מועד א' חורף 2014 - פתרונות חלקיים והערות על הבדיקה

שאלה 1 (20 נק')

א.

```
def multi_merge1(lst_of_lsts):
    m = len(lst_of_lsts)
    merged = []

    for i in range(m)                                    :

        merged = merge(merged, lst_of_lsts[i]

    return merged
```

ב. 1. $O(m)$

2. $O(n \cdot m^2)$

ג. פתרון נכון יבצע שתי קריאות רקורסיביות, לכל אחד מחצאי lst_of_lsts, ואח"כ ימזג את שני החצאים ע"י merge. טעויות רבות הופיעו באינדקסים שמועברים לקריאות הרקורסיביות. למשל r//2 כאינדקס האמצע (זה לא נכון כאשר l איננו עוד שווה לאפס).

שאלה 2 (20 נק')

א. התשובה היא: 1 / **2** / 3 / 4

ב. התשובה היא: 1 / 2 / 3 / **4**

ג. התשובה היא: 1 / 2 / **3** / 4 ac → "01001", cb → "01001"

ד.

```
dict={"0":"a","01":"b","011":"c"} # ניתן להשתמש בזה, אבל לא חובה

def decode(b):
    result=[]
    n=len(b)
    p=0
    while p<n:
```

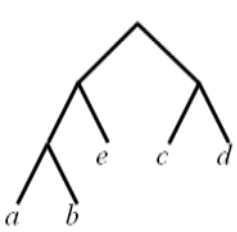
```
k=p+1
while k<n and b[k]=="1":
  k +=1
result.append(dict[b[p:k]])
p = k
```

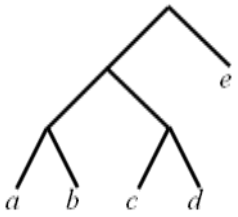
פתרונות נכונים אחרים: פיענוח מימין לשמאל, או בדיקה אד-הוק של כל המקרים האפשריים. בפתרון האד-הוקי, צריך לוודא טיפול בקצה הקלט. אלגוריתם שפועל רק עבור קוד חפשי רישא לא קיבל נקודות.

```
    return "".join(x for x in result)
```

שאלה 3 (20 נק')

א. שני עצים אפשריים:





ב.

הטענה נכונה / לא נכונה (הקיפו בעיגול)

יש פתרונות רבים אפשריים, למשל עם התדירויות 1,1,1,2,3 או 1,1,2,2,4

ג.

מישל צודקת / **ברק צודק** (הקיפו בעיגול)

יש פתרונות רבים אפשריים, למשל עם התדירויות 2,3,4,5

שאלה 4 (20 נק')

א. הרעיון: יש לבצע חיפוש בינארי על השורה. פתרון ליניארי קיבל לכל היותר 3 נקודות. רוב הטעויות היו באי דיוקים באינדקסים. לפעמים הדבר אף גרם ללולאות אינסופיות.

ב. הרעיון הוא לעבור על כל שורה ולחשב את ההפרש בין האינדקס הימני ביותר לבין האינדקס השמאלי ביותר של פיקסל שחור (אם קיימים). שני האינדקסים הללו יחושבו ע"י קריאות לפונקציות מהסעיף הקודם / של גברת מוח.

שאלה 5 (20 נק')

סעיף א: 3. ייוצרו רק זוגות מהצורה (0,n)

סעיף ב:

```
...
for j in range(i):
    yield(j)
i+=1
```

סעיף ג:

```
pairs = PairsGen()
while True:
  pair = next(pairs)
  yield (pair[1],pair[0])
```

סעיף ד:

```
while True:
  yield next(gen1)
  yield next(gen2)
```