**1.** 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?

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

**4.** 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.

**5.** 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.

**6.** 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, to jest nepoužijí se operace Insert a Delete. Přesun proběhne jen manipulací s ukazateli. Předpokládejte, že v každém uzlu v *A* i v *B* je k dispozici ukazatel na rodičovský uzel.

**7.** Do nejprve prázdného stromu splay tree vkládejte postupně klíče 2, 7, 1, 4, 3, 9, 5, 6. Nakreslete strom po každém vložení.

**8.** Splay tree obsahuje 7 klíčů 1, 2, ..., 7. a je ideálně vyvážený, to jest má hloubku 2. Po vyhledání prvku s klíčem 1 se tento prvek přesune do kořene stromu. Jakou hloubku bude mít výsledný strom?

**9.** Předpokládejme, že perfektně vyvážený splay tree má hloubku h > 0. Po přístupu k prvku s nejmenším klíčem ve stromu se strom změní a vzroste jeho hloubka. Jaká bude nová hloubka stromu? Řešte zvlášťpro sudé a liché h.

Červenočerný strom má řadu vlastností, které je nutno si pamatovat, odvozují se špatně. Citujeme:

*1. Every node is either red or black*

*2. Every leaf (nil) is black*

*3. If a node is red, then both its children are black*

*4. Every simple path from a node to a descendant leaf contains the same number of black nodes*

*5. Root is black*

**10.**  Navrhněte červenočerné obarvení daných stromů tak, aby vznikl korektní červenočerný strom. Prázdné (nil) listy nejsou zobrazeny.



A. B. C.