מעבדה 7. נושא: עץ חיפוש בינארי

תאריך הגשה: 21.12.2021 בשעה 23:00 (בזוגות)

יש לקרוא היטב לפני תחילת העבודה !

מבוא:

במעבדה הנוכחית נממש עצי חיפוש בינאריים.

רקע:

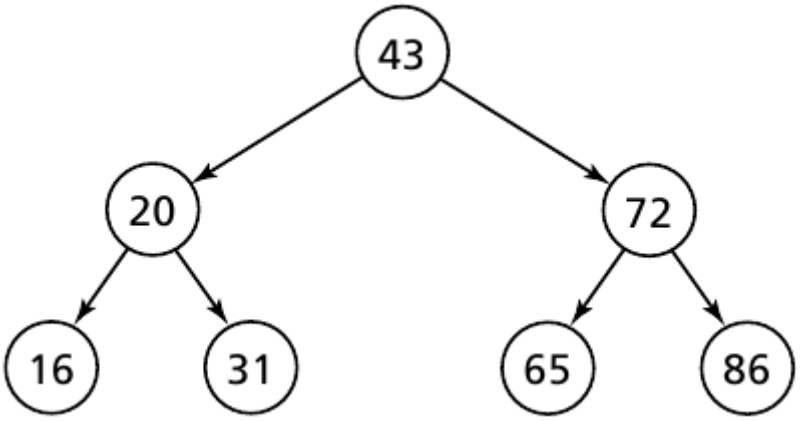
הגדרה1 (עץ בינארי):

עץ שלכל אחד מהצמתים הפנימיים שלו לכל היותר שני בנים, נקרא עץ בינארי

הגדרה 2 (עץ חיפוש בינארי):

עץ בינארי בו מפתחות של כל הצמתים הנמצאים בתת-עץ שמאלי קטנים ממפתח של שורש העץ וכל המפתחות של כל הצמתים הנמצאים בתת-עץ ימני גדולים ממפתח של שורש העץ, נקרא עץ חיפוש בינארי.

דוגמא לעץ חיפוש בינארי:



הממשק הגנרי Comparable (ממשק עם שיטה אחת בלבד)

https://docs.oracle.com/javase/8/docs/api/java/lang/Comparable.html

תאור התרגיל:

1) כתבו מחלקה **Person** המכילה מידע על אנשים.

המחלקה תשב בחבילה: **package** il.ac.telhai.ds.misc

שדות המחלקה:

```
String id;
String firstName;
String lastName;
```

2) הוסיפו למחלקה בנאי המקבל ערכים לשלשת השדות, והוסיפו getters ו setters לשדות, מלבד set ל-id.

3) בנוסף, המחלקה Person תממש את הממשק Comparable<Person>, כאשר הסדר נקבע לקסיקוגרפית לפי id בלבד. שני מופעים של Person בעלי id זהים - יחשבו לאותו ה-Person.

4) הוסיפו מתודות hashCode, equals כפי שניתן להוסיף באופן אוטומטי ב-eclipse (לחצו על source → generate hashCode and equals), לפי השוואת id בלבד.

(שאלה: מדוע חשוב להוסיף מתודות אלו?)

5) הוסיפו מתודת toString ל-Person.

"Person [id=" + #id + ", firstName=" + #firstName + ", lastName=" + #lastName + "]"
כאשר במקום #field רושמים את ערך השדה field.
ניתן להשתמש באפשרות האוטומטית ב-eclipse (לחצו על source → generate toString).

6) העתיקו את המחלקה BinaryTree שרשמתם במעבדה הקודמת לפרוייקט זה.

המחלקה תשב בחבילה: **package** il.ac.telhai.ds.trees

7) השלימו את המחלקה הגנרית **BinarySearchTree** הנתונה לכם, היורשת מהמחלקה הגנרית BinaryTree שנכתבה בתרגיל הקודם.

מחלקה זו נמצאת בחבילה: **package** il.ac.telhai.ds.trees

מחלקה זו מממשת עץ חיפוש, כאשר ההשוואה של האיברים מתבצעת לפי compareTo.

יש לדרוס את המתודות getLeft ו-getRight כך שיחזירו עותק של הבנים. מדוע כ״כ חשוב לבצע זאת באופן הזה?

שימו לב גם שערך ההחזרה של המתודות הללו הוא BinarySearchTree, למרות שהמתודות שנדרסות על ידן מחזירות ערך מטיפוס BinaryTreeI.

חישבו האם יש צורך לדרוס את המתודות setLeft, setRight? ומה לגבי מתודת setValue?

