פתרון מבחן בקורס "מבוא כללי לתכנות ולמדעי המחשב"
(הקבץ הסייבר)

סמסטר א' 5-2014

מועד א'

מרצה: אמיר רובינשטיין

משך הבחינה: שעה ו- 45 דקות (15:17 עד 00:19).

חומר עזר מותר: כל חומר עזר, **למעט** אלקטרוני (מחשב, מחשבון) וביולוגי (חברים).

- יש לכתוב את כל התשובות בטופס בחינה זה. מחברות הטיוטה לא ייאספו.
- יש לכתוב את כל התשובות במקום המוקצב ובכתב קריא. חריגות משמעותיות מהמקום המוקצב, תשובות הכתובות בכתב קטן / לא ברור או תשובות שדורשות מאמצים רבים להבנתן עלולות לגרור הורדת ציון.
- במבחן 8 עמודים ו- 7 שאלות – בידקו שכולם בידיכם.
- מומלץ לא "להיתקע" על אף שאלה בודדת, אלא להמשיך לשאלות אחרות ולחזור לשאלה אח"כ.
- קיראו היטב את השאלות. הקפידו לענות בדיוק על מה שנשאלתם, ולנמק אם נדרשתם.

נא לרשום מספר תעודת זהות (ללא שם): _____

בהצלחה !

שאלה 1 (15 נק')

השאלה עוסקת באלגוריתם לחיפוש בינארי ברשימה ממוינת. לנוחיותכם מופיע בהמשך הקוד שנלמד בכיתה.

א. נתונה הרשימה הבאה: L1 = [1,2,3,4,5,6,7,8,9,10]. מבצעים בה חיפוש בינארי למציאת איבר x כלשהו, וידוע שלאחר 4 איטרציות האלגוריתם מחזיר True. כלומר, האלגוריתם בדק בסה"כ 4 פעמים את התנאי

```
if my_list[mid]==x:
```

כאשר ב- 3 הפעמים הראשונות התנאי היה False והאלגוריתם המשיך בחיפוש, ואילו בפעם הרביעית התנאי היה True ואלגוריתם עצר (עם return True).

רישמו את כל הערכים האפשריים עבור x. אין צורך להסביר.

תשובה: 4,7,10

ב. מחפשים ברשימה ממוינת כלשהי L2 שאורכה לא ידוע לכם איבר כלשהו x שלא נמצא בה, באמצעות חיפוש בינארי. ידוע שהאלגוריתם החזיר False לאחר 23 איטרציות. כעת מגדילים את אורך הרשימה פי 2, באופן כזה שהיא עדיין ממוינת ו- x הנ"ל עדיין לא נמצא בה. כמה איטרציות ייקח לאלגוריתם להחזיר False כעת? הסבירו בקצרה.

תשובה: 24. חיפוש בינארי הוא אלגוריתם בעל סיבוכיות לוגריתמית – O(logn). לכן כאשר מגדילים את הקלט שלו פי 2, הוא יבצע עוד מספר קבוע של איטרציות. ואכן, אחרי איטרציה אחת שבה "זורקים" מחצית מהרשימה המוגדלת, מגיעים לגודל שהיה לפני ההגדלה, ולכן מבצעים בדיוק איטרציה אחת יותר.

```
def binary_search(my_list, x):
    ''' search for x in my_list, which MUST BE SORTED !! '''
    left = 0
    right = len(my_list)-1

    while left<=right:
        mid = (left+right)//2
        if my_list[mid]==x:
            return True
        else:
            if my_list[mid] < x:    #go to right half
                left = mid+1
            else:                   #go to left half
                right = mid-1
    return False  #if we got here the search failed
```

שאלה 2 (20 נק')

השאלה עוסקת בקוד המשלב קוד זוגיות (parity bit code) עם קוד חזרות (repetition code).

בתרגיל הבית נשאלתם על קוד שבו, בהינתן הודעה באורך 2, משכפלים אותה פעמיים וגם מוסיפים בסוף ביט זוגיות בין שני הביטים המקוריים. כעת נגדיר קוד מעט שונה: בהינתן הודעה באורך 2, קודם נוסיף לה ביט זוגיות של שני הביטים של ההודעה, ואח"כ נשכפל את 3 הביטים הללו פעמיים (סה"כ ישודרו 6 ביטים). למשל:

01 → 011011

11 → 110110

א. רישמו את יתר ההודעות החוקיות האפשריות (בנוסף ל- 2 הנ"ל).

