אוניברסיטת תל אביב

סמסטר א' תשפ"ב

מבני נתונים – פרויקט מספר 1 – עץ דרגות

הקדמה:

בתרגיל זה שני חלקים:

א. חלק המעשי: מימוש של עץ AVL. עמודים 1-2 במסמך זה מתארים את החלק הזה.

ב. חלק ניסויי-תיאורטי: בהתבסס על המימוש מהחלק המעשי, נבצע מספר "ניסויים" עם ניתוח תיאורטי נלווה. עמודים 3-4 מתארים את החלק הזה.

שימו לב: בסוף המסמך (עמוד 4) ישנן הוראות הגשה – הקפידו לפעול לפיהן. **תאריך הגשה: 9.12.2021**.

חלק מעשי

דרישות

בתרגיל זה יש לממש עץ AVL, לפי ההגדרות שניתנו בכיתה. לכל איבר בעץ יש ערך (info) מסוג מחרוזת (String), ומפתח (key) שהוא מספר שלם (int). כל המפתחות שונים זה מזה, והסדר על צמתי העץ מתייחס כרגיל אך ורק למפתחות. המימוש יהיה **בשפת ג'אווה וצריך להיות מבוסס על קובץ השלד המופיע באתר הקורס**. הפעולות שיש לממש הן:

פעולה	תיאור
empty()	הפונקציה מחזירה ערך TRUE אם ורק אם העץ ריק.
search(int k)	הפונקציה מחפשת איבר בעל המפתח k. אם קיים איבר כזה, היא מחזירה את הערך השמור עבורו, אחרת היא מחזירה null.
insert(int k, String s)	הכנסת איבר בעל ערך s ומפתח k לעץ, אם המפתח לא קיים. הפונקציה מחזירה את מספר פעולות האיזון שנדרשו בסה"כ בשלב תיקון העץ על מנת להשלים את הפעולה (גלגולי LR ו- RL נחשבים ל-2 פעולות איזון). אם קיים איבר בעל מפתח k בעץ הפונקציה מחזירה 1- ולא מתבצעת הכנסה.
delete(int k)	מחיקת איבר בעל המפתח k בעץ, אם הוא קיים. הפונקציה מחזירה את מספר פעולות האיזון שנדרשו בסך-הכל בשלב תיקון העץ על מנת להשלים את הפעולה. אם לא קיים איבר בעל המפתח k בעץ הפונקציה מחזירה 1-.
min()	מחזירה את ערך (info) האיבר בעץ בעל המפתח המינימלי, או null בעץ ריק.
max()	מחזירה את ערך (info) האיבר בעץ בעל המפתח המקסימלי, או null בעץ ריק.
keysToArray()	הפונקציה מחזירה מערך ממוין המכיל את כל המפתחות בעץ, או מערך ריק אם העץ ריק.
infoToArray()	הפונקציה מחזירה מערך מחרוזות המכיל את כל המחרוזות בעץ, ממוינות על פי סדר המפתחות. כלומר הערך ה j במערך הוא המחרוזת המתאימה למפתח שיופיע במיקום ה j במערך הפלט של הפונקציה keysToArray(). גם הפונקציה הזאת מחזירה מערך ריק אם העץ ריק.
size()	הפונקציה מחזירה את מספר האיברים בעץ.
split(int x)	הפונקציה מקבלת מפתח x שנמצא בעץ. על הפונקציה להפריד את העץ ל-2 עצי AVL כאשר המפתחות של האחד גדולים מ-x ושל השני קטנים מ-x. יש לממש את הפונקציה על פי המימוש שנלמד בהרצאה בסיבוכיות $O(\log n)$.
join(IAVLNode x, AVLtree t)	הפונקציה מקבלת צומת x ועץ t שכל ה-keys שלהם קטנים, או שכולם גדולים, מה-keys של העץ הנוכחי שביחס אליו קראנו ל-join. על הפונקציה לאחד את x,t לעץ הנוכחי כפי שמומש בהרצאה. על הפעולה לרוץ בזמן $O(\log n)$. על הפעולה להחזיר את העלות של פעולת ה-join (הפרש גבהי העצים+1).
getRoot()	הפונקציה מחזירה מחזירה את השורש של העץ (אובייקט AVLNode)

בנוסף למימוש הפונקציות האלו, יש לממש את מחלקת AVLNode כפי שמתואר בקובץ. ניתן להוסיף מחלקות נוספות, אך כל מחלקה שמייצגת צומת בעץ צריכה לממש את הIAVLNode interface. מטעמי נוחות, נדרוש שלכל עלה יהיו 2 בנים "וירטואליים", כלומר, צמתים ללא מפתח. באופן זה, נוח יותר לממש גלגולים מכיוון שלכל צומת יהיו 2 בנים.

