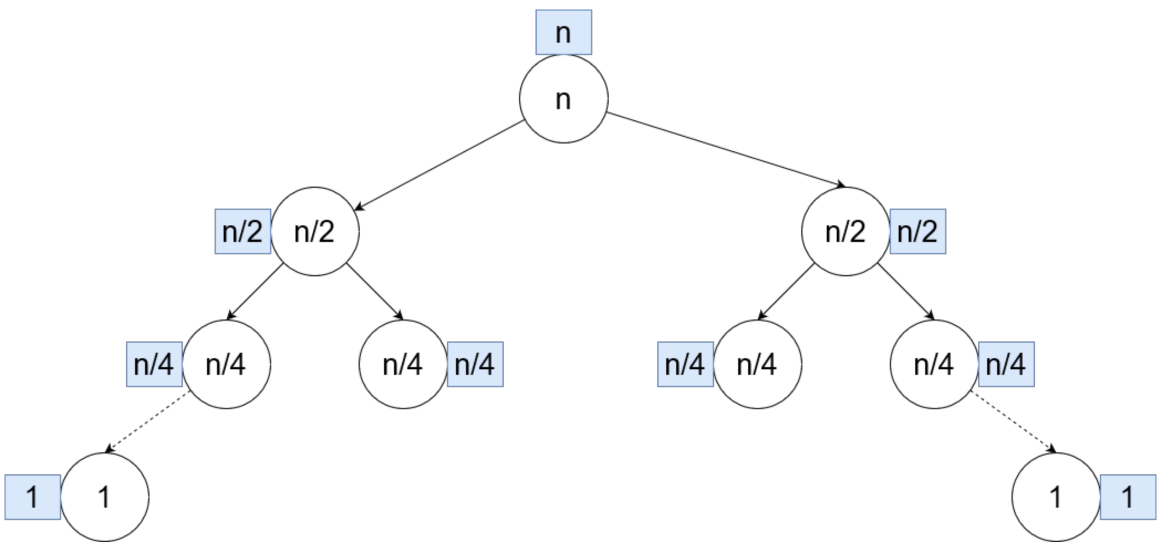
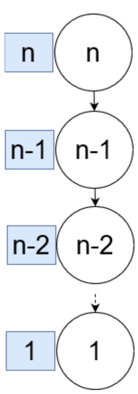
<u>שאלה 1</u>

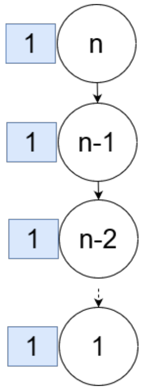
<u>סעיף א׳</u>

1. בעץ הרקורסיה הנ״ל כל צמת מייצג קריאה רקורסיבית ל**max_v1**. בתוך הצומת מצוין אורך הרשימה ולצדו מצוינת כמות העבודה שמתבצעת. בכל צומת מתבצע **slicing** פעמיים, כך שבכל פעם אורך ה**slice** הוא חצי אורך הרשימה. סך אורך ה**slice** הוא אורך הרשימה בצומת וזהו סך כמות העבודה שמתבצעת בצומת. ניתן לראות שסכום כמות העבודה בכל שורה בעץ הוא **n**. כיוון שעומק הרקורסיה הוא **log(n)**, כמות העבודה הכוללת של רשימה באורך **n** היא **n*log(n)**. סה״כ סיבוכיות של **O(n*log(n))**.



2. הפעם בכל צומת מתבצע **slicing** יחיד באורך **n-1** ולכן כמות העבודה בכל צומת היא אורך הרשימה בצומת. כמו כן בכל צומת מתבצעת קריאה רקורסיבית על הרשימה פחות האיבר הראשון בה. סה״כ בעץ ישנן **n** שורות. נסכום את סך כמות העבודה בעץ (סכום סדרה חשבונית) ונקבל **0.5*(n**2 +n)**. על כן קיבלנו סיבוכיות של **O(n**2)**.





