מבוא לתכנות 52304

תרגיל K - רקורסיה

להגשה בתאריך XX בשעה YY

בתרגיל זה נתרגל מבני רקורסיה שונים. התרגיל מורכב ממספר משימות בלתי תלויות. ניתן להניח תקינות הקלט בכל המשימות.

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 factorial(n)
עליכם לממש את הפונקציה המקבלת את המספר n (int), ומחזירה את הערך של עצרת !n. להזכירכם, $n! = n * (n-1) * \ldots * 1$. הקלט n יכול לקבל ערכים של 0,1,2,... (כאשר: 0!=1).

2. הפונקציה is_palindrome(lst)
באחד התרגילים הקודמים מימשנו את הפונקציה is_palindrome בעזרת לולאה. כעת עליכם לממש זאת בעזרת רקורסיה. על הפונקציה להחזיר True אם הרשימה היא palindrome ו-False אחרת.

3. הפונקציה sum_of_digits(n)
עליכם לממש את הפונקציה המקבלת מספר אי-שלילי n (int), ומחזירה את סכום ספרותיו. למשל, אם n=123, הפונקציה תחזיר 1+2+3=6.

חלק שני: רקורסיה לא לינארית

לצורך פתרון המשימות בחלק זה, תוכלו להיעזר גם בלולאות. עם זאת, הפתרון צריך להיות רקורסיבי במהותו, גם אם ניתן לממש את הפונקציה ללא רקורסיה.

4. הפונקציה(convert_binary(n
עליכם לממש את הפונקציה המקבלת מספר שלם אי-שלילי n (int), ומדפיסה את הייצוג הבינארי שלו. לדוגמא, עבור n=13, הפונקציה תדפיס 1101. ראו הסבר לגבי ההמרה בקישור הבא:
https://en.wikipedia.org/wiki/Binary_number#Decimal

5. הפונקציה nested_list_sum(lst)
עליכם לממש את הפונקציה המקבלת רשימה מקוננת (ייתכן כי איברים ברשימה גם הם רשימות שבהן איברים גם הן רשימות וכן הלאה) ומחזירה את סכום כל האיברים שבכל הרשימות. לדוגמא, אם הרשימה המקורית היא [5, [[4, 3], 2, [1, 0]] הפונקציה תחזיר 15. ניתן להניח שהקלט תקין: הרשימות מכילות רק מספרים, וקיים לפחות מספר אחד בקלט.

6. הפונקציה coin_pick_winner(n)

שני שחקנים משחקים משחק בו יש על השולחן n מטבעות. בכל תור, לשחקן מותר לקחת 1, 2 או 4 מטבעות. השחקן אשר משאיר את השולחן ללא מטבעות מנצח. הניחו ששני השחקנים חכמים ואינם עושים טעויות.
לדוגמא, אם n=2, השחקן הראשון ירים 2 מטבעות וינצח. לעומת זאת, אם n=3, לא משנה מה השחקן הראשון יעשה (ירים מטבע בודד או שניים), השחקן השני ינצח. עליכם לממש את הפונקציה המחזירה tuple בו האיבר הראשון הוא True אם השחקן שמתחיל ינצח ו False אחרת. האיבר השני הוא מספר האפשרויות השונות של השחקן בתור הראשון שיבטיחו
לו ניצחון. אם אין אפשרות לשחקן לנצח, אז האיבר השני יציין את מספר האפשרויות השונות בתור הראשון שיבטיחו לו הפסד.

עבור הדוגמא n=3, הפונקציה תחזיר (False, 2) עבור שתי האפשרויות השונות שבתור הראשון מכיוון שמפסידים בשתיהן.

עבור n=4, הפונקציה תחזיר (True, 2) שכן השחקן הראשון יכול להרים 4 וכך לנצח אבל יכול גם להרים מטבע 1 ועדיין לנצח בהמשך.
לעומת זאת עם ירים 2 מטבעות השחקן השני ינצח.

עבור n=5, הפונקציה תחזיר (True, 1) שכן רק אם ירים השחקן 2 מטבעות יגרום לשחקן השני להפסיד

לתשומת לבכם: לא כדאי לנסות את הפונקציה עבור n גדולים (מעל 30) מדי שכן ייתכן והריצה תהיה איטית ו/או תצרוך הרבה זכרון. עם זאת, צריך לכתוב קוד כללי אשר ללא מגבלת משאבים פותר את הבעיה לכל n.

נהלי הגשה

עליכם להגיש את הקובץ ex6.zip (בלבד) בקישור ההגשה של תרגיל 6 דרך אתר הקורס על ידי לחיצה על "Upload file".

ex6.zip צריך לכלול את הקבצים:

1. ex6.py
2. README (כמפורט בנהלי הקורס)

הנחיות כלליות בנוגע להגשה

- הנכם רשאים להגיש תרגילים דרך מערכת ההגשות באתר הקורס מספר רב של פעמים. ההגשה האחרונה בלבד היא זו שקובעת ושתיבדק.
- לאחר הגשת התרגיל, ניתן ומומלץ להוריד את התרגיל המוגש ולוודא כי הקבצים המוגשים הם אלו שהתכוונתם להגיש וכי הקוד עובד על פי ציפיותיכם.
- ניתן לאחר בהגשת התרגיל עד שלושה ימים, אך זה יהיה כרוך בניכוי ציון, כפי שמפורט באתר הקורס ובמסמך נהלי הקורס.

- קראו היטב את קובץ נהלי הקורס לגבי הנחיות נוספות להגשת התרגילים.
- שימו לב - יש להגיש את התרגילים בזמן!

בהצלחה!