

מטלת מנחה (ממ"ן) 11

הקורס: תכנות וניתוח נתונים בשפת פייתון (20606)

נושאי המטלה: יסודות השפה, תנאים,

חומר הלימוד למטלה: יחידות 1-4

לולאות ופונקציות

משקל המטלה: 3 נקודות

מספר השאלות: 4

מועד אחרון להגשה: 22.11.2025

סמסטר: 2026א

(ת)

שימו לב:

- יש להקפיד על שמות המחלקות בדיוק כמו שנכתבו.
- יש לתעד את התכניות בתיעוד פנימי באנגלית בלבד (בתחילת התכנית התיעוד מסביר מה מבצעת התכנית באופן כללי ובמהלך התכניות התיעוד מסביר את הקוד) על פי תקן [PEP 8](#) וכפי שמודגם בדוגמאות ביחידה 1.9 באתר הקורס.
- אין להשתמש בחומר מתקדם ובפרט במבני נתונים מתקדמים ובפרט רשימות.
- יש להשתמש בקבועים היכן שאפשר.
- יש להקפיד על הזחה (אינדנטציה - עימוד) נכונה, ועל שמות משתנים בעלי משמעות (באנגלית) ולפי המוסכמות בקורס.
- יש להקפיד על פורמט הפלט בדיוק כפי שמצוין בשאלה: איות נכון, אותיות גדולות וקטנות, רווחים, וכו'.
- יש טסטר באתר הקורס, הכולל קטע קוד שאתם מעתיקים לתוך קובץ המטלה שלכם, ואז מריצים כדי שהפונקציות שלכם יופעלו ותוכלו לוודא שהכל עובד כמצופה. שימו לב שלא מגישים עם קוד הטסטר, אלא מוחקים את הקוד הזה אחרי ההרצה לבדיקה.
- מטלה שלא תעבור את הטסטר הבסיסי שספקנו לכם, תקבל ציון 0.
- הגשת המטלה נעשית אך ורק בעזרת מערכת המטלות המקוונת שבאתר הקורס.
- אל תשכחו לשמור את מספר האסמכתא שתקבלו מהמערכת לאחר ההגשה.

שאלה 1 (25 נק')

א. כתבו פונקציה בשם `calc` המקבלת שני מספרים שלמים: `num1`, `num2` ופעולה חשבונית

(חיבור/חיסור/כפל/חילוק) `op` מטיפוס **מחרוזת**. הפונקציה תחזיר את תוצאת הפעולה.

במקרה של פעולת חילוק, יש למנוע חילוק ב-0 ולהחזיר `None`.

במקרה והפעולה `op` אינה מייצגת פעולה חוקית, יש להחזיר `"invalid"`.

כותרת הפונקציה היא:

```
def calc(num1, num2, op)
```

דוגמאות:

- עבור הפרמטרים (משמאל לימין): 4, 10 ופעולת חילוק ('/') יוחזר:

~~10/4=2.5~~

- עבור הפרמטרים (משמאל לימין): 3, 5 ופעולת כפל ('*') יוחזר:

~~5*3=15~~

- עבור הפרמטרים (משמאל לימין): 0, 12 ופעולת חילוק ('/'):

None

ניתן להניח שהפרמטרים `num1`, `num2`, מייצגים מספרים שלמים (חיובי, שלילי או אפס) **והפעולה החשבונית מייצג מחרוזת והוא אחד מהתווים: +, -, *, /. אין צורך לבדוק זאת.**

ב. כתבו פונקציה בשם `compute_calcs` המקבלת מספר שלם חיובי `n`. הפונקציה תקלוט

מהמשתמש שלשות כמספר ערכו של n: זוג מספרים שלמים ומחרוזת המייצגת פעולה חשבונית (חיבור/חיסור/כפל/חילוק). עבור כל שלשה כזו, יש להדפיס תרגיל המייצג שלשה זו (כפי שמוצג בדוגמאות לעיל).

במקרה של חלוקה ב-0 יודפס: `"Can't divide by zero"`.

במקרה של פעולה לא חוקית (כלומר פעולה שאינם תואמת לאחת הפעולות: +, -, *, /), יש להדפיס `"The requested calculation is invalid"`.

ניתן להניח שהפרמטר `n` מייצג מספר שלם חיובי. אין צורך לבדוק זאת.

כותרת הפונקציה היא:

```
def compute_calcs(n)
```

המשך השאלה בעמוד הבא

דוגמה :

עבור $n=4$, ניתן להציג את הפלט האפשרי הבא :

```
Please insert the first number: 4
Please insert the second number: 5
Please insert an operator: \
The requested calculation is invalid
Please insert the first number: 5
Please insert the second number: 6
Please insert an operator: /
5 / 6 = 0.8333333333333334
Please insert the first number: 37
Please insert the second number: 6
Please insert an operator: *
37 * 6 = 222
Please insert the first number: 8
Please insert the second number: 0
Please insert an operator: /
Can't divide by zero
```

שימו לב!

- חובה להשתמש בפונקציה שכתבתם בסעיף א.
- הקפידו על רווחים במקומות הרלוונטיים (לפני ואחרי הפעולה החשבונית וסימן (=).
- במקרה של פעולת חילוק אין צורך ליפות את ההדפסה (למשל, במקרה של תרגיל $10 / 3$, התוצאה היא 3.3333333333333335).

שאלה 2 (25 נק')

א. מספר ארמסטרונג (¹Armstrong Number) הוא מספר שלם בו סכום ספרותיו בחזקת מספר ספרותיו שווה לערך המספר המקורי.