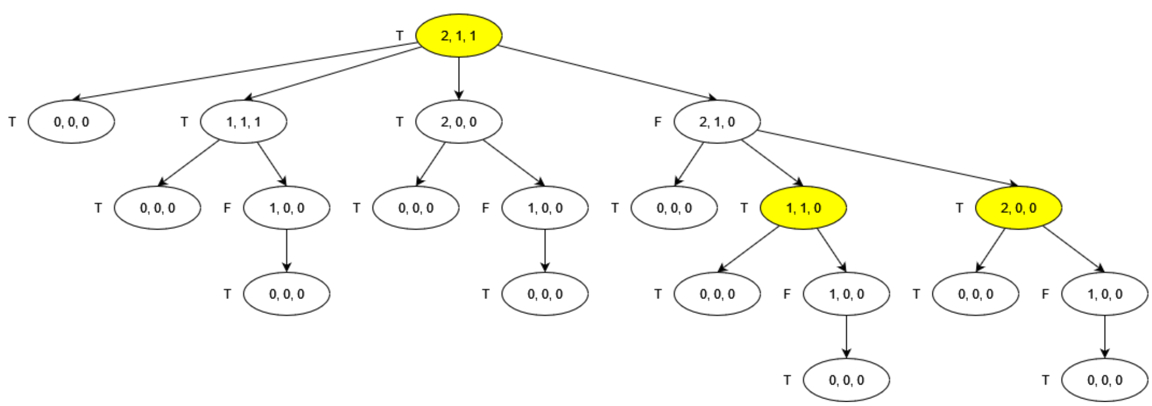
3. בכל צומת כמות העבודה היא **O(1)**. כמו כן בכל צומת מתבצעת קריאה רקורסיבית על הרשימה פחות איבר אחד. סה״כ בעץ ישנן **n** שורות. נסכום את סך כמות העבודה בעץ ונקבל **n**. על כן קיבלנו סיבוכיות של **O(n)**.

<u>סעיף ג׳</u>

i. יודפס:

```
recommended move: [1, 1, 0] --> [1, 0, 0]
recommended move: [2, 0, 0] --> [1, 0, 0]
recommended move: [2, 1, 1] --> [2, 1, 0]
```

בעוד שהפונקציה המקורית הדפיסה דרך מנצחת אם קיימת לשחקן הראשון שהיא מתאפשרת לו, הפונקציה הנוכחית תדפיס דרך מנצחת אם קיימת רק לשחקן שפותח את המשחק (שהוגדר מראש להיות **player 0**). (על כן הפלט הנוכחי מציג דרך לשחקן שמתחיל את המשחק לנצח אותו).



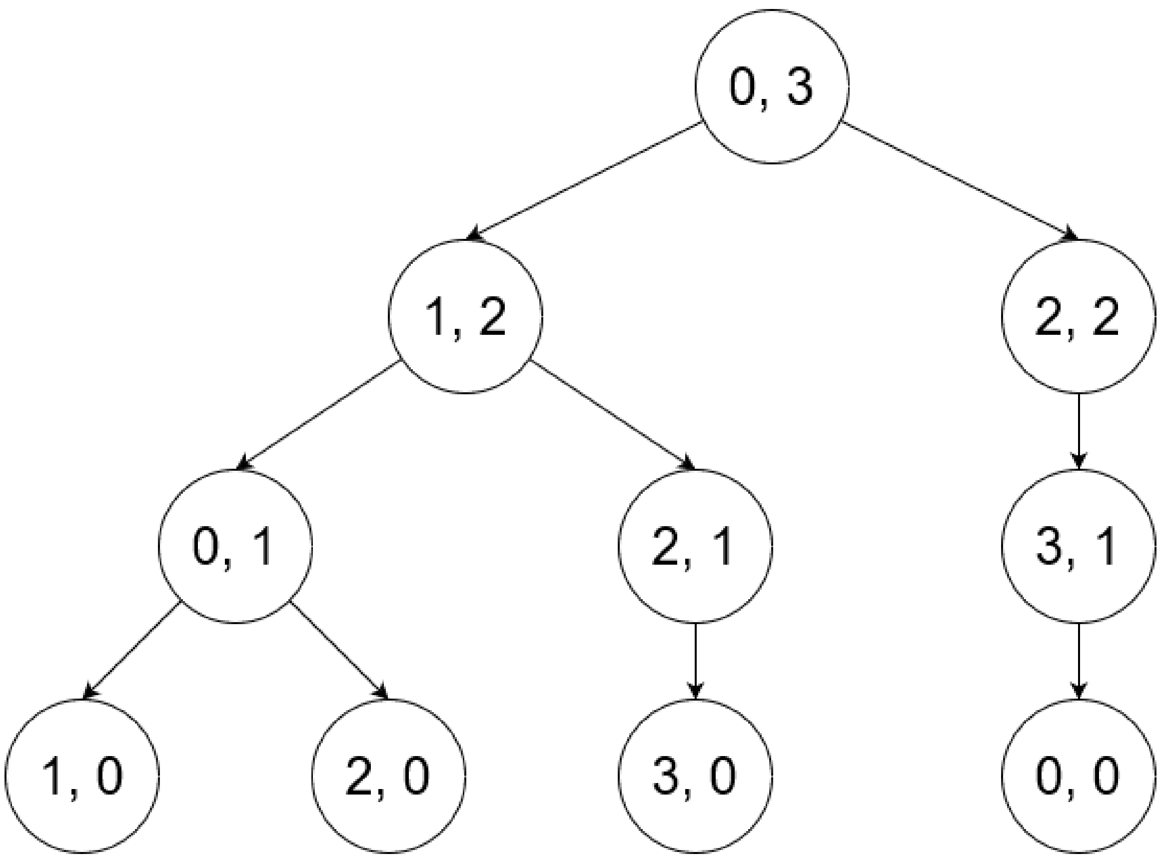
iii. (בספירה, חיפוש מסוג ״**if d[key] == val**״ נחשב הצלחה באופן אוטומטי כיוון שכל חיפוש כזה מחזיר ערך שחושב קודם לכן)

