תרגיל בית 6

הנחיות כלליות:

- קראו בעיון את השאלות והקפידו שהתכניות שלכם פועלות בהתאם לנדרש.
- את התרגיל יש לפתור לבד!
- הקפידו על כללי ההגשה המפורסמים באתר. בפרט, יש להגיש את כל הפתרונות לשאלות יחד בקובץ ex6_012345678.py המצורף לתרגיל, לאחר החלפת הספרות 012345678 במספר ת.ז. שלכם, כל 9 הספרות כולל ספרת ביקורת.
- מועד אחרון להגשה: כמפורסם באתר.
- בדיקה עצמית: כדי לוודא את נכונותן ואת עמידותן של התוכניות לקלטים שגויים, בכל שאלה הריצו את תוכניתכם עם מגוון קלטים שונים, אלה שהופיעו כדוגמאות בתרגיל וקלטים נוספים עליהם חשבתם (וודאו כי הפלט נכון).
- אין לשנות את שמות הפונקציות והמשתנים שכבר מופיעים בקובץ השלד של התרגיל.
- היות ובדיקת התרגילים עשויה להיות אוטומטית, **יש להקפיד על פלטים מדויקים על פי הדוגמאות (עד לרמת הרווח).**
- אופן ביצוע התרגיל: שימו לב, בתרגיל זה עליכם להשלים את הקוד בקובץ המצורף.

שאלה 1

כתוב פונקציה רקורסיבית reverse_string אשר מקבלת כקלט מחרוזת ומחזירה את המחרוזת בהיפוך סדר האותיות. אין להשתמש בפונקציות מובנות של פייתון להיפוך מחרוזות.

דוגמאות הרצה:

```
>>> reverse_string("abc")
'cba'
>>> reverse_string("Hello!")
'!olleH'
```

ניתן להניח ש-s מטיפוס str.

(המשך בעמוד הבא)

שאלה 2

בתרגול ראינו את הפונקציה sublist_sum אשר מקבלת כקלט רשימת מספרים numbers ומספר target ומחזירה True אם קיימת תת רשימה ב-numbers אשר סכומה שווה ל-target, ואחרת False.

ייתכן שהרשימה numbers כוללת מספר תתי רשימות שסכום כל אחת מהן היא target. כתוב פונקציה רקורסיבית min_sublist_sum אשר מקבלת רשימת מספרים ומחזירה את מספר האיברים ברשימה שאורכה מינימלי מבין הרשימות שסכומן target. אם לא קיימת ברשימה תת רשימה שסכומה target, הפונקציה תחזיר את הערך inf אשר ערכו **אינסוף** (כלומר, לכל ערך מספרי y מתקיים $\inf \geq y$). הערך inf מתקבל באמצעות הפקודה float("inf") (הפעלת הפונקציה float() על המחרוזת "inf"). לביטוי זה יש **ערך מספרי** (זו איננה מחרוזת):

```
>>> x = float("inf")
>>> type(x)
<type 'float'>
```

דוגמאות הרצה:

```
>>> min_sublist_sum([0,1,2,1], 2)
1
>>> min_sublist_sum([0,1,1], 2)
2
>>> min_sublist_sum([0,1,1], 3)
inf
```

ניתן להניח ש-numbers הינה רשימה של מספרים אי-שליליים (גדולים או שווים ל-0) וש-target הינו מספר אי-שלילי.

שאלה 3

שולה המוקשים (Minesweeper) הינו משחק מחשב שבו ניתן לוח (טבלה) ובתאים מסויימים חבויים מוקשים. בכל תא בלוח שאין בו מוקש, כתוב כמה מוקשים יש מסביבו. בתחילת המשחק כל תאי הלוח מוסתרים. השחקן בכל תור בוחר תא אחד בלוח, אשר חושף את מה שמאחוריו – מוקש או מספר.

בשאלה זו נדמה תור אחד במשחק עם לוח חד-מימדי המיוצג על ידי רשימה. נתונים לוח המשחק ms_board בתור רשימה המכילה את הערכים '*',0,1,2, ולוח ms_revealed כרשימה באורך זהה לשל ms_board, אשר מציין אילו תאים מוסתרים (False) ואלו תאים כבר נחשפו (True). בתחילת המשחק, כל התאים בלוח מוסתרים, ולכן כל איברי ms_board הם False.
למשל, בלוח המשחק הבא ישנם שלושה מוקשים (באינדקסים 0, 6 ו-8). באינדקסים 1, 5 ו-9 מופיע המספר 1 מכיוון שנמצא בצמוד לכל אחד מהם רק מוקש אחד, ובאינדקס 7 מופיע המספר 2 מכיוון שהוא נמצא בין שני מוקשים.

```
ms_board = ["*", 1, 0, 0, 0, 1, "*", 2, "*", 1, 0, 0, 0, 0]
```

