מבוא מורחב למדעי המחשב - תרגיל בית 1

ת.ז : 206867517

שאלה 1 :

קוד

```
""" Question 1 - solution """

my_string = "xyzw"
my_list = ['x', 'y', 'z', 'w']

# string but not list functions

print(str.capitalize(my_string))  # first method
print(my_string.capitalize())  # second method

print(str.replace(my_string, "xy", "<(^__^)>"))  # first method
print(my_string.replace("xy", "<(^__^)>"))  # second method

print(str.istitle(my_string))  # first method
print(my_string.istitle())  # second method

print("\n")
```

```python
# list but not string functions

list.pop(my_list)  # first method
print(my_list)
my_list = ['x', 'y', 'z', 'w']  # pop() is inplace
my_list.pop() # second method
print(my_list)
my_list = ['x', 'y', 'z', 'w']  # pop() is inplace

list.clear(my_list)  # first method
print(my_list)
my_list = ['x', 'y', 'z', 'w']  # clear() is inplace
my_list.clear()  # second method
print(my_list)   # clear() returns None
my_list = ['x', 'y', 'z', 'w']  # clear() is inplace

list.insert(my_list, 3, "<(^__^)>")  # first method
print(my_list)
my_list = ['x', 'y', 'z', 'w']  # insert() is inplace
my_list.insert(3, "<(^__^)>")  # second method
print(my_list)
```

פלט

```
Run: dry_questions
/usr/bin/python3.9 /home/ram/Documents/LocalTAU/cs1001/cs1001py/hw/hw1/dry/dry_questions.py
Xyzw
Xyzw
<(^__^)>zw
<(^__^)>zw
False
False


['x', 'y', 'z']
['x', 'y', 'z']
[]
[]
['x', 'y', 'z', '<(^__^)>', 'w']
['x', 'y', 'z', '<(^__^)>', 'w']

Process finished with exit code 0
```

שאלה 2:

א.

1. קליטת טיפוס שאינו מחרוזת:

```
/usr/bin/python3.9 /home/ram/Documents/LocalTAU/cs1001/cs1001py/hw/hw1/dry/dry_questions.py
Traceback (most recent call last):
  File "/home/ram/Documents/LocalTAU/cs1001/cs1001py/hw/hw1/dry/dry_questions.py", line 68, in <module>
    print(control_digit(20686751.2))
  File "/home/ram/Documents/LocalTAU/cs1001/cs1001py/hw/hw1/dry/dry_questions.py", line 49, in control_digit
    assert isinstance(israeli_id, str)
AssertionError

Process finished with exit code 1
```

ללא שימוש ב- "assert"תתקבל שגיאת "type" כיוון שלא ניתן לגשת לאינדקס [i] במשתנה מסוג.float

2. קליטת מחרוזת באורך שונה מ-8:

I. ללא שימוש ב- "assert", עבור מחרוזת באורך קטן מ-8 תתקבל שגיאת "אינדקס מחוץ לתחום" כיוון שבמהלך ריצת הלולאה ניסינו לגשת למקום במחרוזת שאינו קיים.

```
/usr/bin/python3.9 /home/ram/Documents/LocalTAU/cs1001/cs1001py/hw/hw1/dry/dry_questions.py
Traceback (most recent call last):
  File "/home/ram/Documents/LocalTAU/cs1001/cs1001py/hw/hw1/dry/dry_questions.py", line 68, in <module>
    print(control_digit("206"))
  File "/home/ram/Documents/LocalTAU/cs1001/cs1001py/hw/hw1/dry/dry_questions.py", line 54, in control_digit
    val = int(israeli_id[i])  # converts chart to int
IndexError: string index out of range

Process finished with exit code 1
```

II. עבור קליטת מחרוזת שאורכה גדול מ-8 לא תודפס שגיאה, אך הפלט יתאם לת.ז שמורכבת מ-7 הספרות הראשונות שנקלטו (מצב לא רצוי כי אין משמעות לפלט.)

ב.