	8 * [5]	**16 * [5]**
חיפושים במילון	9940	275328
חיפושים שהסתיימו בהצלחה	8423	249779

<u>שאלה 2</u>

<u>סעיף ב׳</u>

i. בכל צומת הערכים הם **s, k** בהתאמה.



ii. יהי n. עבור:

- A = [[0, 1, 1, 0, …, 0], [1, 0, 1, 0, …, 0], [1, 1, 0, …, 0] * (n-2)]

(הכוונה ב$(n-2)*$ היא n-2 עותקים של הרשימה, הכוונה היא כזו גם בדוגמאות הבאות).

- $s = 0, t = n, k = n$

נקבל שהסיבוכיות היא $O(2**n)$ כיוון שבעץ הרקורסיה נקבל שלכל צמת יש שני בנים בדיוק, ועומק העץ, כערך k, הוא n.

בכל קריאה לפונקציה מתבצעות שתי קריאות רקורסיביות נוספות (לא כולל בעלים של העץ כמובן), כיוון שכל אחת מהקריאות הרקורסיביות מחזירות False (לא ניתן להגיע לs=n כיוון שאף נקודה לא מתחברת לנקודה n). לכן הלולאה לא תיקטע ולא יהיו "דילוגים".

סעיף ג'

i. עבור

- A =[[0, 1, 1, 0, 0], [1, 0, 1, 0, 0], [0, 0, 0, 1, 0], [1, 0, 0, 0, 0], [0, 0, 0, 0, 0]]
- k=3 ,t=4 ,s=0

כלל לא יהיה פלט, הפונקציה תקלע לרקורסיה אין סופית כיוון שעבור קריאה לפונקציה עם k=1 תמיד תגרור קריאה נוספת לפונקציה עם k=1

(mid=0, **k-mid**=1).

iii. עבור

- A =[[0, …, 0] * n]
- k=n ,t=n ,s=0

נקבל שהסיבוכיות היא $O((n)**log(n-1))$ כיוון שבעץ הרקורסיה נקבל שלכל צומת יש לפחות n בנים (בתנאי עם הקריאה הכפולה נקבל False כבר בקריאה הראשונה ולכן לא נבצע את השנייה, אבל כן נעבור לאיטרציה הבאה בלולאה שכן התנאי איננו True, סה"כ בצענו n קריאות רקורסיביות- כמספר האיטרציות) ועומק הרקורסיה הוא log(n-1) (בכל שורה מחלקים את k ב 2, n-1 כיוון שעוצרים בk=1). קל לראות שמדובר

בזמן ריצה סופר-פולינומיאלי שכן לכל קבוע c החל מ n0 = 2**c בוודאי מתקיימת ההגדרה.

