מס' ת"ז_____ מס' מחברת_____

מבחן בקורס מבוא מורחב למדעי המחשב, CS1001.py

סמסטר א' 2013

מועד א

מרצים: פרופ' בני שור, ד"ר דניאל דויטש

מתרגלים: אילן בן בסט, אמיר רובינשטיין, אדם ויינשטוק

משך הבחינה: 3 שעות.

חומר עזר מותר: שני דפי עזר (דו צדדיים) בגודל A4 כ"א.

- יש לכתוב את כל התשובות בטופס הבחינה. המחברת תשמש כטיוטה בלבד ולא תקרא.
- במבחן 12 עמודים – בידקו שכולם בידיכם. המבחן מכיל 4 שאלות "חצי סגורות" ושתי שאלות "פתוחות", אשר בכ"א מהן שני סעיפים. מומלץ לקרוא תחילה את כל השאלות.
- אנו ממליצים לא "להיתקע" על שום שאלה בודדת, להמשיך לשאלות אחרות, ולחזור לשאלה אח"כ.
- בכל סעיף בשאלות הפתוחות, ניתן לכתוב "איני יודע/ת" ולא לכתוב שום טקסט נוסף. במקרה זה יינתן 20% מציון הסעיף (מעוגל כלפי מעלה).
- יש לכתוב את כל התשובות במסגרת המקום המוקצב ובכתב קריא. חריגות משמעותיות מהמקום המוקצב, או הכתובות תוך שימוש בפונט זערורי, לא ייקראו. תשובות שמצריכות עזרים אופטיים מתקדמים לקריאתן, כאלה הכתובות בכתב חרטומים, או הדורשות כוחות על טבעיים להבנתן, עלולות לגרום להשפעה שלילית על הציון.
- לכתיבת קוד יש להקדים תיאור מילולי קצר שיסביר את הקוד במקום המיועד לכך.
- ניתן לצטט טענות שנטענו בהרצאה, בתרגול, או בשיעורי הבית.

בהצלחה !

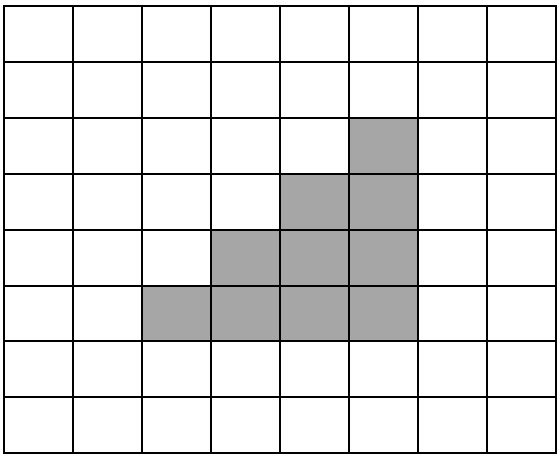
טבלת ציונים: (לשימוש הבודקים)

שאלה	ניקוד	
1		
2		
3		
4		
5		
6		
סה"כ		

<u>שאלה 1 (14 נק')</u>

במהלך צילום תמונת לוויין של אנטארקטיקה (שהיא כידוע לבנה לגמרי) חלף בשמי הקוטב מטוס חמקן (שהוא כידוע שחור לגמרי). על מנת להעלים ראיות לקיומו של המטוס הסודי הוחלט בממשלה לנקות את הכתם שנוצר בתמונה ע"י ביצוע איטרציות של local medians עם k=1, עד לניקוי מוחלט של הכתם (הפיכתו ללבן).

להלן התמונה המוכתמת (10 הפיקסלים בתמונה שאינם לבנים הם שחורים).

