תרגיל בית מספר 3 - להגשה עד 24 בנובמבר בשעה 23:55

קיראו בעיון את הנחיות העבודה וההגשה המופיעות באתר הקורס, תחת התיקייה assignments. חריגה מההנחיות תגרור ירידת ציון / פסילת התרגיל.

הנחיות והערות ספציפיות לתרגיל זה:

- קוד משאלות 2, 4 ו- 5א' ו- 5ב' יש לממש בקובץ השלד (skeleton3.py) המצורף לתרגיל זה. אין לצרף לקובץ ה-py את הקוד ששימש לפתרון יתר השאלות.

  בקובץ השלד תמצאו פונקציה test שמטרתה לעזור לכם לבדוק את עצמכם. זיכרו שהבדיקות שבה מכסות רק חלק מהמקרים שייבדקו על ידנו.

- כל יתר השאלות – נא להגיש בקובץ pdf בלבד (ולא doc/docx כפי שצוין בעבר).

- שימו לב: מותר להוסיף לקובץ השלד פונקציות נוספות שכתבתם או שנלמדו בכיתה, אך אין לשנות את חתימת הפונקציות שהוגדרו בשאלות, אחרת הבדיקות האוטומטיות ייכשלו.

  לא לשכוח לשנות את שם הקובץ למספר ת"ז שלכם לפני ההגשה, עם סיומת py.

- בסה"כ מגישים שני קבצים בלבד. עבור סטודנט שמספר ת"ז שלו הוא 012345678 הקבצים שיש להגיש הם 012345678.docx (או doc. או pdf.) ו- 012345678.py.

- הקפידו לענות על כל מה שנשאלתם.

- תשובות מילוליות והסברים צריכים להיות תמציתיים, קולעים וברורים.
  להנחיה זו מטרה כפולה:
  1. על מנת שנוכל לבדוק את התרגילים שלכם בזמן סביר.
  2. כדי להרגיל אתכם להבעת טיעונים באופן מתומצת ויעיל, ללא פרטים חסרים מצד אחד אך ללא עודף בלתי הכרחי מצד שני. זוהי פרקטיקה חשובה במדעי המחשב.

CC BY-NC-SA 3.0

אוניברסיטת תל אביב - בית הספר למדעי המחשב
מבוא מורחב למדעי המחשב, חורף 2014

שאלה 1

א. להלן קטע קוד תמים למראה. הריצו אותו והסבירו בקצרה את מה שקורה, בהקשר של מודל הזיכרון.

```
lst = [1,2,3]
for e in lst:
        lst += ["a"]
```

ב. האם יש שינוי כאשר השורה lst += ["a"] הופכת ל- lst = lst+ ["a"] ? הסבירו במשפט.

ג. לכל אחד מקטעי הקוד הבאים, נתחו את סיבוכיות זמן ריצתו כתלות ב- n. הניחו כי כל פעולה בודדת דורשת $O(1)$ זמן. ציינו בבירור את התשובה הסופית, ופרטו את החישוב. על הנימוק להיות קולע, קצר וברור, ולהכיל טיעונים מתמטיים או הסברים מילוליים, בהתאם לצורך.
על התשובה להינתן במונחי $O(\ldots)$, ועל החסם להיות הדוק (נמוך) ככל שניתן. למשל, אם הסיבוכיות היא $O(n)$ ובתשובתכם כתבתם $O(n\log n)$, התשובה לא תקבל ניקוד (על אף שפורמלית O הוא חסם עליון בלבד).

רמז: היעזרו במה שלמדתם על מודל הזיכרון.
רמז נוסף: בכל מקרה מתקבלת סדרה פשוטה.

1.
```
lst = []
i=1
while i<n:
        lst = list(range(i))
        i *= 2
```

2.
```
lst = []
i=1
while i<n:
        lst = lst + [i]
        i *= 2
```

3.
```
lst = []
i=1
while i<n:
        lst += [i]
        i *= 2
```


CC BY-NC-SA 3.0

שאלה 2

כיתבו פונקציה בשם to_decimal שמקבלת כקלט מחרוזת s ומספר b מתוך range(2,37). הפונקציה מחזירה כפלט את המספר הממשי החיובי (טיפוס float) שהייצוג שלו בבסיס b הוא s. אין להשתמש בפונקציות int או float.

בשאלה זו עליכם לבדוק את תקינות הקלט באופן מלא, באמצעות פקודות assert בתחילת הפונקציה (חישבו על כל הבדיקות הנחוצות כאן). קלט תקין יכול להכיל את הספרות 0-9, אותיות מבין 'a' עד 'z' ששייכות לבסיס b ('a' מייצג 10, 'b' מיצג 11 וכן הלאה), וכן נקודה עשרונית אחת, המפרידה בין החלק השלם לשברי (ייתכן שלא תופיע, ואז למעשה המספר הוא שלם). לא יכולים להופיע סימנים אחרים.

