(4)) / (Square(2)))
```

נמחק את הריבוע (שימו לב שהאינדקסים ברשימת הצורות נשארים רצופים):

```
Enter command ('help' for the list of available commands): del 1

List of the available shapes:
0.      Triangle(4)
1.      (Triangle(4)) / (Square(2))
2.      3 * ((Triangle(4)) / (Square(2)))
```

ועדיין ניתן לצייר את הבית, הכול ממשיך לעבוד:

```
Enter command ('help' for the list of available commands): draw 2
   *
  * *
 *   *
* * * *
* *
* *
   *
  * *
 *   *
* * * *
* *
* *
   *
  * *
 *   *
* * * *
* *
* *

List of the available shapes:
0.      Triangle(4)
1.      (Triangle(4)) / (Square(2))
2.      3 * ((Triangle(4)) / (Square(2)))
```

אם נרצה לשחזר את הבית*3 לגודל המקורי, כבר אין איך לשנות את הריבוע שנמחק מהרשימה, אבל אפשר לשנות אותו עצמו (וזה משפיע גם על המשולש שבתוכו, אבל לא על המשולש המקורי ולא על הבית המקורי):

```
Enter command ('help' for the list of available commands): en 2 2

List of the available shapes:
0.      Triangle(4)
1.      (Triangle(4)) / (Square(2))
2.      3 * ((Triangle(8)) / (Square(4)))
```

ואז להקטין את המשולש המקורי (מה שמשפיע על כל מי שמשתמש בו):

```
Enter command ('help' for the list of available commands): red 0 2

List of the available shapes:
0.      Triangle(2)
1.      (Triangle(2)) / (Square(2))
2.      3 * ((Triangle(4)) / (Square(4)))
```

נצייר כדי לראות את התוצאה:

```
Enter command ('help' for the list of available commands): draw 2
   *
  * *
 *   *
* * * *
* * * *
*     *
*     *
* * * *
   *
  * *
 *   *
* * * *
* * * *
*     *
*     *
* * * *
   *
  * *
 *   *
* * * *
* * * *
*     *
*     *
* * * *

List of the available shapes:
0.      Triangle(2)
1.      (Triangle(2)) / (Square(2))
2.      3 * ((Triangle(4)) / (Square(4)))

Enter command ('help' for the list of available commands): help
The available commands are:
* cre(ate shape) <t - triangle | r - rectangle | s - square> x [y] -
create new shape according to the chosen letter, with the given
size(s) (y must be given only for a rectangle)
* en(large) num n - enlarge the size of the sides of shape #num by n
(1-10)
* red(uce) num n - reduce the size of the sides of shape #num by n
(1-10)
* draw num - draw shape #num
* dup(licate) num n - create a new shape which is a n times
(vertical) duplication of shape #num
* stack num1 num2 - create a new shape by stacking shape number #num1
over shape number #num2
* del(ete) num - delete shape #num from the shape list
* help - print this command list
* exit - exit the program

List of the available shapes:
```

```
0.      Triangle(2)
1.      (Triangle(2)) / (Square(2))
2.      3 * ((Triangle(4)) / (Square(4)))
```

ניצור מלבן (שימו לב לאופן ההדפסה, כדי להבדיל בין הרוחב והגובה):

```
Enter command ('help' for the list of available commands): cre r 4 7

List of the available shapes:
0.      Triangle(2)
1.      (Triangle(2)) / (Square(2))
2.      3 * ((Triangle(4)) / (Square(4)))
3.      Rectangle(w: 4, h: 7)
```

ונצייר אותו:

```
Enter command ('help' for the list of available commands): draw 3
* * * *
*     *
*     *
*     *
*     *
*     *
* * * *

List of the available shapes:
0.      Triangle(2)
1.      (Triangle(2)) / (Square(2))
2.      3 * ((Triangle(4)) / (Square(4)))
3.      Rectangle(w: 4, h: 7)
```

נשים את הבית*3 על המלבן:

```
Enter command ('help' for the list of available commands): stack 2 3

List of the available shapes:
0.      Triangle(2)
1.      (Triangle(2)) / (Square(2))
2.      3 * ((Triangle(4)) / (Square(4)))
3.      Rectangle(w: 4, h: 7)
4.      (3 * ((Triangle(4)) / (Square(4)))) / (Rectangle(w: 4, h: 7))
```

ונצייר:

```
Enter command ('help' for the list of available commands): draw 4
   *
  * *
 *   *
* * * *
* * * *
*     *
*     *
* * * *
   *
  * *
 *   *
* * * *
* * * *
*     *
*     *
* * * *
   *
  * *
 *   *
* * * *
* * * *
*     *
*     *
* * * *
* * * *
*     *
*     *
*     *
*     *
*     *
* * * *

