**תיעוד מימוש עץ AVL – לירון כהן 207481268 ויובל מור 209011543**

**מתודות AVLTree**

**AVLTree()**

המתודה בונה עץ ריק, כלומר מאתחלת את שורש העץ להיות null ואת גודל העץ להיות 0.

סיבוכיות המתודה היא כיוון שהיא קובעת ערכים של שדות.

**Empty()**

המתודה בודקת האם שורש העץ הוא null. אם כן, מחזירה אמת ואחרת שקר.

סיבוכיות המתודה היא כיוון שהשוואת שורש העץ ל-null נעשית בזמן קבוע.

**Search(int k)**

המתודה מקבלת מספר שלם המייצג מפתח בעץ ומבצעת חיפוש בינארי על צמתי העץ.

בכל שלב, המתודה משווה את מפתח הצומת הנבדק לעומת המפתח שהתקבל כארגומנט.

נתחיל ממצביע לשורש, ונמשיך בחיפוש כל עוד לא הגענו לצומת שהיא null:

אם המפתחות שווים, מחזירה את ערך הצומת.

אם המפתח שהתקבל קטן ממפתח הצומת, מצביע הצומת יעבור לבן השמאלי.

אם המפתח שהתקבל גדול ממפתח הצומת, מצביע הצומת יעבור לבן הימני.

אם לא נמצא המפתח עד לשלב זה, המפתח אינו נמצא בעץ ומוחזר null.

סיבוכיות המתודה היא מכיוון שמדובר בעץ AVL (שהינו עץ חיפוש בינארי מאוזן), עומק העץ הוא ובמקרה הגרוע בכל איטרציה של לולאת ה-while לא מוצאים את המפתח ולכן יורדים רמה בעץ, עד להגעה ל-null.

**Insert(int k, String s)**

* לחשוב האם להוסיף שדה rank, דן ממליץ.
* במקרה שבו המפתח קיים בעץ, לא צריך לעדכן את שדה הinfo במסגרת פעולת הinsert
* לקרוא בפורום את שרשור "ספירת פעולות כחלק מinsert"

**Delete(int k)**

**Min()**

המתודה מתחילה עם מצביע לשורש העץ ויורדת עם המצביע לבן השמאלי כל עוד הבן השמאלי קיים (והוא לא עלה וירטואלי, כדי להגיע לעלה האמיתי הנדרש). המתודה מחזירה את ערך העלה השמאלי ביותר בעץ.

סיבוכיות המתודה היא מכיוון שמדובר בעץ AVL (שהינו עץ חיפוש בינארי מאוזן), עומק העץ הוא וכדי להגיע לעלה השמאלי ביותר בעץ (בעל הערך המינימלי) נצטרך לרדת בלולאה פעמים.

**Max()**

המתודה מתחילה עם מצביע לשורש העץ ויורדת עם המצביע לבן הימני כל עוד הבן הימני קיים (והוא לא עלה וירטואלי, כדי להגיע לעלה האמיתי הנדרש). המתודה מחזירה את ערך העלה הימני ביותר בעץ.

סיבוכיות המתודה היא מכיוון שמדובר בעץ AVL (שהינו עץ חיפוש בינארי מאוזן), עומק העץ הוא וכדי להגיע לעלה הימני ביותר בעץ (בעל הערך המקסימלי) נצטרך לרדת בלולאה פעמים.

**nodesToArray() – מתודת עזר**

המתודה מחזירה מערך צמתים שמכיל את צמתי העץ מסודרים לפי מפתחות בסדר עולה.

המתודה מגדירה מחסנית ששומרת את כל הצמתים שהמצביע עבר בהם.

כל עוד המצביע לא הגיע לעלה וירטואלי או שיש עוד צמתים במחסנית, נכניס את הצומת הנוכחי למחסנית ונזוז לבן השמאלי. כשנגיע לעלה וירטואלי נכניס למערך את הצומת העליון במחסנית (המינימום באותו הרגע) ונזוז לבן הימני של הצומת שהכנסנו למערך.

סיבוכיות המתודה היא מכיוון שנבקר בכל צומת לכל היותר 3 פעמים (בדרך לבן השמאלי, בחזרה מהבן השמאלי אל הימני ובחזרה מהימני להורה), לכן נקבל ביקורים. פעולות המחסנית, כמו גם בדיקות והחזרת שדות נעשות ב- ולכן נקבל סיבוכיות לינארית.

**KeysToArray()**

המתודה מחזירה מערך שלמים שמכיל את מפתחות העץ בסדר עולה.