דוגמאות הרצה:

```
>>> to_decimal('123.45', 10)
123.45
>>> to_decimal('3e8.6', 16)
1000.375
>>> to_decimal('10q', 30)      #no decimal point here
926                            #return value is of integer type
```

שאלה 3

עליזה ובוב החליפו ביניהם מפתח סודי באמצעות פרוטוקול Diffie-Hellman, כפי שניתן לראות בהרצות הבאות (הפונקציות find_prime ו- DH_exchange הן אלו שראינו בתרגול):

```
>>> p = find_prime(10)
>>> p
593
>>> g, a, b, x, y, key_A = DH_exchange(p)
>>> g, a, b, x, y, key_A
(9, 530, 147, 574, 527, 156)
```

הסטודנטית חווה מנסה לגלות את המפתח המשותף הנ"ל. לשם כך היא מאזינה לתשדורת בין עליזה ובוב, ולכן יש ברשותה את x ואת y ששלחו זה לזו. בנוסף, כמובן, יש ברשותה את p ואת g הפומביים. כעת, חווה מנסה להריץ את הפונקציה crack_DH מהתרגול, שמטרתה לפתור את בעיית ה- Discrete log.

להלן ההרצה שביצעה:

```
>>> crack_DH(p,g,x)
234
```

כפי שניתן לראות, הפונקציה לא גילתה את הסוד המקורי a=530, אלא ערך אחר, a'=234.

האם יכולה חווה, באמצעות המידע שברשותה, לחשב את הסוד המשותף של עליזה ובוב? אם לדעתכם כן, הוכיחו זאת באופן פורמלי, עבור a'≠a כלשהו, ועבור p, g, x, y, b כלשהם. אם לדעתכם לא, הסבירו מדוע לא.

CC BY-NC-SA 3.0

שאלה 4

בהינתן n ו- d שלמים חיוביים, ברצוננו לדעת את שארית החלוקה ב- d של הביטוי

$$p_1^{p_1} + p_2^{p_2} + p_3^{p_3} + \cdots + p_n^{p_n}$$

כאשר p_i הוא הראשוני ה- i. למשל, עבור $n=6$ ו- $d=3$:

$$(1^1 + 2^2 + 3^3 + 5^5 + 7^7 + 11^{11}) \bmod 3 = 1$$

כיתבו פונקציה בשם mod(n,d), המקבלת n ו- d כנ"ל, ומחזירה את שארית החלוקה המתאימה. מותר להניח כי הקלט לפונקציה חוקי.

לדוגמה:

```
>>> mod(6,3)
1
>>> mod(2,5)
0
```

<u>הנחייה</u>: על הפונקציה לסיים את ריצתה תוך פחות מעשר שניות עבור n=5432 ו- d=11, אם מריצים אותה על מחשב "סביר" (כדוגמת אלו שבחוות המחשבים בבניין שרייבר).

<u>רמז</u>: שימו לב שעבור n-ים גדולים, הביטוי הנ"ל עלול להיות גדול מדיי על מנת שיהיה אפשר לחשב אותו בשלמותו.

שאלה 5

א. כתבו פונקציה בשם steady_state(L) שמקבלת כקלט רשימה L של מספרים שלמים שונים זה מזה, ממוינת בסדר עולה, ומחזירה כפלט אינדקס $i \geq 0$ המקיים $L[i] == i$. אם יש יותר מאינדקס אחד כזה, יוחזר אחד מהם – לבחירתכם (שרירותי). אם לא קיים כזה, יוחזר הערך None. דוגמאות הרצה:

```
>>> steady_state([-4,-1,0,3,5])
3
>>> steady_state([-4,-1,2,3,5])
2                              #could also return 3 instead
>>> steady_state([-4,-1,0,4,5])
>>>
```

ב. כתבו פונקציה בשם cnt_steady_states(L) שמקבלת רשימה L כמו בסעיף הקודם, ומחזירה **<u>כמה</u>** אינדקסים i בה מקיימים $L[i] == i$.

(המשך השאלה בעמוד הבא)

CC BY-NC-SA 3.0

דוגמאות הרצה:

```
>>> cnt_steady_states([-4,-1,0,3,5])
1
cnt_steady_states([-4,-1,2,3,5])
2
>>> cnt_steady_states([-4,-1,0,4,5])
0
```

כרגיל, מותר ומומלץ לכתוב פונקציות עזר בהתאם לצורך.

<u>הנחיה לשני הסעיפים</u>: על הפתרון לרוץ במקרה הגרוע בסיבוכיות זמן $O(\log n)$ כאשר n הוא מספר האיברים ברשימה. פתרונות שבמקרה הגרוע רצים בסיבוכיות זמן גבוהה יותר (למשל $O(n)$) יאבדו ניקוד משמעותי.

<u>סוף.</u>

CC BY-NC-SA 3.0