למחלקה AVLNode יש את הפונקציות הבאות (המפרט המלא נמצא בקובץ השלד):
getKey – מחזיר את המפתח של הצומת, או 1- אם הצומת הוא וירטואלי.
getValue – מחזיר את הinfo של הצומת או null אם הצומת הוא וירטואלי.
getLeft – מחזיר את הבן השמאלי של הצומת, או null אם אין כזה.
getRight – מחזיר את הבן הימני של הצומת, או null אם אין כזה.
isRealNode – מחזיר TRUE אם הצומת מייצג צומת אמיתי בעץ (קרי: צומת שאינו וירטואלי).
getHeight – מחזיר את גובה הצומת, 1- עבור צומת וירטואלי. יש למשש בסיבוכיות O(1).

<u>הערות חשובות:</u>

1. בקובץ השלד מופיעים ה header ים של כל הפונקציות. **המימוש יבוצע על ידי מילוי קובץ השלד. במידת הצורך, ניתן להרחיב את המימוש** (למשל להוסיף פונקציות עזר שאינן מופיעות בשלד), אך **אסור לשנות את הגדרות הפונקציות לעיל**. על כל הפונקציות/מחלקות להופיע בקובץ יחיד.
2. **אין להשתמש באף מימוש ספרייה של מבנה נתונים.**

סיבוכיות

יש לתעד בקוד ובמסמך נפרד (ביותר פירוט) את סיבוכיות זמן הריצה במקרה הגרוע (האסימפטוטית, במונחי O הדוקים) של כל פונקציה, כתלות במספר האיברים בעץ n. עליכם להשיג סיבוכיות זמן ריצה (במקרה הגרוע ביותר) נמוכה ככל הניתן עבור כל אחת מהפונקציות.

פלט

אין צורך בפלט למשתמש.

תיעוד

בנוסף לבדיקות אוטומטיות של הקוד שיוגש, קובץ המקור ייבדק גם באופן ידני. חשוב להקפיד על תיעוד לכל פונקציה, וכמות סבירה של הערות. **הקוד צריך להיות קריא**, בפרט הקפידו על בחירת שמות משתנים ועל אורך השורות.
יש להגיש בנוסף לקוד גם מסמך תיעוד חיצוני. המסמך יכלול את תיאור המחלקה שמומשה, ואת תפקידו של כל חבר במחלקה. עבור כל פונקציה במחלקה יש לפרט מה היא עושה, כיצד היא פועלת **ומה סיבוכיות זמן הריצה שלה**. בפרט, אם פונקציה קוראת לפונקציית עזר, **יש** להתייחס גם לפונקציית העזר בניתוח.

בדיקות

התרגילים ייבדקו באמצעות תוכנת טסטר שקוראת לפונקציות המפורטות מעלה בתרחישים שונים, ומוודאת את נכונות התוצאות. קובץ הטסטר שלנו **לא יפורסם** לפני הבדיקות.
מומלץ מאוד לממש אוסף בדיקות עבור המימוש, לא בשביל ההגשה, אלא כדי לבדוק שהקוד לא רק מתקמפל, אלא גם נכון!

בקובץ שתגישו **לא תהיה פונקציית main**, דבר זה יפגע בטסטר שיבדוק לכם את התרגילים. אם הפרויקט מתקמפל לבדו (ללא טסטר), זה סימן שמשהו לא נכון במימוש.

הקוד ייבדק על מחשבי בית הספר על גירסא Java8.