תשובה:

00 → 000000

10 → 101101

ב. אמיר שולח לנגה שידור חוקי מסוים (6 ביטים), ובדרך נופלות בשידור זה 3 שגיאות (כלומר 3 ביטים כלשהם משתנים). האם נגה תדע להבחין כי השידור שהגיע אליה שגוי (כלומר מכיל מספר כלשהו של שגיאות)? סמנו את התשובה הנכונה, והסבירו בקצרה. אם בחרתם בתשובות I או III, הסבירו. אם בחרתם ב- II, תנו דוגמה לכל מקרה.

I. כן, נגה תמיד תדע שהשידור אינו חוקי, לא משנה היכן נפלו 3 השגיאות.
II. ישנם מקרים בהם נגה תדע זאת, וישנם מקרים בהם היא תחשוב שההודעה שהגיעה אליה היא חוקית.
III. בשום מקרה נגה לא תדע זאת, ותמיד היא תחשוב שהשידור שהגיע אליה חוקי.

הסבר: 3 שגיאות תמיד יתגלו. הסבר אפשרי הוא שמספר ה- 1 בכל הודעה חוקית הוא 0 או 4, ולכן אם נפלו 3 שגיאות בהודעה חוקית, מספר ה- 1 לא יכול להישאר חוקי (כלומר בטוח לא יהיה 0 ולא יהיה 4), ולכן נגה תוכל להבחין בכך. יש הסברים אפשריים אחרים.

ג. אמיר מציע את הרעיון הבא: כדי לשפר את היכולת לגלות שגיאות, הוא יוסיף לקוד הנ"ל עוד ביט זוגיות אחד על כל 6 הביטים הראשונים. כלומר השידורים יהיו כעת באורך 7 ביטים, כאשר הביט האחרון הוא ביט זוגיות של 6 הראשונים שנוצרו כמתואר בתחילת השאלה. לדוגמה (הביט הנוסף מסומן בקו): 01 → 0110110

הקיפו את הטענה הנכונה והסבירו בקצרה:

I. ההצעה של אמיר מאפשרת במקרים מסויימים לגלות יותר שגיאות מאשר קודם.
II. ההצעה של אמיר מאפשרת בכל המקרים לגלות אותו מספר שגיאות כמו קודם.
III. ההצעה של אמיר מאפשרת בכל המקרים לגלות פחות שגיאות מאשר קודם.

הסבר: במקרה הזה ביט הזוגיות לא מוסיף כלום, כי הוא תמיד שווה 0 (מדוע?).

שאלה 3 (15 נק')

להלן שלוש פונקציות. כל אחת מקבלת תמונה im ומחזירה את התמונה לאחר מניפולציה כלשהי. עליכם להתאים את הפונקציות לתמונות שמתחת, ולהסביר בקצרה את בחירתכם.

```
def what1(im):
    w,h = im.size
    mat = im.load()

    for i in range(w):
        for j in range(h):
            if i%100 == 0:
                mat[i,j] = 255

    return im
```

```
def what2(im):
    w,h = im.size
    mat = im.load()

    for i in range(w):
        for j in range(h):
            if (i-j)%100 == 0:
                mat[i,j] = 255

    return im
```

```
def what3(im):
    w,h = im.size
    mat = im.load()

    for i in range(w):
        for j in range(h):
            if (i+j)%100 == 0:
                mat[i,j] = 255

    return im
```

תמונה 1

תמונה 2

תמונה 3

<u>תשובה</u>:

what1 מתאימה לתמונה מס' 1

what2 מתאימה לתמונה מס' 2

what3 מתאימה לתמונה מס' 3

<u>הסבר</u>: what1 מלבינה עמודות אנכיות כי j כלל לא משתתף בהחלטה מה להלבין.
what2 מלבינה אלכסונים בהם i-j הוא כפולה של 100. למשל, את האלכסון הראשי (זה שמתחיל בפינה השמאלית העליונה), שבו i=j כלומר i-j=0.
what3 מתאימה לתמונה 3, באלימינציה. ניתן גם לראות כי היא מלבינה אלכסונים בהם i+j הוא כפולה של 100.
אלכסונים כאלו הולכים לכיוון מטה ושמאלה: כאשר i קטן (=שמאלה) j גדל (למטה) באותה מידה בדיוק.

