**שאלות פתוחות**

תרגיל 1:­­

הראו כיצד ניתן לממש תור באמצעות שתי מחסניות. נתחו את זמן הריצה של הפעולות על התור.

תרגיל 2:

הראו כיצד ניתן לממש 2 מחסניות באמצעות רשימה A בגודל n. ניתן להניח שסכומי גודלי המחסניות אינו עולה על n. על פעולות push and pop להתבצע בסיבוכיות O(1).

תרגיל 3:

נתון שלכל צומת בעץ מתקיים: בן שמאלי קטן ממנו ובן ימני גדול ממנו. האם העץ בהכרח עץ חיפוש בינארי? במידה וכן הסבירו למה. במידה ולא, תתנו דוגמה נגדית.

תרגיל 4:

יהי T **עץ בינארי**.

קבעו עבור כל אחת מהטענות הבאות האם היא נכונה או לא נכונה.

1. בהינתן סריקת pre-order של T, ניתן לשחזר את העץ T באופן יחיד.
2. בהינתן סריקת in-order של T, ניתן לשחזר את העץ T באופן יחיד.
3. בהינתן סריקת post-order של T, ניתן לשחזר את העץ T באופן יחיד.
4. בהינתן סריקת in-order של T  וסריקה pre-order, ניתן לשחזר את העץ T באופן יחיד.

תתנו הסבר ודוגמא לכל סעיף.

תרגיל 5:

יהי T **עץ חיפוש בינארי.**

הראו כי בהינתן סריקת pre-order של T, ניתן לשחזר את העץ T באופן יחיד.

תרגיל 6:

בכיתה  למדנו איך למצוא את העוקב של צומת x בעץ חיפוש בינארי T שיש לו n צמתים – הפעולה Successor(x).

אם נתחיל מהאיבר המינימלי בעץ ונבצע n פעמים פעולת Successor. מהו זמן הריצה של סדרת פעולות זו?

תרגיל 7:

נתון עץ AVL ריק. מבצעים עליו n פעולות insert, כאשר בפעולת insert נתון רק מצביע לשורש העץ והאיבר החדש.

1. מהו זמן הריצה הכולל במקרה הגרוע? (חסם הדוק)
2. מהו מספר ה rotations במקרה הגרוע? (חסם עליון)

תרגיל 8:

תארו אלגוריתם לינארי לבניית עץ AVL ממערך ממוין.

תרגיל 9:

נתונים שני עצי AVL, כל אחד בעל n צמתים.

תארו אלגוריתם יעיל למיזוג שני העצים לעץ AVL אחד.

תרגיל 10:

נתונים k מערכים ממויינים L1, … , Lk המכילים בסך הכול n איברים. תארו אלגוריתם הממזג אותם למערך ממוין מגודל n בסיבוכיות O(nlogk).

תרגיל 11:

ממשו מילון ללא חזרות התומך גם בפעולה הבאה: less(D, x)

כלומר החזרת סכום המפתחות שקטנים / שווים למפתח של האיבר ה-x.

(רמז: קשור לעצי AVL).

תרגיל 12:

נתון מערך של מספרים שלמים.

אנחנו רוצים למצוא האם יש זוג איברים במערך שסכומם הוא 11 בזמן ריצה O(n).

אם נתון שהמספרים הם גם חיובים, איך נעשה זאת בזמן ריצה O(n) במקרה הגרוע?

תרגיל 13:

1. האם מערך ממוין מייצג ערימה בינארית? ממוין = סדר מונוטוני לא יורד.
2. האם ערימה בינארית מיוצגת תמיד במערך ממוין?

תרגיל 14:

תאר/י מימוש של מבנה הנתונים Sשבו הפעולה , Max לוקחת O(1) זמן במקרה הגרוע. הפעולות

Init(L) - אתחל את מבנה הנתונים עם n איברים הנמצאים ברשימה L.

Insert(x, S)- מוסיף איבר למבנה הנתונים

Delete(x, S)  - מוחק איבר ממבנה הנתונים (ניתן להניח ש-x נמצא במבנה הנתונים לפני הפעולה delete)

- Min(S)מחזיר את האיבר הקטן ביותר ממבנה הנתונים.

Max(S)- מחזיר את האיבר הגדול ביותר ממבנה הנתונים.

הפעולות צריכות להיות יעילות ככל האפשר במקרה הגרוע.

סיבוכיות של Insert : \_\_\_\_\_\_\_\_\_\_

סיבוכיות של Delete : \_\_\_\_\_\_\_\_\_\_

סיבוכיות של init : \_\_\_\_\_\_\_\_\_\_

סיבוכיות של Max : \_\_\_\_\_\_\_\_\_\_

סיבוכיות של Min : \_\_\_\_\_\_\_\_\_\_

תרגיל 15:

האם את הפעולות Min / Max מתרגיל 13 ניתן לפתור בעזרת ערימה אחת?

