אוניברסיטת תל אביב
סמסטר א' תשפ"א

מבני נתונים - פרויקט מספר 1 - עץ דרגות

דרישות

עליכם לממש עץ AVL, לפי ההגדרות שניתנו בכיתה. לכל איבר בעץ יש ערך (info) מסוג מחרוזת (String), ומפתח (key) שהוא מספר **טבעי**. 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 למשש הן:

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 הוא לא קיים. הפונקציה מחזירה את מספר פעולות האיזון שנדרשו בסה"כ בשלב תיקון העץ על מנת להשלים את הפעולה (גלגולי LR ו- RL נחשבים כ-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 בסה"כ בשלב תיקון העץ על מנת להשלים את הפעולה. אם לא קיים איבר בעל המפתח k בעץ הפונקציה מחזירה 1-.

min() - מחזירה את ערכו (info) של האיבר בעץ בעל המפתח המינימלי, או null אם העץ ריק.

max() - מחזירה את ערכו (info) של האיבר בעץ בעל המפתח המקסימלי, או null אם העץ 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 למשש 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 גדולים) מה-keys של העץ הנוכחי. 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 גבהיי העצים+1).

getRoot() – מחזיר את השורש של העץ (אובייקט AVLNode)

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

לAVLNode יש את הפונקציות הבאות (את המפרט המלא תמצאו 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 כן אם הצומת מייצג צומת אמיתי בעץ (צומת שאינו וירטואלי)
getHeight – מחזיר את גובה הצומת (1- עבור צומת וירטואלי). יש לממש בסיבוכיות O(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.
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 שלכם, 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 מתודה 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, ומוודאת את נכונות התוצאות. קובץ הטסטר שלנו **לא יפורסם** לפני הבדיקות. עליכם לבדוק את המימוש בעצמכם! בפרט, כדאי מאוד לממש טסטר, כדי לבדוק את תקינות ונכונות המימוש.

בקובץ שתגישו לא תהיה פונקציית main (דבר זה יפגע בטסטר שיבדוק לכם את התרגילים). אם הצלחתם לקמפל את הפרוייקט לבדו (ללא טסטר), זה סימן שמשהו לא נכון במימוש שלכם.

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

מדידות

1) בשאלה זאת נדון ב insertion-sort. תתבקשו לממש פעם אחת את האלגוריתם בצורה הסטנדרטית (מיון הכנסה סטדנרטי של מערך) ופעם אחת על ידי הכנסת איברי המערך בזה אחר זה לעץ AVL ולאחר מכן חישוב המערך הממויין על ידי סריקת in-order של העץ שהתקבל.
שימו לב: לצורך השאלה הזאת, פעולת ה search הראשונית המתבצעת בהכנסה לעץ AVL תמומש בשיטת finger-treen כאשר החיפוש יתחיל מהאיבר **המקסימלי** בעץ.

א. בצעו 2 ניסויים. בניסוי הראשון הגרילו מערך אקראי מגדלים n=10000*i (כאשר i=1,...,10) ומיינו אותו ב2 השיטות שתיארנו. בניסוי השני יש לבנות מערך ממויין יורד ולמיין אותו ב2 השיטות שתיארנו. יש לתעד בטבלה את תוצאות הניסויים שביצעתם. עבור מיון רגיל (לא בעזרת AVL) תתבקשו לספור כמה פעולות swap בוצעו ועבור המיון באמצעות AVL תתבקשו רק לציין את עלות פעולות הsearch שביצעתם, כלומר את אורך המסלול בין האיבר המקסימלי למיקום ההכנסה.

מספר סידורי	גודל המערך	כמות החילופים במיון רגיל עבור מערך אקראי	כמות החילופים במיון רגיל עבור מערך ממוין הפוך	עלות החיפושים במיון AVL עבור מערך אקראי	עלות החיפושים במיון AVL עבור מערך ממוין הפוך
1	10,000				
2	20,000				
...					

פרטו מהן התוצאות שציפיתם לקבל בטבלה על סמך החומר התיאורטי שנלמד בכיתה, והאם התוצאות שקיבלתם בפועל תואמות את הציפיות. הסבירו את משמעות המדידות שביצעתם. על התשובות להכיל ניתוח תאורטי במונחי O ולהשתמש בערכים n (גודל המערך) ו-H (כמות החילופים במערך).
ב. הוכיחו חסם אסימפטוטי הדוק ככל האפשר לסיבוכיות של insertion-sort באמצעות עצי AVL. על התשובה להיות במונחי n,H.

2) הכניסו לעץ AVL n=10000*i איברים טבעיים אקראיים (כאשר i=1,...,10). נרצה לנתח את העלות של פעולות ה-join המתרחשות במהלך ביצוע split. נבצע 2 ניסויים. בניסוי האחד נבצע split על מפתח אקראי בעץ, ובניסוי השני נבצע split על המפתח המקסימלי בתת העץ השמאלי של השורש (כלומר עלה). בכל אחד מהניסויים עליכם לתעד בטבלה את העלות הממוצעת של פעולות ה-join ואת העלות של פעולת ה-join היקרה ביותר.

מספר סידורי	עלות join ממוצע עבור split אקראי	עלות join מקסימלי עבור split אקראי	עלות join ממוצע עבור split של איבר מקס בתת העץ השמאלי	עלות join מקסימלי עבור split של איבר מקס בתת העץ השמאלי
1				
2				
...				

פרטו את תוצאות הטבלה כמו בסעיף הקודם, והסבירו האם התוצאות מתיישבות עם הניתוח התאורטי של סיבוכיות הזמן.
בונוס: נתחו תאורטית (במונחי O של) את 2 תוצאות הטבלה במקרה של split של איבר אקראי.

הגשה

הגשת התרגיל תתבצע באופן אלקטרוני באתר הקורס במודל.

הגשת התרגיל היא בזוגות בלבד!

כל זוג ייבחר נציג **אחד** ויעלה תחת שם המשתמש שלו את קבצי התרגיל (תחת קובץ zip) למודל. על ההגשה לכלול שלושה קבצים:

קובץ המקור (הרחבה של קובץ השלד שניתן) תחת השם AVLTree.java.

קובץ טקסט info.txt המכיל את פרטי המגישים הבאים: תז, שמות ושמות משתמש.

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 **שני המגישים** 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!