אוניברסיטת ת"א
סמסטר ב', תשפ"ה
הגשה עד 09/06

תרגיל בית 4 במבני נתונים

על כל התשובות להיות מנומקות. בכל שאלה יש לבחור במימוש היעיל ביותר האפשרי מבחינת סיבוכיות זמן. יש לענות על השאלות במקומות המוגדרים לכך.

שאלה 1

למבנה B-tree יש פרמטר d, שמגדיר את גודל הצמתים שבעץ. למשל עבור עץ 2-4 מתקיים d=2.
נגדיר גובה עץ = מרחק בין שורש לעלה (עץ עם שורש בלבד בגובה 0).

א.נותנים לנו 70,000,000 איברים. אנחנו רוצים לאחסן אותם במבנה B-tree, כך שגובה העץ לא יהיה יותר מ-7. מהו הערך הקטן ביותר של d שעומד בתנאי הנ"ל?

ב.חזרו על הסעיף הקודם, כאשר לא נתונים לנו האיברים מראש. אנחנו מתחילים מעץ ריק, מבצעים פעולות הכנסה והוצאה, כך שלעולם לא יהיו בעץ יותר מ-70,000,000 איברים וגובה העץ לא יהיה יותר מ-7.

~~שאלה 2~~

הוכיחו חסם אמורטייזד O(1) על מספר פעולות split/fuse במהלך הכנסות ומחיקות מלמטה למעלה בעץ (d,2d).

הדרכה: ניתן להשתמש בפונקצית פוטנציאל שסופרת α פעמים את מספר הצמתים המינימליים ועוד β פעמים את מספר הצמתים המקסימליים, כאשר $\alpha, \beta > 0$ קבועים שאותם תגדירו.

שאלה 3

הציעו מבנה נתונים למימוש טיפוס הנתונים המופשט (ADT) הבא. הניחו כי לכל האיברים יש מפתח שהוא ערך מספרי כלשהו. כל דרישות הסיבוכיות הן amortized. לשם פשטות, מותר להניח כי בשום שלב לא יהיו במבנה הנתונים שני איברים בעלי מפתחות שווים (ואין צורך לבדוק זאת).

- $\mathrm{Insert}(x)$ – הכנסת האיבר x למבנה. סיבוכיות: $O(1)$.
- $\mathrm{DeleteMin}()$ – מחיקת איבר בעל מפתח מינימלי מהמבנה. סיבוכיות: $O(\log n)$.
- $\mathrm{DecreaseKey}(x, \Delta)$ – הקטנת המפתח של האיבר x (נתון מצביע אליו) ב- $\Delta > 0$. סיבוכיות: $O(1)$.
- $\mathrm{Print100}()$ – הדפסת 100 האיברים הכי קטנים במבנה בסדר ממוין. סיבוכיות: $O(1)$.

~~שאלה 4~~

בתרגול דיברנו על ההגדרה של עץ חיפוש בינארי ושלא מספיק רק לדרוש שהמפתח של כל צומת גדול מבנו השמאלי וקטן מבנו הימני. נאמר שעץ בינארי הוא **עץ נחמד** אם הוא מקיים את התכונה החלשה הזאת. הציגו אלגוריתם לינארי המקבל מערך A ובונה ממנו עץ נחמד מאוזן (כלומר גובה לוגריתמי).
רמז: דרך אחת לבנות עץ כזה היא לבצע תהליך דומה ל *build-heap* של ערימות בינאריות.

~~שאלה 5~~

ענו על השאלה הבאה עבור כל אחד מסוגי הערימה שנלמדו. נוסיף לערימה את הפעולה $WhereToInsert(x)$ שמקבלת מפתח x, ומחזירה באיזה מקום בערימה האיבר היה מוכנס, אם הוא היה מוכנס(אילו היינו מבצעים קריאה ל*insert(x)*).

תארו מימוש לפונקציה זאת עבור:

א. ערימה בינארית (הפונקציה תחזיר את האינדקס במערך שבו היה יושב x בתום פעולת

ההכנסה), עם זמן הריצה $O(loglog(n))$

ב. ערימה בינומית (הפונקציה תחזיר את דרגת העץ אליו היה מוכנס x בתום פעולת ההכנסה),

עם זמן ריצה $O(\log n)$

ג. ערימת פיבונאצ'י (הפונקציה תחזיר את דרגת העץ אליו היה מוכנס x בתום פעולת ההכנסה),

עם זמן ריצה $O(1)$.

* שימו לב כי הפעולה אינה משנה את הערימה, אלא רק מדווחת מה היה קורה אילו הערך היה מוכנס.

שאלה 6

השאלה מתייחסת לערמה בינומית עצלה.

בשאלה הזו נדון במימוש חלופי ל-successive linking במקום המימוש שלמדנו בכיתה: עוברים על כל העצים ושמים כל עץ בתא שלו לפי הדרגה. אם בתא יש כבר עץ, מבצעים linking ומעבירים את התוצר ישירות לרשימה הסופית - במקום לתא הבא (כלומר לא נבצע עליו יותר link אפילו אם ניתקל בהמשך בעץ בעל אותה דרגה). בסוף התהליך מעבירים גם עצים שלא בוצע עליהם linking לתוצר הסופי.

א. נכניס לערמה בינומית את האיברים 1,2,3,4,….,n. לאחר מכן נבצע שתי פעולות delete-min בזו אחר זו.

מהי סיבוכיות זמן הריצה של כל אחת מפעולות delete-min אלו כאשר משתמשים במימוש החלופי של successive-linking (ומימוש שאר הפעולות כפי שהוגדר בכיתה)?

מהי סיבוכיות זמן הריצה של כל אחת מהפעולות עם successive-linking הממומש בדרך המקורית?

ב. בכיתה הוכחנו חסמי אמורטייזד אסימפטוטיים לפעולות delete-min,insert,meld,find-min כאשר successive-linking ממומש בדרך המקורית. הוכיחו את אותם חסמי אמורטייזד אסימפטוטיים לפעולות הנ"ל כאשר משתמשים במימוש החלופי של successive-linking (ומימוש שאר הפעולות נשאר כפי שהוגדר בכיתה).