דוגמאות:

$$153 = 1^3 + 5^3 + 3^3 = 1 + 125 + 27 = 153$$

$$407 = 4^3 + 0^3 + 7^3 = 64 + 0 + 343 = 407$$

$$8208 = 8^4 + 2^4 + 0^4 + 8^4 = 4096 + 16 + 0 + 4096 = 8208$$

כתבו פונקציה בוליאנית בשם `is_armstrong` המקבלת מספר שלם `num`. הפונקציה תבדוק האם המספר שהתקבל כפרמטר מהווה מספר ארמסטרונג. אם כן, יוחזר `True`. אחרת, יוחזר `False`.

כותרת הפונקציה היא:

```
def is_armstrong(num)
```

דוגמאות:

- עבור הפרמטר 153 יוחזר `True`.
- עבור הפרמטר 1634 יוחזר `False` משום שהמספר שהתקבל אינו חיובי.
- עבור הפרמטר 4321 יוחזר `False` משום שהמספר שהתקבל אינו מהווה מספר ארמסטרונג.
- עבור הפרמטר 8208 יוחזר `True`.

ב. כתבו פונקציה אשר תדפיס את כל מספרי ארמסטרונג הקיימים ממספרים טבעיים מ-1 ועד 10000. בנוסף, הפונקציה תחזיר את מספר מספרי ארמסטרונג הקיימים בטווח זה.

כותרת הפונקציה היא:

```
def count_armstrong_numbers()
```

חובה להשתמש בפונקציה שכתבתם בסעיף א.

לנוחיותכם, ישנם 16 מספרי ארמסטרונג בטווח זה: כל הספרות (1-9), 153, 370, 371, 407, 1634, 8208, 9474.

¹ https://en.wikipedia.org/wiki/Narcissistic_number

שאלה 3 (25 נק')

דחיסת נתונים מאפשרת לנו לעשות שימוש יעיל במדיה (תמונות, קול, וידאו). הדחיסה מאפשרת יעילות גבוהה בפעולות כגון העברת תמונות על גבי האינטרנט (מקור: ויקיפדיה).

נגדיר את פרוטוקול הדחיסה הבא:

עבור כל תו המופיע ברצף (שתיים ומעלה), ניתן לדחוס אותו כך שיופיע פעם אחת ואחריו כמות המופעים של התו ברצף. אם התו לא מופיע ברצף, יש לדחוס אותו ללא שינוי.

דוגמאות:

עבור המחרוזת "abcde", המחרוזת הדחוסה היא "abcde" (כל תו מופיע לא ברצף).

עבור המחרוזת "aabccccceeedab", המחרוזת הדחוסה היא "a2bc3e5dab".

כתבו פונקציה ששמה decompressed המקבלת את המחרוזת הדחוסה compress. הפונקציה תחזיר את המחרוזת המקורית על פי פרוטוקול הדחיסה שתואר לעיל.

הערות:

- ניתן להניח כי המחרוזת compressed המתקבלת כפרמטר מייצגת אובייקט מטיפוס מחרוזת (str) והתווים במחרוזת מורכבים מאותיות אנגליות קטנות (a-z) וספרות.
- אין לעשות שימוש באופרטור '*' המוגדר על מחרוזות.
- ניתן להניח כי שהמספרים המייצגים את כמות התווים הזהים ברצף שנדחסו מייצגים מספרים חד-ספרתיים (9-2).

שאלה 4 (25 נק')

כתבו פונקציה בשם `count_sub` המקבלת שתי מחרוזות, `s`, `sub`. הפונקציה תחזיר את מספר הפעמים שהמחרוזת `sub` נמצאת במחרוזת `s`. הניחו שהמחרוזת `sub` אינה ריקה.

שימו לב!

- אין להשתמש באף פונקציה מהמחלקה `str`, למעט מנגנון חיתוך (`slicing`).
- הפונקציה הסטטית `len` מותרת.
- מותר להשתמש באופרטור `+` (שרשור).

דוגמאות :

עבור המחרוזות `s="ababa"`, `sub="aba"` יוחזר הערך 2.

עבור המחרוזות `s="aaaaa"`, `sub="aa"` יוחזר הערך 4.

עבור המחרוזות `s="aba"`, `sub="abab"` יוחזר הערך 0.

להלן חתימת השיטה :

```
def count_sub(s, sub)
```

הגשה

1. הגשת הממ"ן נעשית בצורה אלקטרונית בלבד, דרך מערכת שליחת המטלות.
2. יש להגיש קובץ אחד המאגד את כל הפונקציות לשאלות 1-4 בשם `mmn11.py`.
3. ארזו את קובץ הפתרון בקובץ `zip` (ולא `rar`) יחיד ושלחו אותו בלבד.
4. אל תשכחו לשמור את מספר האסמכתא שקיבלתם מהמערכת לאחר ההגשה. אם לא קיבלתם מספר אסמכתא, סימן שההגשה לא התקבלה.
5. שימו לב, אתם יכולים לשלוח שוב ושוב את המטלה במערכת, אם אתם רוצים לתקן משהו בה. כל הגשה דורסת את ההגשה הקודמת. אבל עשו זאת אך ורק עד לתאריך ההגשה. אחרי התאריך, ייחשב לכם כאילו הגשתם באיחור, גם אם ההגשה הראשונה היתה בזמן! כמו כן, אם המנחה הוריד כבר את המטלה שלכם מהמערכת, לא תוכלו לשלוח עותק מעודכן יותר.

בהצלחה