Постановка задачі: Реалізуйте розширюване дерево.

Розв'язок: задача пошуку дерева Splay вирішена за допомогою стандартного пошуку бінарного пошуку дерева. Якщо пошук виконан успішно, то знайдений вузол розвертається і стає новим коренем . Ще один останній доступ, доступ до якого був досягнутий до NULL розвертається і стає новим коренем.

Zig: Вузол є дочірнім по відношенню до кореня (вузол не має бабусі і дідусі). Вузол - або лівий дочірній корінь (ми робимо правильне обертання), або вузол є правим дочірнім елементом його батька (ми робимо ліве обертання).

Вузол має як батька, так і дідуся. Можуть бути такі випадки:

**a) Zig-Zig и Zag-Zag** Node - це залишився дочірній елемент батьківського елемента, а батьківський елемент також є дочірнім елементом батьківського батька (два правильних обертання). OR node - це правильний дочірній елемент батьківського елемента, а батьківський елемент також правий дитина великого батька (два лівих обертання).

**b) Zig-Zag** і **Zag-Zig** Node - це лівий дочірній елемент батька, а батько - правильний дитина великого батька (ліве обертання з наступним правим обертанням). OR node - це правильний дитина його батька і parent - це ліве дочірнє особа великого батька (праве обертання з наступним лівим обертанням).

Важливо відзначити, що операція пошуку або відтворення не тільки повертає шуканий ключ до кореня, а й балансує BST. Наприклад, у наведеному вище випадку висота бінарного дерева пошуку зменшується на 1.

Висновок:

1)Всі операції дерева splay приймають середнє значення O (Log n). Splay дерева можуть бути строго показані для виконання в O (log n) середній час на операцію, по будь-якій послідовності операцій (за умови, що ми починаємо з порожнього дерева)

2) Дерева Splay простіше в порівнянні з AVL і Red-Black Trees, так як в кожному вузлі дерева не потрібне додаткове поле.

3) На відміну від дерева AVL дерево splay може змінюватися навіть за допомогою операцій тільки для читання, таких як пошук.