<u>סעיף ד׳</u>

ii. הסיבוכיות היא O(n**2) כיוון שרשימה היא mutable, ולכן בכל פעם שאנו מאפסים תא באחת מתתי הרשימות של A הוא מתעדכן לכל הקריאות הרקורסיביות הבאות (שבאות אחריו בסדר הקריאות בעץ). לכן, לכל היותר יכולות להיות n**2 קריאות רקורסיביות, (אחרי n**2 קריאות לבטח A מאופסת ולכן לא יהיו קריאות נוספות).

שאלה 3

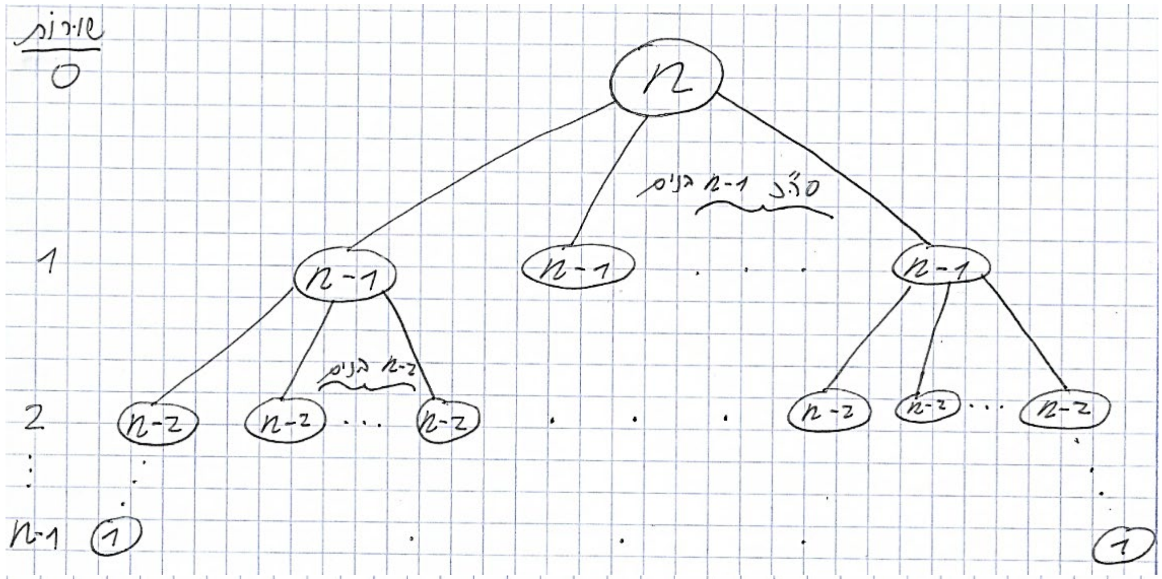
<u>סעיף א׳</u>

בעץ הרקורסיה לכל צומת יש שני בנים (קריאה לפונקציה כשחיברנו את האיבר הראשון, וקריאה נוספת כשחיסרנו) והעבודה שנעשית בצומת היא מסיבוכיות O(1) (נדגיש ש pop(), append() מסיבוכיות של O(1)). עומק העץ הוא n, כאורך הרשימה (בכל שורה מתבצעות קריאות עם איבר אחד פחות מהרשימה בשורה מעל). סה״כ 2**n קריאות, ולכן סיבוכיות של O(2**n).

<u>סעיף ב׳</u>

בעץ הרקורסיה לכל צומת יש חמישה בנים (קריאה לפונקציה כשחיברנו את האיבר הראשון, קריאה נוספת שחיסרנו, קריאה שחיברנו פעמיים, קריאה שחיסרנו פעמיים וקריאה שלא השתמשנו בו כלל) והעבודה שנעשית בצומת היא מסיבוכיות O(1). עומק העץ הוא n, כאורך הרשימה (בכך שורה מתבצעות קריאות עם איבר אחד פחות מהרשימה בשורה מעל). סה״כ 5**n קריאות, ולכן סיבוכיות של O(5**n).