הנחיות להשמשת סביבת העבודה בבית (ג'אווה+אקליפס):
http://courses.cs.tau.ac.il/software1/1415b/misc/workenv.pdf
מדריך לעבודה עם Eclipse (סעיפים 9-5, 15):
http://www.vogella.com/
הנחיות לפתיחת חשבון מחשב, למי שמעוניינ/ת לעבוד במעבדת בית הספר:
http://cs.tau.ac.il/system/accounts0
שימוש בג'אווה 8 במעבדות האוניברסיטה:
http://courses.cs.tau.ac.il/software1/1415b/misc/lab-eclipse.pdf

חלק ניסויי/תיאורטי

שאלה 1:

בשאלה זו נדון ב insertion-sort באמצעות עצי AVL. אופן המיון מתבצע באופן הבא: מכניסים את האיברים לפי הסדר (הלא ממוין) אל העץ, ובסיום מבצעים סריקת in-order לקבלת הסדר הממוין.
<u>שימו לב:</u> לצורך השאלה הזאת, פעולת ה search הראשונית המתבצעת בהכנסה לעץ AVL תמומש בשיטת ה- finger-tree כאשר החיפוש יתחיל מהאיבר **המקסימלי** בעץ (ולא מהשורש!).

- לצורך הניתוח, נבנה עצי AVL בגדלים שונים. מספר איברים שנכניס לעץ יהיה n=1000*(2^i) כאשר i=1,...,5. כלומר, עבור i=1 העץ בגודל 2000, ועבור i=5 העץ בגודל 32,000.
- לכל גודל של עץ, נבצע 2 ניסויים נפרדים:
  - בניסוי אחד סדר האיברים יהיה **ממוין-הפוך**, מגדול לקטן.
  - בניסוי שני סדר האיברים יהיה **אקראי**.

א. עבור כל ניסוי, יש לציין את מספר החילופים במערך המקורי (זוגות $i > j$ כך ש- $a[i] < a[j]$), ואת עלות המיון הכוללת ב-AVL שנגדיר להיות סכום עלויות פעולות ה-search לאורך ההכנסות. את המספרים יש למלא בטבלה:
(נזכיר שעלות search בודד הוא אורך המסלול מהאיבר המקסימלי אל מיקום ההכנסה, כי תצורת העבודה בשאלה היא finger-tree.)

מספר סידורי i	מספר חילופים במערך ממוין-הפוך	עלות החיפושים במיון AVL עבור מערך ממוין-הפוך	מספר חילופים במערך מסודר אקראית	עלות החיפושים במיון AVL עבור מערך מסודר אקראי
1				
2				
...				
5				

ב. נתחו באופן תיאורטי את מספר החילופים **וגם** את עלות החיפושים של ה-AVL במקרה של **מערך ממוין-הפוך**. התשובות צריכות להיות כתלות בגודל המערך n. את מספר החילופים יש לתת באופן מפורש, עלות החיפושים יכולה להיות מתוארת במונחי $\Theta(\cdot)$ (הציגו חסמי $O(\cdot)$ ו- $\Omega(\cdot)$).

ג. האם הערכים בטבלה מסעיף א' והניתוח מסעיף ב' מתאימים? נמקו.
הדרכה: במקרה שמצפים לביטוי ליניארי, ניתן לבדוק את מידת ההתאמה האמפירית של הנתונים על-ידי חישוב קו-מגמה. בתוכנת "אקסל", למשל, ניתן לחשב קו-מגמה ומדד ה-R^2 מעיד על איכות הקירוב. כאשר הקשר אינו ליניארי, נעבד תחילה את הנתונים כדי להעביר אותם לצורה שבה הקשר המצופה יהיה ליניארי (בקואורדינטות החדשות).

ד. נתון מערך שמספר החילופים בו הוא h. למשל, אם $h = 0$ המערך ממוין, ובסעיף ב' מצאנו את h_{max}. הדקו את החסם-העליון על עלות המיון באמצעות עץ ה-AVL במונחי h, n.
כלומר: חזרו על ניתוח הסיבוכיות והציגו <u>חסם-עליון</u> הדוק ככל האפשר לסיבוכיות של insertion-sort באמצעות עצי AVL. על התשובה להיות במונחי $O\big(f(n,h)\big)$ עבור פונקציה $f(n,h)$ מתאימה.
רמז: ניתן "לפרק" את מספר החילופים הגלובאלי h ל"אחריות" של כל איבר i על החילופים הקשורים אליו h_i בעת הכנסתו. נזכיר שמתקיים $h_1 + \cdots + h_n = h$, ונזכיר גם את אי-שוויון הממוצעים.

