1.

Čísla ze zadané posloupnosti postupně vkládejte do prázdného binárního vyhledávacího stromu (BVS), který nevyvažujte. Jak bude vypadat takto vytvořený BVS? Poté postupně odstraňte první tři prvky. Jak bude vypadat výsledný BVS?

Posloupnost a) 14 24 5 13 1 3 22 10 19 11
Posloupnost b) 10 16 5 17 4 15 3 1 23 13 2 11
Posloupnost c) 17 4 15 2 5 9 1 12 3 19 16 18

2.

Mějme klíče 1, 2, 3, ..., *n*. Číslo *n* je liché. Nejprve vložíme do BVS všechny sudé klíče v rostoucím pořadí a pak všechny liché klíče také v rostoucím pořadí. Jaká bude hloubka výsledného stromu? Změnil by se nějak tvar stromu, kdybychom lichá čísla vkládali v náhodném pořadí?

3.
Je dán BVS s n uzly. Máme za úkol spočítat hodnotu součtu všech klíčů v tomto stromě. Jaká bude složitost této operace, když to uděláme efektivně?

4.
Uzel binárního binárního vyhledávacího stromu obsahuje tři složky: Klíč a ukazatele na pravého a levého potomka. Navrhněte rekurzívní funkci (vracející `bool`), která porovná, zda má dvojice stromů stejnou strukturu. Dva BVS považujeme za strukturně stejné, pokud se dají nakreslit tak, že po položení na sebe pozorovateli splývají, bez ohledu na to, jaká obsahují data.

5.

Napište rekurzivní verze operací: TreeMinimum, která vrátí referenci na uzel s nejmenší hodnotou v BVS.

6.

Napište funkci, která obrátí pořadí prvků v binárním vyhledávacím stromu. Obrácené pořadí znamená, že po výpisu v pořadí inorder (který neimplementujte!), budou prvky srovnány od největšího k nejmenšímu. Proveďte rekurzivně i nerekurzivně.

7.

Napište funkci, jejímž vstupem bude ukazatel (=reference) na uzel X v BVS a výstupem ukazatel (=reference) na uzel s nejbližší vyšší hodnotou ve stromu.

8.

Navrhněte algoritmus, který spojí dva BVS *A* a *B*. Spojení proběhne tak, že všechny uzly z *B* budou přesunuty do *A*, přičemž se nebudou vytvářet žádné nové uzly ani se nebudou žádné uzly mazat. Přesun proběhne jen manipulací s ukazateli. Předpokládejte, že v každém uzlu v *A* i v *B* je k dispozici ukazatel na rodičovský uzel.