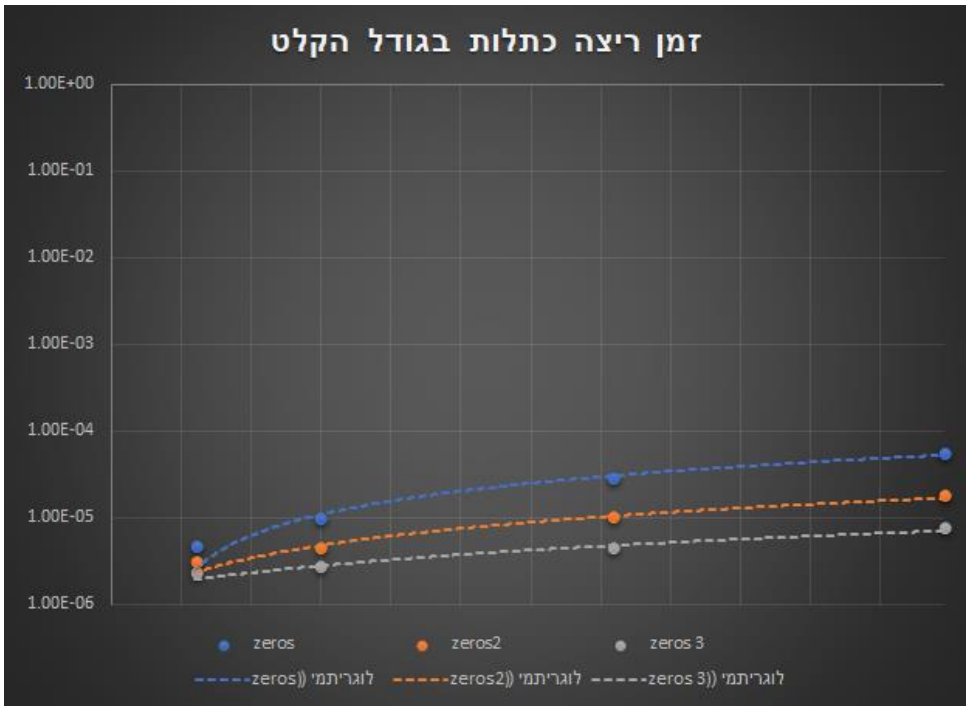
- טבלת מעקב עבור הקלט "87654321":

total	val	ID[i]	i	iteration
8	8	'8'	0	**1**
13	7	'7'	1	**2**
19	6	'6'	2	**3**
20	5	'5'	3	**4**
24	4	'4'	4	**5**
30	3	'3'	5	**6**
32	2	'2'	6	**7**
34	1	'1'	7	**8**

- טבלת מעקב עבור הקלט "20686751" (מספר הת.ז שלי:)

iteration	i	ID[i]	val	total
1	0	'2'	2	2
2	1	'0'	0	2
3	2	'6'	6	8
4	3	'8'	8	15
5	4	'6'	6	21
6	5	'7'	7	26
7	6	'5'	5	31
8	7	'1'	1	33

<u>שאלה 3</u>:



תרשים 1

א .

פתרון / קלט	*2^100*	*2^250*	*2^600*	*2^1400*
zeros	4.5*10^-6	9.4*10^-6	2.8*10^-5	1.1*10^-4
zeros2	3*10^-6	4.2*10^-6	9.8*10^-6	1.4*10^-5

את הנתונים שמדדתי באמצעות המחלקה Time הצגתי כגרף פיזור (תרשים 1). לאחר בחירה באפשרות של קו מגמה לוגריתמי התקבל ערך R^2 הקרוב ביותר ל-1. לכן אני מסיק שקצב הגידול של זמן הריצה כתלות בגודל הקלט עבור שני הפתרונות הראשונים הוא לוגריתמי. בנוסף, ניתן להסיק מתרשים 1 שפתרון

2 יעיל יותר מהפתרון הראשון מבחינת זמני הריצה עבור אותם קלטים שנדגמו, מכיוון שהגרף הכחול נמצא מעל לגרף הכתום ותלילותו של הגרף הכחול גדולה יותר.

ב .

פתרון / קלט	2^100	2^250	2^600	2^1400
zeros3	2.2*10^-6	2.7*10^-6	4.3*10^-6	9.7*10^-6

כמו שניתן לראות בגרף, קו המגמה עבור זמני הריצה של הפתרון השלישי נמצא מתחת לקווי המגמה של הפתרונות הקודמים => הפתרון השלישי יעיל יותר מהשניים הקודמים (כדי להבין את הסיבה בדקתי בקוד המקור של פייתון בגיטהאב וראיתי שכתוב שהמימוש של str.count מתבסס בעיקרו על שני אלגוריתמי חיפוש מתקדמים שאחד מהם נקרא אלגוריתם בויאר-מור.)