ה. **<u>בונוס</u> (אין חובת הגשה)**: מדוע בסעיף ד' ביקשנו חסם-עליון בלבד, ולא $\Theta(\cdot)$ כמו בסעיף ב'?

שאלה 2:

בשאלה זו נרצה לנתח את העלות של פעולות ה-join המתרחשות במהלך ביצוע split.

- לצורך הניתוח, נבנה עצי AVL בגדלים שונים. מספר איברים שנכניס לעץ יהיה n=1000*(2^i) כאשר i=1,...,10. כלומר, עבור i=1 העץ בגודל 2000, ועבור i=10 העץ בגודל כמיליון. את האיברים יש להכניס **בסדר אקראי**.
- לכל גודל של עץ, נבצע 2 ניסויים נפרדים (ולכן נבנה שני עצים עם אותו סדר הכנסה, לכל גודל):
  - בניסוי אחד נבצע split על מפתח אקראי בעץ.
  - בניסוי שני נבצע split על המפתח המקסימלי בתת העץ השמאלי של השורש.

א. תעדו בטבלה שלהלן את העלות הממוצעת של פעולות ה-join ואת העלות של פעולת ה-join היקרה ביותר.

מספר סידורי i	עלות join **ממוצע** עבור split **אקראי**	עלות join **מקסימלי** עבור split **אקראי**	עלות join **ממוצע** עבור split של **האיבר** מקסימלי בתת העץ השמאלי	עלות join **מקסימלי** עבור split של **איבר** מקסימלי בתת העץ השמאלי
1				
...				
10				

ב. נתחו באופן תיאורטי את העלות של **join ממוצע לשני התרחישים** (split אקראי או על האיבר המסוים שבחרנו), והסבירו אם התוצאות מתיישבות עם ניתוח הסיבוכיות התאורטי.

ג. נתחו באופן תיאורטי את העלות של **join מקסימלי בתרחיש אחד** של split על האיבר המסוים שבחרנו, והסבירו אם התוצאות מתיישבות עם ניתוח הסיבוכיות התאורטי.

ד. **<u>בונוס</u> (אין חובת הגשה)**: נתחו תאורטית את <u>התוחלת</u> של עלות join מקסימלי עבור split אקראי, בעץ אקראי, במונחי $O(\cdot)$. (האקראיות היא אחידה על-פני בחירת מבנה העץ, ובחירת האיבר בתוך העץ.)

הוראות הגשה

הגשת התרגיל תתבצע באופן אלקטרוני באתר הקורס במודל.

הגשת התרגיל היא בזוגות בלבד!

כל זוג יבחר **נציג/ה** ויעלה **<u>רק</u>** תחת שם המשתמש של הנציג/ה את קבצי התרגיל (תחת קובץ zip) למודל.

על ההגשה לכלול שלושה קבצים:

1. קובץ המקור (הרחבה של קובץ השלד שניתן) תחת השם AVLTree.java.
2. קובץ טקסט info.txt המכיל את פרטי הזוג: מספר ת"ז, שמות, ושמות משתמש.
3. מסמך תיעוד חיצוני, המכיל גם את תוצאות המדידות. את המסמך יש להגיש באחד הפורמטים הבאים: doc, docx או pdf.

שמות קובץ התיעוד וקובץ הzip צריכים לכלול את שמות המשתמש האוניברסיטאיים של **הזוג המגיש** לפי הפורמט AVLTree_username1_username2.pdf/doc/zip/... בתוכן הקבצים יש לציין את שמות המשתמש, תעודות הזהות ושמות המגישים (בכותרת המסמך ובשורת הערה בקובץ המקור).

הגשת שיעורי הבית באיחור - באישור מראש בלבד. הגשה באיחור ללא אישור תגרור הורדת נקודות מהציון.
הגשת התרגיל היא חובה לשם קבלת ציון בקורס.

בהצלחה!