זכרו, חשוב מאד לשמור שמופע של המחלקה **BinarySearchTree** ייצג תמיד עץ חיפוש. חישבו איך זה ישפיע גם על מימוש הבנאי המקבל את תתי העצים - השמאלי והימני.

סדר העבודה ופרטים טכניים

- שליפת הפרויקט DS-Lab07-SearchTree מתוך GITHUB בקישור:
  https://github.com/michalHorovitz/DSLab2021-2022Public
  - אם אין לכם גישה לפרויקט שהורדתם מ GITHUB במעבדות הקודמות יש לבצע שליפה מחדש.
  - אם יש לכם גישה לפרויקט שהורדתם מ GITHUB במעבדה הראשונה אז בצעו:
    - קליק על שם הפרויקט.
    - עכבר ימני
    - Team-->Pull
    - File-->Import->Git->Projects From Git->Existing Local Repository

פורמט קובץ ההגשה ובדיקתו:

פורמט: יש להגיש קובץ ZIP בשם

37_lab07_123456789_987654321.zip

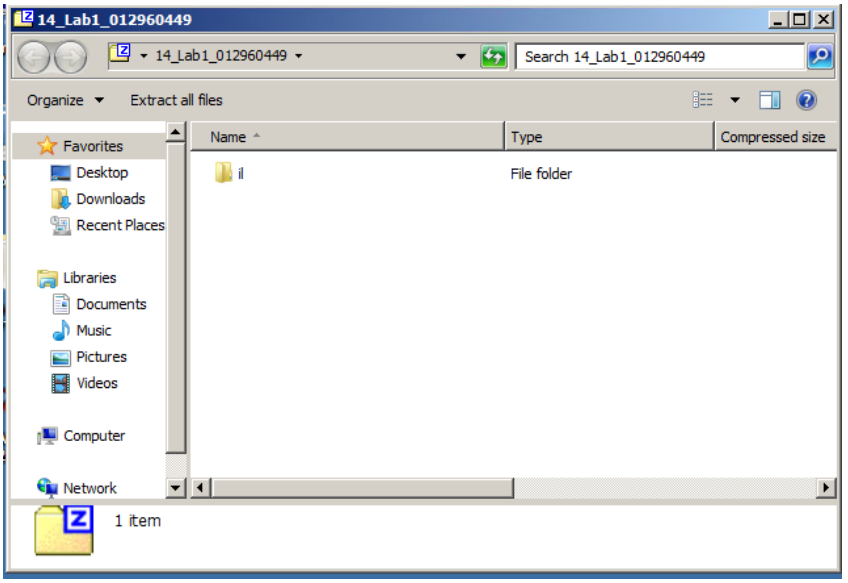
(כמובן, יש להחליף את המספרים עם מספרי ת.ז. של המגישים).

על הקובץ להכיל את כל קבצי ה JAVA שכתבתם כאשר הם נמצאים בתיקייה

il/ac/telhai/ds/trees

כלומר, השורש של קובץ ההגשה יכיל רק תיקייה בשם il.

ומכיל את כל קבצי - java . להמחשה תמונה של קובץ כזה שנפתח ב - WindowsExplorer



<u>בדיקת קובץ ההגשה:</u> בדקו את הקובץ שיצרתם בתוכנת הבדיקה בקישור:

https://cs.telhai.ac.il/homework/

ראו סרטון הדגמה של השימוש בתוכנת הבדיקה.

חשוב !!!

בדיקת ההגשות תבוצע ברובה ע"י תוכנית הבדיקה האוטומטית הנ"ל. תוצאת הבדיקה תהייה בעיקרון זהה לתוצאת הבדיקה הנ"ל שאתם אמורים לערוך בעצמכם . כלומר, אם ביצעתם את הבדיקה באתר החוג, לא תקבלו הפתעות בדיעבד. אחרת, ייתכן שתרגיל שעבדתם עליו קשה ייפסל בגלל פורמט הגשה שגוי וכו'. דבר שהיה ניתן לתקנו בקלות אם הייתם מבצעים את הבדיקה.

היות ואין הפתעות בדיעבד, לא תינתן אפשרות של תיקונים, הגשות חוזרות וכד'.

הגשה שלא מגיעה לשלב הקומפילציה תקבל ציון 0.

הגשה שלא מתקמפלת תקבל ציון נמוך מ- 40 לפי סוג הבעיה.

הגשה שמתקמפלת תקבל ציון 40 ומעלה בהתאם לתוצאות הריצה, ותוצאת הבדיקה הידנית של הקוד (חוץ ממקרה של העתקה).

תכנית הבדיקה האוטומטית מכילה תוכנה חכמה המגלה העתקות. מקרים של העתקות יטופלו בחומרה