<u>סעיף ג׳</u>



לכל שורה k, $1 \le k \le n-1$, ישנם $(n-1)(n-2)\cdots(n-k)$ צמתים. אורך הרשימה שמקבלת צומת בשורה ה k, היא $n-k$. בכל צומת מבצעים slicing באורך מקסימלי של $n-k$ וחיבור רשימות בסיבוכיות $O(n-k)$ לצורך החישוב (נניח בה"כ שהקבוע הוא 1 וחיבור הרשימות שווה ערך ל $n-k$ פעולות) כ $n-k-1$ פעמים. אזי סך הפעולות בערך:

$$\sum_{k=0}^{n-1} (n-1)(n-2)\cdots(n-k)\cdot(n-k)(n-k)\cdot(n-k-1)$$

נוכיח $O(n!)$: נכפיל ונחלק ב $n!$:

$$= n!\sum_{k=0}^{n-1} \frac{(n-1)\cdots(n-k)^3(n-k-1)}{n!} = n!\sum_{k=0}^{n-1}\frac{1}{n}\cdot\frac{(n-k)^2}{(n-k-2)!}$$

$$\le n!\sum_{k=0}^{n-1}\frac{1}{n-k-1}\cdot\frac{(n-k)^2}{(n-k-2)!} = n!\sum_{k=0}^{n-1}\frac{(n-k)^2}{(n-k-1)!} = n!\sum_{k=0}^{n-1}\frac{(n-k)^3}{(n-k)!}$$

$$= n!\sum_{k=1}^{n}\frac{k^3}{k!} \le C_k\cdot n! \quad C_k\in\mathbb{R}$$

שאלה 5

נבחן את כמות הפעולות שמתבצעות באלגוריתם הכפל של שני מספרים A,B כך, ש n_A, n_B מספרי הביטים של A,B בהתאמה. מתבצעות בדיוק $n_A \cdot n_B$ פעולות עבור כפל הביטים ולערך $n_A \cdot n_B$ פעולות עבור חיבור הביטים (נראה בהמשך (*) שלכמות הפעולות המדויקת אין השפעה על סיבוכיות התכנית שנחשב). למכפלה n_A+n_B ביטים.

יהיו a,b מספרים כך, ש n מספר הביטים ב a ו m מספר הביטים ב b.

נשים לב שמספר פעולות הכפל המירבי ב power2 (ומכאן באופן ישיר גם הכפל של המספרים הארוכים ביותר) מתקבל אם בכל איטרציה בלולאת ה while נקבל ש b אי-זוגי. אכן קורה עבור מספרים מהצורה $b=2^k-1$.

יהי $0 \le i \le \lfloor \log_2 b \rfloor$ מספר האיטרציה בלולאת ה while (מתחילים לספור מ0). עבור $i=0$ מספר הביטים של a בתחילת הלולאה הוא n, ובסופה $2n$ $(2n=n+n \Leftarrow a=a\cdot a)$ ומספר הביטים של result בתחילת הלולאה הוא 1 ובסופה n $(result=1\cdot a)$. מאינדוקציה פשוטה, באיטרציה ה-i מספר הביטים של a בתחילת הלולאה הוא $2^i \cdot n$ ושל result הוא $n\cdot(2^i-1)$, שנתייחס אליו כאל $n\cdot 2^i$ מחוסר השפעה על הסיבוכיות הסופית: אז הפקודות $a=a\cdot a$, $result=result\cdot a$

גורמות $\underbrace{2}_{(*)}\cdot(2^i n)^2$ פעולות כל אחת, אז סך כל הפעולות:

$$\sum_{i=0}^{\lfloor \log_2 b\rfloor} 2\cdot \underbrace{2}_{(*)}\cdot(2^i\cdot n)^2 = \sum_{i=0}^{m} 4\cdot(2^2)^i\cdot n^2 = 4n^2\cdot\sum_{i=0}^{m} 4^i$$

סכום סדרה הנדסית

$$\stackrel{\uparrow}{=} 4\cdot n^2\cdot \frac{1\cdot 4^{m+1}-1}{3} = \frac{4}{3}\cdot n^2(4^{m+1}-1) \le \overbrace{\frac{4}{3}}^{(*)}\cdot n^2\cdot 4^{m+1}$$

ולכן הסיבוכיות של power2 היא $O(4^{m+1}\cdot n^2)$.