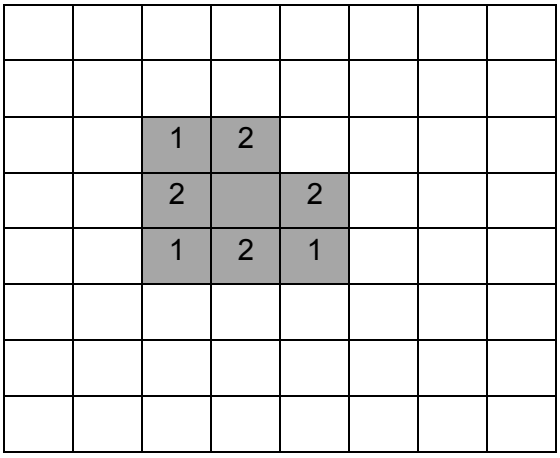


א. לאחר כמה איטרציות יעלם הכתם לחלוטין (סמנו ∞ אם קיים פיקסל שלא יתנקה לעולם)?

תשובה: _____

ב. **בתרשים למעלה**, סמנו את מצב הכתם לאחר ביצוע <u>שתי</u> איטרציות של local medians. בכל פיקסל מוכתם (מסומן באפור), רישמו באיזו איטרציה הוא התנקה. פיקסל שלא נוקה לאחר שתי איטרציות יש להשאיר ריק.

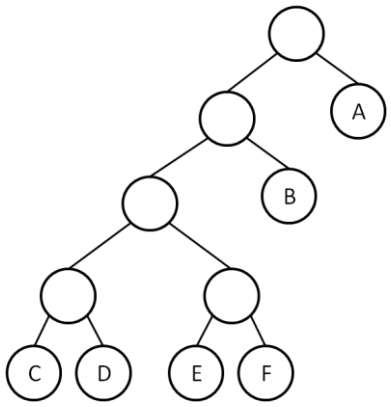
לדוגמא (עבור **כתם אחר**):



<u>שאלה 2 (14 נק')</u>

א. תנו דוגמה לקורפוס שממנו יכול להיווצר עץ האפמן הבא.

נקבל כל דוגמה שמתאימה לעץ הנתון. שימו לב כי הדוגמה שלכם יכולה להתאים גם לעצים אחרים.



הקורפוס: _____

ב. ציירו עץ האפמן המתקבל מתדירויות הקורפוס הבא: ABCDEEEFFF

שימו לב שייתכן יותר מעץ האפמן חוקי אחד שמתאים לקורפוס הזה, ואנו נקבל כל עץ חוקי.

<u>שאלה 3 (12 נק')</u>

קוד החזרה שראינו בהרצאה, $R:\{0,1\}^2 \rightarrow \{0,1\}^6$, מעתיק שני ביטים b_0b_1 לשישה ביטים, ע"י שכפול כל אחד מהביטים המקוריים 3 פעמים: $b_0b_1 \rightarrow b_0b_0b_0b_1b_1b_1$. יש ארבע מילות קוד בקוד זה, והן איברים של $\{0,1\}^6$.

אלו מהטענות הבאות מתקיימות ביחס לקוד החזרה הנ"ל? <u>הקיפו בעיגול</u>.

א) לכל איבר x של $\{0,1\}^6$ שאיננו מילת קוד, יש מילת קוד c יחידה שהיא הקרובה ביותר ל-x, והמרחק בין c ל-x הוא 1.

ב) לכל איבר x של $\{0,1\}^6$ שאיננו מילת קוד, יש מילת קוד c יחידה שהיא הקרובה ביותר ל-x, והמרחק בין c ל-x הוא 2.

ג) לכל איבר x של $\{0,1\}^6$ שאיננו מילת קוד, יש מילת קוד c יחידה שהיא הקרובה ביותר ל-x, והמרחק בין c ל-x יכול להיות 1 או 2.

ד) כל האפשרויות הקודמות אינן נכונות.

