בית הספר להנדסה ומדעי המחשב ע"ש רחל וסלים בנין

מבוא למדעי המחשב 67101

תרגיל 7 - רקורסיה

להגשה בתאריך 09/12/2016 בשעה 22:00

בתרגיל זה נתרגל מבני רקורסיה שונים. התרגיל מורכב ממספר משימות בלתי תלויות.

שימו לב: בפתרון תרגיל זה אין לעשות שימוש באף מודול חיצוני של python – כלומר, אין לעשות import לאף מודול לצורך הפתרון! בפרט, אין לעשות שימוש במודול math או במודול itertools.

חלק ראשון: רקורסיה לינארית

את המשימות בחלק זה יש לפתור ע"י שימוש בפונקציות רקורסיביות, **ללא שימוש בלולאות מכל סוג שהוא (גם לא בעקיפין)!**

1. **הפונקציה** print_to_n(n)
עליכם לממש את הפונקציה print_to_n, המקבלת את המספר n (int), ומדפיסה את המספרים (השלמים) מ-1 עד n בסדר עולה.

2. **הפונקציה** print_reversed(n)
עליכם לממש את הפונקציה print_reversed_n, המקבלת את המספר n (int), ומדפיסה את המספרים (השלמים) מ-n עד 1 בסדר יורד.

3. **הפונקציה** is_prime(n)
עליכם לממש את הפונקציה is_prime, המקבלת את המספר n (int), ומחזירה True אם הוא ראשוני ו-False אחרת. מספר ראשוני הוא שלם גדול מ- 1, שהשלמים היחידים המחלקים אותו ללא שארית הם 1 והמספר עצמו.
רמז: תוכלו לממש ולהיעזר בפונקציה has_divisor_smaller_than(n, i), הבודקת האם ל- n מחלק (שונה מ- 1) קטן מ- i.
הערה: מכאן ואילך עליכם להבין בעצמכם האם נדרשות פונקציות עזר ואילו.

4. **הפונקציה** divisors(n)
עליכם לממש את הפונקציה divisors, המקבלת את המספר n (int), **ומחזירה רשימה** של כל השלמים (החיוביים) המחלקים אותו ללא שארית, לפי סדר עולה.
לדוגמא, על קריאה ל- divisors(12) להחזיר: [1, 2, 3, 4, 6, 12].

5. **הפונקציה** exp_n_x(n, x)
עליכם לממש את הפונקציה exp_n_x, המקבלת את המספר n (int), והמספר x (real), ומחזירה את $exp_n(x)$ – פונקציית הסכום האקספוננציאלי, כאשר:
$$exp_n(x) = \sum_{i=0}^{n} \frac{x^i}{i!} \approx e^x$$
תוכלו להניח כי n אי שלילי.

חלק שני: רקורסיה לא לינארית

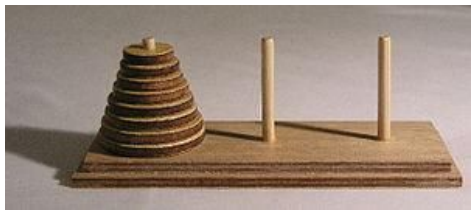
(לצורך פתרון המשימות בחלק זה, תוכלו להיעזר גם בלולאות)

6. **הפונקציה** play_hanoi(hanoi, n, src, dest, temp)

עליכם לממש את הפונקציה play_hanoi, הפותרת משחק "מגדלי הנוי".
משחק "מגדלי הנוי" כולל:

- שלושה מוטות אנכיים ("המגדלים").
- מספר דיסקיות בגדלים שונים שניתן להשחיל על המוטות, כאשר כל דיסקית - בגודל שונה.

בתחילת המשחק, הדיסקיות מסודרות על פי גודלן על אחד המוטות, כשהגדולה ביותר למטה והקטנה ביותר למעלה. מטרת המשחק היא להעביר את כל הדיסקיות ממוט זה אל אחד משני המוטות הנותרים, בכפוף לשני חוקים:



- מותר להזיז רק דיסקית אחת בכל פעם - מראש מוט אחד לראש מוט אחר.
- אסור להניח דיסקית אחת על דיסקית שקטנה ממנה.