שאלה 4 (15 נק')

אמיר רוצה לשלוח לנגה את מספר כרטיס האשראי שלו באופן מוצפן. הוא מציע את השיטה הבאה: במקום לשלוח את המספר עצמו, הוא יבצע *הסטה* של ספרות המספר k מקומות שמאלה (באופן מעגלי). למשל, אם מספר הכרטיס הוא 12345678, ו- k=2, הוא ישלח את 34567812.

עליכם להשלים את הפונקציה הבאה encrypt, שמקבלת מחרוזת st המייצגת מספר, ושלם חיובי k, ומחזירה (return) את ההסטה של המספר st בדיוק k מקומות שמאלה באופן מעגלי כמתואר לעיל. דוגמאות הרצה:

```
>>> encrypt("11234", 1)
'12341'
>>> encrypt("11234", 2)
'23411'
>>> encrypt("11234", 3)
'34112'
>>> encrypt("11234", 4)
'41123'
>>> encrypt("11234", 5)
'11234'
```

השלימו את הפונקציה:

```
def encrypt(st, k):

    res = "" #empty string

    for i in range(_______k_______, ____len(st)___):

        res = res + _____st[i]____

    for i in range(_______k_______):

        res = res + st[i]_________

    return _______res_____________
```

לנוחיותכם, תזכורת לאופן הפעולה של range:

```
>>> for i in range(3,7):
        print(i)

3
4
5
6
```

שאלה 5 (10 נק')

בכיתה ראינו את שתי הפונקציות הבאות, כחלק ממימוש צופן החלפה (substitution cipher):

```
import random

def create_cipher(alphabet):
    original = list(alphabet)
    shuffled = list(alphabet)
    #print(shuffled)  # before
    random.shuffle(shuffled)
    #print(shuffled)  # after

    encrypt_dict = {} #a dictionary
    for i in range(len(original)):
        encrypt_dict[original[i]] = shuffled[i]

    return encrypt_dict

def encrypt(text, enc_dict):
    """ converts text using en_dict as substitution key"""
    result = ""
    for ch in text:
        if ch in enc_dict:
            result += enc_dict[ch]
        else:
            result += " " #characters not in the alphabet are replaced by spaces
    return result
```

התבוננו בהרצות הבאות:

```
>>> cipher = create_cipher("abcd")

>>> cipher

{'a': 'a', 'c': 'd', 'b': 'c', 'd': 'b'}

>>> encrypt(encrypt("abcd", cipher), cipher)

???
```

מהו הפלט של הפקודה האחרונה? <u>סמנו בעיגול</u> את התשובה הנכונה והסבירו בקצרה.

I. 'adbc' II. 'aaaa' III. 'abcd' IV. 'dcba'

<u>הסבר</u>:

```
encrypt(encrypt("abcd", cipher), cipher)
=
encrypt("acdb", cipher)
=
"adbc"
```

שאלה 6 (15 נק')

לכל אחת מהטענות הבאות ציינו האם היא נכונה או לא, והסבירו. אם לדעתכם הטענה לא נכונה, אפשר להסביר ע״י דוגמה נגדית לגרף שמפריך את הטענה.

שני הסעיפים מתייחסים לגרפים בלתי מכוונים.

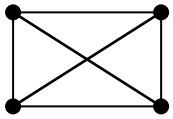
תזכורות:

- **מסלול אוילר** בגרף הוא מסלול שעובר בכל קשת בדיוק פעם אחת.
- **מעגל** בגרף הוא מסלול שבו הצומת הראשון והאחרון שווים. אורך המעגל הוא מספר הקשתות בו.
- **גרף מלא (קליקה)** הוא גרף שבו יש קשת בין כל שני צמתים שונים.

א. לכל גרף שיש בו מעגל באורך זוגי, יש בו גם מסלול אוילר.