הצדיקו את תשובתכם ע"י מתן הסבר קצר או דוגמא קצרה שיכתבו בשורות הבאות (הסברים ארוכים יותר יקוצצו בבדיקה):

<u>שאלה 4 (8 נק')</u>

קוד "הארנב והצב" שהוצג בכיתה מזהה האם יש מעגל ברשימה מקושרת, ומשתמש לשם כך בזיכרון בגודל קבוע, שאינו תלוי באורך הרשימה. תלמיד חסר סבלנות הציע להאיץ את ביצוע הקוד, באופן הבא: בהינתן פרמטרים שהם מספרים טבעיים $0 < s < f$, הצב יבצע בכל איטרציה s התקדמויות לאורך הרשימה המקושרת, והארנב יבצע f התקדמויות. הקוד המתאים להצעה מוצג כאן. שימו לב כי עבור s=1 ו-f=2 אנו מקבלים קוד שקול (אך כתוב באופן שונה במקצת) לקוד המקורי מההרצאה.

```
def detect_cycle3(lst,f,s):
    assert isinstance(f,int) and isinstance(s,int)
    assert f>s>0
    slow = fast = lst
    while True:
        if slow==None or fast==None:
            return False
        for i in range(f):
            if fast.next==None:
                return False
            else:
                fast=fast.next
        for j in range(s):
            slow=slow.next
        if (slow is fast):
            return True
```

אלו מהטענות הבאות מתקיימות ביחס לקוד שהוצע ע"י אותו תלמיד חסר סבלנות? <u>הקיפו בעיגול</u>, והצדיקו את תשובתכם ע"י מתן הסבר קצר או דוגמא קצרה שיכתבו בשורות הבאות (הסברים ארוכים יותר יקוצצו בבדיקה):

א) לכל זוג מספרים טבעיים $0 < s < f$ ולכל רשימה מקושרת lst (מעגלית או קווית), הקוד עוצר ומחזיר תשובה נכונה. לכל זוג מספרים טבעיים $0 < s < f$ ורשימה מקושרת lst הקוד החדש מבצע פחות השמות next מהקוד המקורי.

ב) יש זוגות מספרים טבעיים $0 < s < f$ ורשימות מקושרות **קוויות** lst, עבורן הקוד עוצר ומחזיר תשובה שגויה.

ג) יש זוגות מספרים טבעיים $0 < s < f$ ורשימות מקושרות **מעגליות** lst, עבורן הקוד אינו עוצר.

ד) לכל זוג מספרים טבעיים $0 < s < f$ ולכל רשימה מקושרת lst (מעגלית או קווית), הקוד עוצר ומחזיר תשובה נכונה. יש זוגות מספרים טבעיים $0 < s < f$ ורשימות מקושרות lst עבורם הקוד החדש מבצע יותר השמות next מהקוד המקורי.

<u>שאלה 5 (25 נקודות)</u>

בשאלה זו נדון במשימה הבאה: בהינתן שתי רשימות של מספרים, אחת באורך n והשניה באורך m, מטרתנו היא למצוא את האורך המקסימלי של תת רשימה (רצופה) שמופיעה בשתיהן. נסמן אורך זה ב- k.

נעיר כי הגודל של k **אינו ידוע מראש** לתוכנית או למתכנת/ת.

<u>סעיף א' (12 נקודות)</u>

השלימו בעמוד הבא את הפונקציה max_repetition1, כך שתבצע את המשימה בסיבוכיות זמן O(m*n*k).

תנו תחילה הסבר מילולי קצר, שמתאר באופן כללי מאוד את האלגוריתם שלכם.

def max_repetition1(lst1,lst2):

<u>סעיף ב' (13 נקודות)</u>

נניח כעת ששתי רשימות המספרים הנתונות כקלט **ממוינות בסדר עולה**, וכי אין חזרות בתוך אותה רשימה. בהינתן ההנחות הללו, כתבו גרסה חדשה של הפונקציה למציאת האורך המקסימלי של תת רשימה (רצופה) שמופיעה בשתי רשימות הקלט, בשם max_repetition2, שפועלת בסיבוכיות זמן ריצה O(m+n).

<u>רמז</u>: אפשר להיעזר בקוד או ברעיון של הפונקציה merge שנלמדה בכיתה (ומופיעה לנוחותכם בעמוד הבא).