לקריאה והסברים נוספים:
https://he.wikipedia.org/wiki/%D7%9E%D7%92%D7%93%D7%9C%D7%99_%D7%94%D7%90%D7%A0%D7%95%D7%99

השימוש בפונקציה נעשה על ידי הרצת הקובץ hanoi_game.py באותה תיקייה שבה נמצא הקובץ ex7.py אותו אתם ממשים. על מנת שהפונקצייה אותה אתם כותבים תבצע שינויים במשחק הגרפי, הפונקציה נקראת עם הפרמטרים הבאים:
hanoi – אובייקט מורכב שהוא המשחק הגרפי בו מתבצע השינוי.
n – מספר (int) הדיסקיות אותן על הפונקציה להעביר.
src – אובייקט מורכב המייצג המוט ממנו מעוניינים להעביר את הדיסקיות.
dest – אובייקט מורכב המייצג את המוט אליו מעוניינים להעביר את הדיסקיות.
temp – אובייקט מורכב המייצג את המוט השלישי במשחק.

שימו לב: האובייקטים המורכבים ניתנים לכם בקריאה המקורית לפונקצייה play_hanoi שמתבצעת בקובץ hanoi_game.py.

על מנת להעביר דיסקית במשחק hanoi ממוט למוט, יש להשתמש בפקודה:
hanoi.move(src, dest)
כאשר שני הפרמטרים הם מוטות במשחק. קריאה לפקודה זו תעביר את הדיסקית העליונה מהמוט src לראש המוט dest.
תוכלו להניח כי בזמן הקריאה הראשונית לפונקצייה מצב המשחק תקין (כלומר, ישנן בדיוק n דסקיות על המוט src מסודרות בצורה חוקית, ואין דסקיות על שאר המוטות). **לא ניתן להניח דבר על מימוש האובייקטים המורכבים.**

שימו לב #2: הקובץ hanoi_game.py יקרא לפונקציה שלכם רק עם ערכי n **חיוביים**. **אך על הפונקציה שלכם להתמודד עם כל ערך שלם!** עבור n שלילי, על הפונקציה להתנהג כאילו התקבל 0.

7. **הפונקציה** print_binary_sequences(n)
עליכם לממש את הפונקציה print_binary_sequences, המדפיסה את כל הצירופים האפשרים של "0" ו- "1" באורך n. שימו לב כי המחרוזת הריקה "" הינה צירוף (יחיד) של "0" ו- "1" באורך 0!

רמז: היעזרו בפונקציה print_binary_sequences_with_prefix(prefix, n) המדפיסה את כל הצירופים של "0" ו- "1" באורך n, שמתחילים ב prefix.
תוכלו להניח כי n אי שלילי.

8. **הפונקציה** print_sequences(char_list, n)
עליכם לממש את הפונקציה print_sequences, המקבלת רשימה של תווים char_list, ומדפיסה את כל הצירופים האפשריים באורך n של תווים מהרשימה, כאשר אותו תו יכול להופיע יותר מפעם אחת.
תוכלו להניח כי מתקבלת רשימה חוקית ולא ריקה של תווים (char) שונים אחד מהשני, וכן כי n אי שלילי.

9. **הפונקציה** print_no_repetition_sequences(char_list, n)
עליכם לממש את הפונקציה print_no_repetition_sequences, המקבלת רשימה של תווים char_list, ומדפיסה את כל הצירופים האפשריים באורך n של תווים מהרשימה, **ללא חזרות** (כלומר,אותו תו **לא** יכול להופיע יותר מפעם אחת).
תוכלו להניח כי מתקבלת רשימה חוקית ולא ריקה של תווים (char) שונים אחד מהשני, כי len(char_list)≥n, וכן כי n אי שלילי.

10. **הפונקציה** no_repetition_sequences_list(char_list, n)
עליכם לממש את הפונקציה no_repetition_sequences_list, המקבלת רשימה של תווים char_list, ומחזירה **רשימה** של **מחרוזות** כל הצירופים האפשריים באורך n של תווים מהרשימה, **ללא חזרות**.
לדוגמא, על קריאה ל-(no_repetition_sequences_list(['i', 'j', 'k'], 2 להחזיר:
['ij', 'ik', 'ji', 'jk', 'ki', 'kj'] (או כל רשימה בעלת אותם האיברים בסדר שונה).
תוכלו להניח כי מתקבלת רשימה חוקית ולא ריקה של תווים (char) שונים אחד מהשני, כי len(char_list)≥n, וכן כי n אי שלילי.

נהלי הגשה

הלינק להגשה של התרגיל הוא תחת השם: ex7

בתרגיל זה עליכם להגיש את הקבצים הבאים:

1. ex7.py – עם המימושים שלכם לפונקציות.
2. README (על פי פורמט ה-README לדוגמא שיש באתר הקורס, ועל פי ההנחיות לכתיבת README המפורטות בקובץ נהלי הקורס).

יש להגיש קובץ zip הנקרא ex7.zip המכיל בדיוק את שני הקבצים הנ"ל.

בהצלחה!