תרגיל 16:

נתון מערך לא ממוין בגודל n. תארו אלגוריתם המוצא את k האיברים הקטנים במערך בזמן O(n).

(בונוס: אפשר לפתור באמצעות ערימות ב(-O(n + k log k)

תרגיל 17:

נתון לנו מערך בגודל n של מספרים. אנחנו יודעים שקיים במערך איבר שחוזר לפחות n/5 פעמים. איך נמצא אותו בזמן לינארי?

תרגיל 18:

בהינתן קבוצה S של n  מספרים ייחודיים וחיוביים, תארו אלגוריתם לינארי שמוצא את k המספרים הקרובים ביותר לממוצע של S .

**שאלות סגורות:**

סיבוכיות זמן ריצה של הוספה של איבר לסוף רשימה (רשימה רגילה של פייתון) היא:

1. תמיד O(1)
2. O(1) בממוצע
3. O(n)
4. תלוי בגודל הרשימה
5. לא ניתן להוסיף איברים לרשימה

קיבלנו string ריק ורשימה של תווים בגודל n. ברצוננו להוסיף את כל התווים לstring אחד אחרי השני בעזרת string concatenation. מה יהיה זמן הריצה?

1. O(n)
2. O(nlogn)
3. O(n^2)
4. O(1)

קיבלנו node הנקרא head המייצג את הרשימה 1->2->3->4->5. את מבצעת הפעולה הבאה:

Head.next.next.next = None

מה תהיה הרשימה עכשיו?

1. 1->2->3->5
2. 1->2->3
3. 1->2->3->4

מה החיסרון המרכזי של רשימות מקושרות על פני רשימות רגילות (מערך)

1. צריכים מצביע על מנת לבצע remove בזמן קבוע.
2. ביצוע מעבר על רשימה מקושרת יקר יותר.
3. לרשימות מקשורות אין גודל קבוע
4. אין אפשרות לבצע random access

מה מהבאים ניתן לאחסן כ-key ב-hash table?

1. רשימה
2. Integer
3. Mutable string
4. Immutable string
5. Boolean
6. Float

מה מהבאים ניתן לאחסן כ-value ב-hash table?

1. רשימה
2. Integer
3. Mutable string
4. Immutable string
5. Boolean
6. Float

נניח יש לך hash function נוראית שתמיד מחזירה את המספר 1337. התנגשויות מטופלות בשיטת chaining. מהו זמן ריצה הממוצע הנדרש עבור בדיקה ש-key קיים עבור ה-hash table שלך?

1. O(n)
2. O(1)
3. O(n) אבל רק אם את מחפשת באיברים שנוספו אחרונים
4. O(n^2)

את צריכה לממש את ה- hash table הפרטית שלך. מה כדאי לעשות? בשאלה זו ניתן לבחור כמה תשובות.

1. לקבוע initial size עצום על מנת להימנע מ- resizing.
2. ב-hash function להתשמש במודולו של מספרים ראשוניים
3. לממש מתודה שמטפלת ב-collisions.
4. להשתמש ב-set ולא ב-list לאיחסון values
5. להחליף זוגות של key-value ישנים כאשר מתרחש collision על מנת תמיד לתחק פעולות ב-O(1)

עבור גודל קלט קטן רשימה יכולה להיות ״מהירה״ יותר משימוש ב-hash table. מדוע?

1. שאלה טריקית – hash map הוא O(1) ורשימות הן O(n). ולכן hash table ״מהירות״ יותר תמיד.
2. כי hash table לא O(1) בממוצע כאשר יש collisions.
3. יש תקורת זמן שקשורה ל-hash table בגלל ה-hash function.
4. כיוון שאם הרשימה קטנה אז העולות הן ב-O(1)

מחסניות ותורים ידועים גם כ:

1. FIFO LIFO
2. LILO and FILO
3. LIFO and LILO
4. LIFO and FIFO

מחסניות ותורים מוגדרים ע״י:

1. הבעיה שאותה את מנסה לפתור
2. ה-interface שלהם – מהן הפעולות שניתן לבצע איתן
3. ה-implementattion שלהם – אם משתמשים ברשימה או רשימה מקושרת
4. היעולות שלהם – אם הפעולות לוקחות זמן קבוע

מה מהבאים היא פעולה על מחסנית? ניתן לבחור ביותר מתשובה אחת.

1. Remove
2. Pop
3. Append
4. Push
5. Peek

מה מהבאים פעולה על תור? ניתן לבחור ביותר מתשובה אחת.

1. Enqueue
2. Append
3. Dequeue
4. Push
5. Pop