List of the available shapes:
0.      Triangle(2)
1.      (Triangle(2)) / (Square(2))
2.      3 * ((Triangle(4)) / (Square(4)))
3.      Rectangle(w: 4, h: 7)
4.      (3 * ((Triangle(4)) / (Square(4)))) / (Rectangle(w: 4, h: 7))
```

ועכשיו להיפך, את המלבן על הבית*3:

```
Enter command ('help' for the list of available commands): stack 3 2

List of the available shapes:
0.      Triangle(2)
1.      (Triangle(2)) / (Square(2))
2.      3 * ((Triangle(4)) / (Square(4)))
3.      Rectangle(w: 4, h: 7)
4.      (3 * ((Triangle(4)) / (Square(4)))) / (Rectangle(w: 4, h: 7))
5.      (Rectangle(w: 4, h: 7)) / (3 * ((Triangle(4)) / (Square(4))))
```

ונצייר:

```
Enter command ('help' for the list of available commands): draw 5
* * * *
*     *
*     *
*     *
*     *
*     *
* * * *
   *
  * *
 *   *
* * * *
* * * *
*     *
*     *
* * * *
   *
  * *
 *   *
* * * *
* * * *
*     *
*     *
* * * *
   *
  * *
 *   *
* * * *
* * * *
*     *
*     *
* * * *

List of the available shapes:
0.      Triangle(2)
1.      (Triangle(2)) / (Square(2))
2.      3 * ((Triangle(4)) / (Square(4)))
3.      Rectangle(w: 4, h: 7)
4.      (3 * ((Triangle(4)) / (Square(4)))) / (Rectangle(w: 4, h: 7))
5.      (Rectangle(w: 4, h: 7)) / (3 * ((Triangle(4)) / (Square(4))))
```

נגדיל את הבית המקורי (כדי להתאים לגודל המלבן בפעולה הבאה):

```
Enter command ('help' for the list of available commands): en 1 2

List of the available shapes:
0.      Triangle(2)
1.      (Triangle(4)) / (Square(4))
2.      3 * ((Triangle(8)) / (Square(8)))
3.      Rectangle(w: 4, h: 7)
4.      (3 * ((Triangle(8)) / (Square(8)))) / (Rectangle(w: 4, h: 7))
5.      (Rectangle(w: 4, h: 7)) / (3 * ((Triangle(8)) / (Square(8))))
```

ונשים אותו על המלבן:

```
Enter command ('help' for the list of available commands): stack 1 3

List of the available shapes:
0.      Triangle(2)
1.      (Triangle(4)) / (Square(4))
2.      3 * ((Triangle(8)) / (Square(8)))
3.      Rectangle(w: 4, h: 7)
4.      (3 * ((Triangle(8)) / (Square(8)))) / (Rectangle(w: 4, h: 7))
5.      (Rectangle(w: 4, h: 7)) / (3 * ((Triangle(8)) / (Square(8))))
6.      ((Triangle(4)) / (Square(4))) / (Rectangle(w: 4, h: 7))
```

וקיבלנו בניין רב קומות :)

```
Enter command ('help' for the list of available commands): draw 6
   *
  * *
 *   *
* * * *
* * * *
*     *
*     *
* * * *
* * * *
*     *
*     *
*     *
*     *
*     *
* * * *

List of the available shapes:
0.      Triangle(2)
1.      (Triangle(4)) / (Square(4))
2.      3 * ((Triangle(8)) / (Square(8)))
3.      Rectangle(w: 4, h: 7)
4.      (3 * ((Triangle(8)) / (Square(8)))) / (Rectangle(w: 4, h: 7))
5.      (Rectangle(w: 4, h: 7)) / (3 * ((Triangle(8)) / (Square(8))))
6.      ((Triangle(4)) / (Square(4))) / (Rectangle(w: 4, h: 7))

