

מבנה נתונים גנרי - SortedBinarySearchTree

בחלק זה אסור להשתמש ב STL.

1.1. תיאור כללי

בחלק זה של התרגיל נממש מבנה נתונים גנרי ב++C תוך שימוש בתבניות, חריגות ואיטרטורים.

1. כמות האיברים ב-BST אינה חסומה.
2. הוספת איברים מתבצעת ע"י פעולת insert.
3. הסרת איברים מבוצעת ע"י פעולת remove.
4. איברי הרשימה ממוינים בכל רגע נתון – פעולות insert, remove שומרות על הסדר בין איברי הרשימה.
לכל איבר יש לכל היותר 2 ילדים.
הסדר בין האיברים מוגדר ע"י אופרטור > של הטיפוס T.

לתבנית המחלקה BST יהיה פרמטר יחיד: class T. פרמטר זה יהיה טיפוס האלמנטים של BST.

שימו לב: עליכם להניח כמות הנחות מינימלית על הטיפוס T. לא ניתן להניח שקיים בנאי חסר ארגומנטים לT.

2.1 ממשק ה-BTS sorted

1. בנאי חסר פרמטרים ליצירת BinarySearchTree ריק..
2. הורס ל-BST
3. בנאי העתקה.
5. insert – מתודה שמקבלת אלמנט חדש כפרמטר ומכניסה אותו לעץ הבינארי, **אין להוסיף איבר אם האיבר כבר קיים.**
6. remove – מתודה שמקבלת איטרטור (יפורט בהמשך) על המבנה כפרמטר ומסירה את האלמנט אליו הוא מפנה מהעץ.
7. inOrderTraversial – מתודה שמדפיסה את האיברים לפי הסדר מהקטן לגדול.
8. getMax – מתודה שמחזירה את האיבר המקסימלי בעץ.
9. getMin – מתודה שמחזיה את האיבר המינימלי בעץ.

3.1 איטרטור קבוע עבור BinarySearchTree

בנוסף לממשק הקיים, נממש מחלקת איטרטור קבוע עבור הרשימה, אשר תוגדר בתור `BinarySearchTree::const_iterator`.

על האיטרטור לספק את הממשק הבא:

1. `Bnאי העתקה ואופרטור השמה` – מקבלים כפרמטר איטרטור אחר ומבצעים פעולה סטנדרטית.
`אסור לאפשר גישה לבנאי הסטנדרטי של המחקלה.`
2. `הורס לאיטרטור.`
3. `מימוש לאופרטור ++ שמקדם את האיטרטור לאיבר הבא בעץ.`
`אם האיטרטור מצביע לסוף העץ, יש לזרוק חריגה מטיפוס – std::out_of_range.`
4. `operator==` – מימוש לאופרטור `==` שמקבל איטרטור נוסף ובודק האם שניהם שווים.
5. `operator*` - מימוש לאופרטור `*` שיחזיר `reference const` לאיבר אליו האיטרטור מפנה.