בתחילת המשחק כל תאי הלוח מוסתרים ולכן ms_revealed מכיל רק False.

```
ms_revealed = [False]*len(ms_board)
```

(המשך בעמוד הבא)

כתוב פונקציה רקורסיבית uncover_cell אשר מקבלת את שני הלוחות (כמצויין בשלד ובדוגמא) ומספר שלם idx, שהוא האינדקס שהשחקן רוצה לחשוף בלוח המשחק בתור בודד מסויים, ומחזירה את ייצוג התאים החשופים לאחר המהלך, כלומר את ms_revealed.
אם תחת האינדקס idx מסתתר מוקש ('*') הפונקציה תחשוף את התא ותדפיס למסך 'boom!'.
אם תחת האינדקס idx מסתתר ערך גדול מ-0, הפונקציה תחשוף את התא ולא תדפיס כלום.
אם תחת האינדקס idx מסתתר 0, משמעות הדבר שבשני התאים שלצידו אין מוקש, ולכן הפונקציה תתנהג בהתאם להוראות לעיל על כל אחד מהם.
בכל מקרה, הפונקציה מחזירה את ms_revealed לאחר החשיפה.

למשל, נאתחל את ms_board ואת ms_revealed כנתון לעיל, ונבצע את הפקודות בהמשך (שים לב שיש רצף בין הקריאות – כולן מתבצעות אחת אחרי השנייה):

```
>>> ms_board = ["*", 1, 0, 0, 0, 1, "*", 2, "*", 1, 0, 0, 0, 0]
>>> ms_revealed = [False]*len(ms_board)

>>> ms_revealed1 = uncover_cell(ms_board, ms_revealed, 1)
>>> ms_revealed1
[False, True, False, False, False, False, False, False, False, False,
False, False, False, False]

>>> ms_revealed2 = uncover_cell(ms_board, ms_revealed1, 2)
>>> ms_revealed2
[False, True, True, True, True, True, False, False, False, False,
False, False, False, False]

>>> ms_revealed3 = uncover_cell(ms_board, ms_revealed2, 11)
>>> ms_revealed3
[False, True, True, True, True, True, False, False, False, True, True,
True, True, True]

>>> ms_revealed4 = uncover_cell(ms_board, ms_revealed3, 6)
boom!
>>> ms_revealed4
[False, True, True, True, True, True, True, False, False, True, True,
True, True, True]
```

ניתן להניח את נכונות הלוחות. ניתן להניח ש-idx הוא מספר שלם. אין להניח הנחות נוספות על idx.

(המשך בעמוד הבא)

שאלה 4

מחרוזת תקינה של סוגריים היא מחרוזת שבה לכל סוגר שמאלי ")" קיים סוגר ימיני "(" בהמשך המחרוזת. למשל, המחרוזת "(()()())" תקינה ואילו המחרוזות הללו אינן תקינות: "(" , ")" , ")(" .

כתוב פונקציה רקורסיבית is_valid_paren שתקבל מחרוזת המורכבת מתווים שונים (לא רק סוגריים) ותחזיר True אם המחרוזת תקינה מבחינת הסוגריים ואחרת False. הפונקציה תקבל ערך נוסף, cnt, אשר יעזור במניית הסוגריים השמאליים והימנים (ראה הדרכה בהמשך).

חתימת הפונקציה כפי שכתובה בקובץ השלד ולהלן, משמעותה שהערך הדיפולטי של cnt הוא 0. כלומר, אם לא קוראים לפונקציה עם ערך מפורש ל-cnt, הוא יקבל 0.

```
def is_valid_paren(s, cnt=0):
```

דוגמאות הרצה:

```
>>> is_valid_paren("(.(a)")
False
>>> is_valid_paren("p(()r((0)))")
True
```

הקריאה האחרונה שקולה לקריאה:

```
is_valid_paren("p(()r((0)))", 0)
```

<u>הדרכה</u>: אם אנחנו עוברים על המחרוזת מהסוף להתחלה (ימין לשמאל), ברגע שפגשנו '(', חייב לבוא משמאלו ')' בהמשך. הבעיה מתעוררת כאשר ראינו ')' שלא היה לפניו '('. כיצד ננצל את cnt כדי לעקוב אחר מספר הסוגריים שנפתחו וטרם נסגרו?