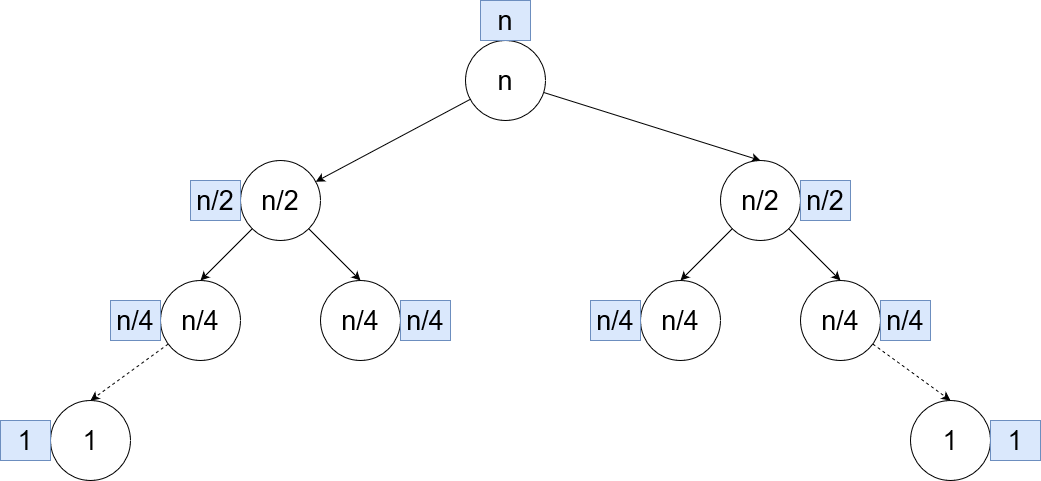
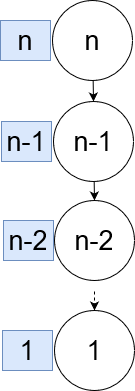
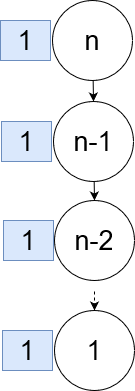
שאלה 1

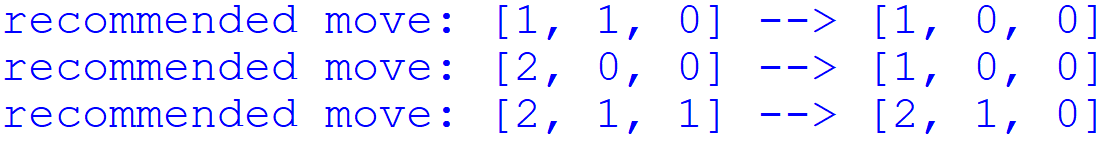
**סעיף א'**

**1. בעץ הרקורסיה הנ"ל כל צמת מייצג קריאה רקורסיבית לmax\_v1. בתוך הצומת מצוין אורך הרשימה ולצדו מצוינת כמות העבודה שמתבצעת. בכל צומת מתבצע slicing פעמיים, כך שבכל פעם אורך הslice הוא חצי אורך הרשימה. סך אורך הslice 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)).**

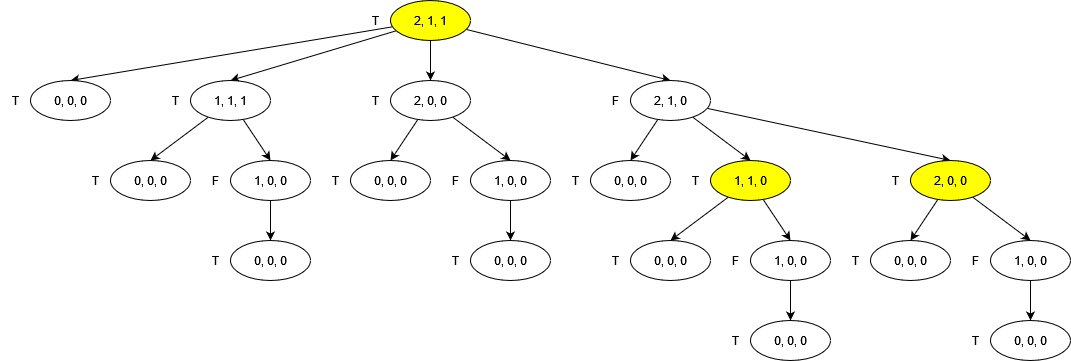
**2. הפעם בכל צומת מתבצע slicing יחיד באורך n-1 ולכן כמות העבודה בכל צומת היא אורך הרשימה. כמו כן בכל צומת מתבצעת קריאה רקורסיבית על הרשימה פחות האיבר הראשון בה. סה"כ בעץ ישנן n שורות. נסכום את סך כמות העבודה בעץ (סכום סדרה חשבונית) ונקבל 0.5\*(n\*\*2 +n). על כן קיבלנו סיבוכיות של O(n\*\*2).**

**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).**

**סעיף ג'**

**i.** יודפס: 

**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).**

****

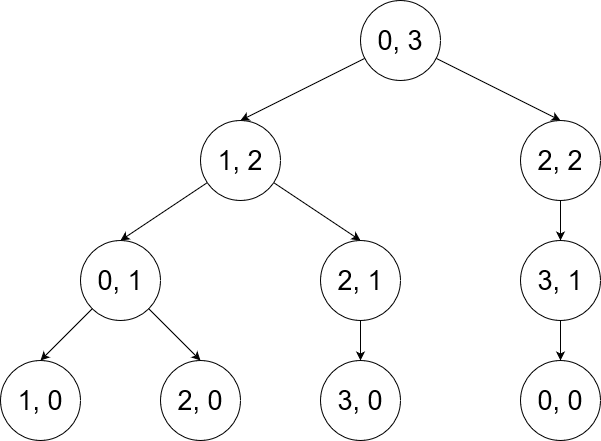
**iii. (בספירה, חיפוש מסוג "if d[key] == val" נחשב הצלחה באופן אוטומטי כיוון שכל חיפוש כזה מחזיר ערך שחושב קודם לכן)**

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

שאלה 2

**סעיף ב'**

**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 תנאי העצירה כמובן), כיוון שלכל תת קריאה הפלט יהיה False (לא ניתן להגיע לs=n כיוון שאף נקודה לא מתחברת לנקודה n).

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]]
* s=0, t=4, k=3

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]
* s=0, t=n, k=n

נקבל שהסיבוכיות היא O(n\*\*log(n)), כיוון שבעץ הרקורסיה נקבל שלכל צומת יש לפחות n בנים (עוברים על כל הנקודות כיוון שבכל נק' הקריאה הראשונה היא False ולכן התנאי כולו הוא False ומתקדמים לנק' הבאה) ועומק הרקורסיה הוא log(n) (בכל שורה מחלקים את k ב 2). קל לראות שמדובר בזמן ריצה סופר-פולינומיאלי שכן לכל קבוע c החל מ n0 = 2\*\*c בוודאי מתקיימת ההגדרה.

סעיף ד'

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**

סעיף א'

בעץ הרקורסיה לכל צומת יש שני בנים (קריאה לפונקציה כשחיברנו את האיבר הראשון, וקריאה נוספת שחיסרנו) והעבודה שנעשית בצומת היא מסיבוכיות O(1) (נדגיש שpop(), 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).

סעיף ב'

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).

סעיף ג'