ג. נבחר שלושה קלטים עם 10^300 (זה המספר העגול - הכמעט - הכי גדול שהאקסל מסוגל לעבד - וזה הגיוני כי זה בקירוב הערך המקסימלי שמשתנה מטיפוס double יכול להכיל) ספרות ומספר שונה של אפסים:

פתרון / קלט	1111...	101010...	10^300
zeros	1.0*10^-4	1.2*10^-4	1.3*10^-4
zeros2	2.2*10^-5	2.0*10^-5	3.0*10^-5
zeros3	8.2*10^-6	8.5*10^-6	9.0*10^-6

ראשית, בחרתי לבדוק שלושה קלטים בעלי 301 ספרות (ניתן לחשב באמצעות לוג 10) מכיוון שזהו בקירוב הערך העליון שניתן להזין באקסל (הרצתי קלטים גדולים יותר ב SHELL-והתוצאה הובילה אותי לאותה מסקנה). כמו כן, בחרתי בשלושה קלטים שמייצגים מצבי קיצון מבחינת כמות האפסים בקלט. בראשון לא מופיעים אפסים כלל, בשני מחצית מהספרות הן אפסים ובשלישי כל הספרות פרט לספרה הראשונה משמאל הן אפסים. מזמני הריצה שהתקבלו עבור שלושת הפתרונות לכל קלט שבחרתי ניתן לראות שההבדלים בזמני הריצה זניחים. ולכן:

***<u>מסקנה</u>**: "השפעת כמות האפסים בקלט עם מספר ספרות דומה על זמן ריצת הלולאה, זניחה ביחס לזמן הריצה של הלולאה עצמה עבור קלט נתון. "*

כלומר זמן הריצה של cnt = cnt + 1 זניח ביחס לזמן הריצה של הלולאה כולה. לבסוף, המסקנה תקפה לקלטים <u>גדולים מספיק</u> - כיוון שבדקתי עבור קלטים עם 300 ספרות ומעלה.

ד .

נבצע הערכה על בסיס לולאת ה for-שבפתרון השני. בפתרון השני הרעיון הוא לעבור על כל הספרות שמרכיבות את המספר שנקלט, ובכל איטרצייה מתבצעת השוואה בין הספרה הנוכחית לבין התו '0'. בהינתן קלט n כלשהו, ניתן לחשב את מספר הספרות בקלט באמצעות $\log_{10}(n)$ מעוגל כלפי מעלה. לפי המדידות, עבור n=2^600 לקחו לפתרון השני s=6.958027370274 06E-06 שניות לביצוע n=log(2^600) פעולות. כלומר עבור קלט זה מתבצעות v=n/s פעולות לשנייה. כעת נתבסס על ההנחות (הסבירות) / העובדות הבאות:

1. זמן הריצה גדל ככל שהקלט גדל
2. 1000^2 > 600^2 >log(600^2)
3. מספר הפעולות שמתבצעות במסגרת הלולאה הנתונה בסעיף עבור קלט נתון = מספר הפעולות עבור אותו קלט במסגרת הלולאה בפתרון מספר 2

אם כן, הלולאה בפתרון השני תרוץ לפחות (2^600) חלקי v שניות שהם כ-7 E178שנים (!). כלומר פרקטית החישוב לא יתבצע אף פעם ולכן גם החישוב עבור הלולאה הנתונה בסעיף.

<u>שאלה 4</u>:

א. על מנת לאפשר בדיקה דומה עבור אותיות **וספרות** הייתי מבצע השמה ל chars-של המחרוזת:

"abcdefghijklmnopqrstuvwxyz0123456789"ומאתחל את chars_counter להיות רשימה של 36 אפסים, שתפקידה לעקוב אחר מופעים של אותיות ומספרים במחרוזת נתונה.

ב. על ידי הפקודה replace_char(text,char,"") בעצם מחליפים כל מופע של char במחרוזת - בתו הריק ("").זו פעולה שקולה למחיקת כל מופע של char במחרוזת ולכן הפעולה מאפשרת למחוק את כל המופעים של תו מסוים)char (במחרוזת כלשהי)text.(