המתודה משתמשת במתודת העזר nodesToArray() ומקבלת ממנה מערך צמתים מסודר לפי מפתחות בסדר עולה. המתודה עוברת על מערך הצמתים ומכניסה למערך שלמים את מפתחות הצמתים לפי הסדר.

סיבוכיות המתודה היא מכיוון שמתודת העזר פועלת ב- ומעבר על המערך נעשה גם כן ב-. שאר הפעולות נעשות בזמן קבוע.

**infoToArray()**

המתודה מחזירה מערך מחרוזות שמכיל את ערכי העץ מסודרים לפי מפתחות בסדר עולה.

המתודה משתמשת במתודת העזר nodesToArray() ומקבלת ממנה מערך צמתים מסודר לפי מפתחות בסדר עולה. המתודה עוברת על מערך הצמתים ומכניסה למערך מחרוזות את ערכי הצמתים לפי הסדר.

סיבוכיות המתודה היא מכיוון שמתודת העזר פועלת ב- ומעבר על המערך נעשה גם כן ב-. שאר הפעולות נעשות בזמן קבוע.

**Size()**

המתודה מחזירה את ערך השדה size של העץ.

סיבוכיות המתודה היא כיוון שהיא מחזירה ערך של שדה.

**Split(int x)**

* לקרוא את הפוסט בפורום בכותרת " שינוי חתימת המחלקה AVLNode"

**Join(IAVLNode x, AVLtree t)**

* אי אפשר להניח שהעץ t בגובה נמוך מהעץ שעליו מופעלת המתודה (Tree) ?
* לקרוא את השרשור שהדס התחילה בכותרת "JOIN"
* לא ניתן להניח שלפחות אחד מהעצים אינו ריק

**getRoot()**

המתודה מחזירה את ערך השדה root של העץ.

סיבוכיות המתודה היא כיוון שהיא מחזירה ערך של שדה.

**מתודות AVLNode**

* לזכור להשתמש ב-casting אם צריך

**AVLNode(int key, String info)**

המתודה בונה עצם מסוג AVLNode, מאתחלת את המפתח והערך שלו למפתח והערך שהתקבלו כארגומנטים. בנוסף, אם המפתח שהתקבל אינו -1, משנה את ערך ה-isReal לאמת (כיוון שבאופן דיפולטי משתנים בוליאניים מאותחלים לשקר).

סיבוכיות המתודה היא כיוון שהיא מבצעת השמות ובדיקות שנעשות בזמן קבוע.

**getKey()**

המתודה מחזירה את המפתח של הצומת.

סיבוכיות המתודה היא כיוון שהיא מחזירה ערך של שדה.

**getValue()**

המתודה מחזירה את הערך של הצומת.

סיבוכיות המתודה היא כיוון שהיא מחזירה ערך של שדה.

**setLeft(IAVLNode node)**

המתודה מגדירה את שדה הבן השמאלי של הצומת לצומת שהתקבל כארגומנט.

סיבוכיות המתודה היא כיוון שהיא קובעת ערך של שדה.

**getLeft()**

המתודה מחזירה את צומת הבן השמאלי של הצומת.

סיבוכיות המתודה היא כיוון שהיא מחזירה ערך של שדה.

**setRight(IAVLNode node)**

המתודה מגדירה את שדה הבן הימני של הצומת לצומת שהתקבל כארגומנט.

סיבוכיות המתודה היא כיוון שהיא קובעת ערך של שדה.

**getRight()**

המתודה מחזירה את צומת הבן הימני של הצומת.

סיבוכיות המתודה היא כיוון שהיא מחזירה ערך של שדה.

**setParent(IAVLNode node)**

המתודה מגדירה את שדה ההורה של הצומת לצומת שהתקבל כארגומנט.

סיבוכיות המתודה היא כיוון שהיא קובעת ערך של שדה.

**getParent()**

המתודה מחזירה את צומת ההורה של הצומת.

סיבוכיות המתודה היא כיוון שהיא מחזירה ערך של שדה.

**isRealNode()**

המתודה מחזירה אמת אם הצומת אמיתית ושקר אם היא עלה וירטואלי.

סיבוכיות המתודה היא כיוון שהיא מחזירה ערך של שדה.

**setHeight(int height)**

המתודה מגדירה את שדה הגובה של הצומת לערך שהתקבל כארגומנט.

סיבוכיות המתודה היא כיוון שהיא קובעת ערך של שדה.

**getHeight()**

המתודה מחזירה את גובה הצומת.

סיבוכיות המתודה היא כיוון שהיא מחזירה ערך של שדה.

**שאלות נוספות**

* לקרוא את שרשור finger search כיצד למקם איברים חדשים