תנו **בעמוד הבא** הסבר מילולי קצר, שמתאר באופן כללי מאוד את האלגוריתם שלכם.

```
def max_repetition2(lst1,lst2):
```

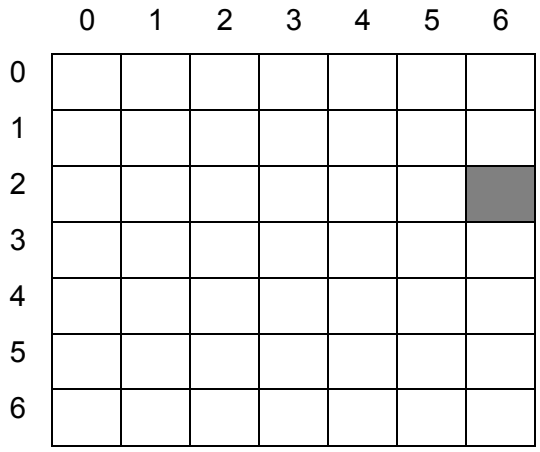
תיאור מילולי קצר של האלגוריתם:

הפונקציה merge שנלמדה בכיתה:

```
def merge(lst1, lst2):
    n1 = len(lst1)
    n2 = len(lst2)
    lst3 = [0 for i in range(n1 + n2)]  # alocates a new list
    i = j = k = 0  # simultaneous assignment
    while (i < n1 and j < n2):
        if (lst1[i] <= lst2[j]):
            lst3[k] = lst1[i]
            i = i +1
        else:
            lst3[k] = lst2[j]
            j = j + 1
        k = k + 1  # incremented at each iteration
    lst3[k:] = lst1[i:] + lst2[j:]  # append remaining elements
    return lst3
```

<u>שאלה 6 (25 נקודות)</u>

צוללת בגודל 1×1 חבויה באוקיינוס ריבועי המחולק ל-$n \times n$ שטחים. אוקיינוס לדוגמא נראה כך:



להלן מוצג קוד של המחלקה Ocean המאפשרת שאילתות לחיפוש הצוללת באוקיינוס.

```
1. class Ocean:
2.     def __init__(self,n,x,y):
3.         self.n = n
4.         self.subX = x
5.         self.subY = y
6.
7.     def __repr__(self):
8.         return "ocean: {0}x{0}".format(self.n)
9.
10.    def size(self):
11.        return self.n

…

…

12.    def ask(self,x,y):
13.        if self.subX > x:
14.            xRes = 1
15.        elif self.subX == x:
16.            xRes =  0
17.        else:
18.            xRes =  -1
19.        if self.subY > y:
20.            yRes = 1
21.        elif self.subY == y:
22.            yRes =  0
23.        else:
24.            yRes =  -1
25.        return xRes,yRes
```

א. (20 נק') השלימו את הקוד של הפונקציה findSub המקבלת אובייקט מסוג Ocean ומחזירה את הקואורדינטות (x,y) בהן מתחבאת הצוללת. למשל, עבור האוקיינוס ocean בדוגמא נקבל:

```
>>> findSub(ocean)
(2,6)
```

<u>הנחיות</u>:

1. אין להשתמש בשדות הפנימיים subX,subY אלא בשאילתות לפונקציה ask בלבד.
2. פתרון בזמן ריצה ליניארי (או גרוע מזה) יזכה בניקוד חלקי ביותר.

תנו **בעמוד הבא** הסבר מילולי קצר, שמתאר באופן כללי מאוד את האלגוריתם שלכם.

```
def findSub(ocean):
```

תיאור מילולי קצר של האלגוריתם:

ב. (5 נק') מבין האפשרויות הבאות, הקיפו בעיגול את זמן הריצה של הקוד שלכם, והסבירו בקצרה (אין צורך בהוכחה פורמלית, רק בנימוק משכנע). יש לסמן את החסם ההדוק ביותר המתאים לקוד שלכם.

$O(1)$, $O(\log n)$, $O(\log^2 n)$, $O(n^{1/2})$, $O(n)$, $O(n^2)$, $O(2^n)$

סוף!