Enter command ('help' for the list of available commands): exit
Goodbye!
```

הערות לגבי צורת המימוש:

- כדאי להשקיע זמן בתיכון התוכנית. עם תיכון נכון, הקוד יהיה די מינימלי ופשוט לכתיבה. בפרט, אל תחששו להשתמש בכלים של הספרייה הסטנדרטית כמו וקטור ומצביעים חכמים, הם יהפכו את הקוד להרבה יותר נקי, יעיל וקצר ויקלו על כתיבת קוד שיעבוד בצורה נכונה.

- המספרים לטובת ממדי הצורות הם double. כל שאר המספרים הם int.
- ניתן להניח שהקלט תקין מהבחינה שבמקום שצריך להגיע מספר אכן קיבלנו מספר (כלומר, זה בסדר לקרוא ישירות לתוך int או double, לפי הצורך). עדיין חובה לבדוק את שם הפקודה, סוג הצורה בפקודת היצירה ואת מספרי הפונקציות האם הם קיימים, וכן לוודא כל מגבלה אחרת שצוינה לעיל.
- כמו בדוגמה, שמרו על תא ריק בין כוכבית לכוכבית ברוחב, כדי שאפשר יהיה לצייר בקלות את המשולש.

קובץ ה־README

יש לכלול קובץ README שיקרא README.docx ,README.doc או README.txt (ולא בשם אחר). הקובץ יכול להיכתב בעברית ובלבד שיכיל את הסעיפים הנדרשים.

קובץ זה יכיל לכל הפחות:

1. כותרת.
2. פרטי הסטודנט: שם מלא כפי שהוא מופיע ברשימות המכללה, ת"ז.
3. הסבר כללי של התרגיל.
4. רשימה של הקבצים שיצרנו, עם הסבר קצר (לרוב לא יותר משורה או שתיים) לגבי תפקיד הקובץ.
5. מבני נתונים עיקריים ותפקידיהם.
6. אלגוריתמים הראויים לציון.
7. **תיכון (design): הסבר קצר מהם האובייקטים השונים בתוכנית, מה התפקיד של כל אחד מהם וחלוקת האחריות ביניהם ואיך מתבצעת האינטראקציה בין האובייקטים השונים.**
8. באגים ידועים.
9. הערות אחרות.

יש לתמצת ככל שניתן אך לא לוותר על אף חלק. אם אין מה להגיד בנושא מסוים יש להשאיר את הכותרת ומתחתיו פסקה ריקה. **נכתוב ב־README כל דבר שרצוי שהבודק ידע כשהוא בודק את התרגיל.**

אופן ההגשה

הקובץ להגשה: ניתן ליצור בקלות קובץ zip המותאם להגדרות ההגשה המפורטות להלן ישירות מתוך VS, כפי המוסבר תחת הכותרת "יצירת קובץ ZIP להגשה או לגיבוי" בקובץ "הנחיות לשימוש ב־Visual Studio 2022". אנא השתמשו בדרך זו (אחרי שהגדרתם כראוי את שמות הצוות ב־MY_AUTHORS: **נשים את שמות המגישים בתוך המרכאות, נקפיד להפריד בין השם הפרטי ושם המשפחה בעזרת קו תחתי ואם יש יותר ממגיש אחד, נפריד בין השמות השונים בעזרת מקף (מינוס '-')**) וכך תקבלו אוטומטית קובץ zip המותאם להוראות, בלי טעויות שיגררו אחר כך בעיות בבדיקה.

באופן כללי, הדרישה היא ליצור קובץ zip בשם oop2_exN-firstname_lastname.zip (או במקרה של הגשה בזוג – oop2_exN-firstname1_lastname1-firstname2_lastname2.zip), כשהקובץ כולל את כל קובצי הפרויקט, למעט תיקיות out ו־vs. כל הקבצים יהיו בתוך תיקייה ראשית אחת.

את הקובץ יש להעלות ל־Moodle של הקורס למשימה המתאימה. בכל מקרה, **רק אחד** מהמגישים יגיש את הקובץ ולא שניהם.

הגשה חוזרת: אם מסיבה כלשהי החלטתם להגיש הגשה חוזרת יש לוודא ששם הקובץ זהה לחלוטין לשם הקובץ המקורי. אחרת, אין הבודק אחראי לבדוק את הקובץ האחרון שיוגש.

כל שינוי ממה שמוגדר פה לגבי צורת ההגשה ומבנה ה־README עלול לגרור הורדת נקודות בציון.

מספר הערות:

1. נשים לב לשם הקובץ שאכן יכלול את שמות המגישים.
2. נשים לב לשלוח את תיקיית הפרויקט כולה, לא רק את קובצי הקוד שהוספנו. תרגיל שלא יכלול את כל הקבצים הנדרשים, לא יתקבל וידרוש הגשה חוזרת (עם כללי האיחור הרגילים).

המלצה כללית: אחרי הכנת הקובץ להגשה, נעתיק אותו לתיקייה חדשה, נחלץ את הקבצים שבתוכו ונבדוק אם ניתן לפתוח את התיקייה הזו ולקמפל את הקוד. הרבה טעויות של שכחת קבצים יכולות להימנע על ידי בדיקה כזו.

בהצלחה!