הטענה נכונה / לא נכונה (הקיפו בעיגול)

לדוגמה, בגרף הבא יש מעגל עם 4 קשתות (זוגי), אבל אין מסלול אוילר (כי כל הדרגות איזוגיות):



דוגמה אחרת אפשרית היא הגרף שמייצג את בעיית הגשרים של קניגסברג.

למעשה אין שום קשר בין שתי התכונות הנ״ל.

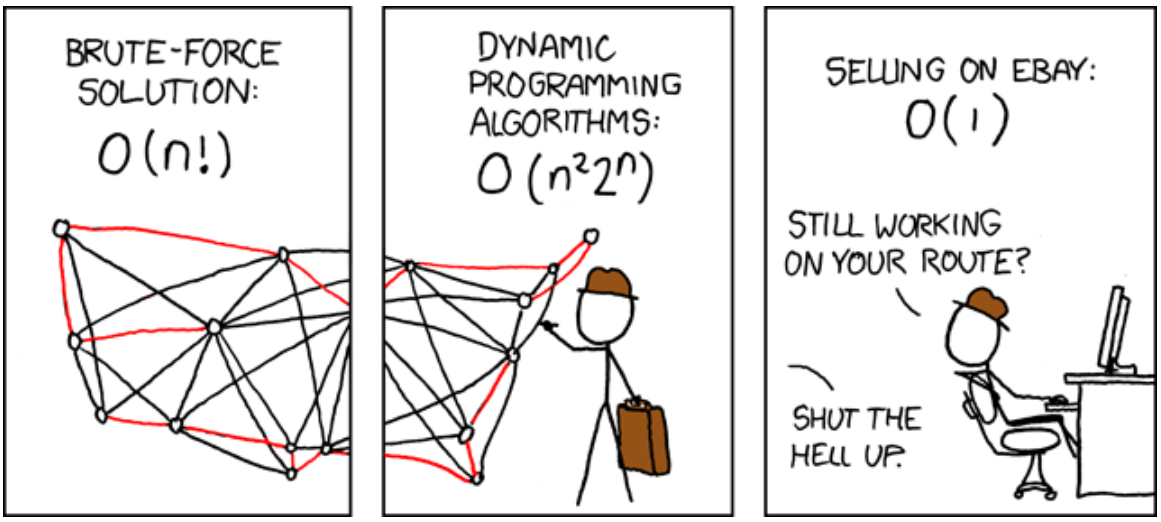
ב. בגרף המלא (קליקה) עם 117 צמתים יש מעגל אוילר.

הטענה נכונה / לא נכונה (הקיפו בעיגול)

בגרף המלא עם 117 צמתים דרגת כל צומת היא 116 (כי הוא מחובר לכל יתר הצמתים). לפיכך כל הדרגות זוגיות, ולכן יש בגרף מעגל אוילר (למעשה הדבר נכון עבור כל מספר אי-זוגי של צמתים).

שאלה 7 (10 נק')

להלן בדיחה קצרה על "בעיית הסוכן הנוסע":



הסבירו בקצרה את תוכן הבדיחה, תוך שימוש נכון במושגים הבאים: המחלקה NPC, פתרון brute-force, וסיבוכיות זמן ריצה O(n!) ו- O(1).

אין צורך להתייחס לחלק האמצעי של הבדיחה.

<u>תשובה</u>: בעיית הסוכן הנוסע שייכת למחלקת הבעיות ה"קשות אבל קלות לאימות" – NP. למעשה היא אפילו שייכת למחלקה NPC של הבעיות שמאוד לא סביר שיש להן פתרון פולינומי. לפיכך, הסוכן מנסה לפתור את הבעיה בדרך הנאיבית, ע"י אלגוריתם "כח-גס", שפשוט עובר על כל המסלולים האפשריים ובוחר מביניהם את הזול ביותר. פתרון כזה הוא אקספוננציאלי, כמספר המסלולים האפשריים, O(n!). לעומתו, סוכן אחר מצא דרך "יעילה" לפתור את הבעיה בזמן קבוע O(1)- להשתמש ב- ebay ובכך למכור את מרכולתו ללא צורך למצוא מסלול זול ביותר.

סוף!