Domácí úkol 6:

Navrhněte úpravu nějakého vyhledávacího stromu tak aby kromě Find, Insert a Delete (v $\mathcal{O}(\log n)$) uměl i následující operace:

- Median() - vrátí medián všech hodnot v čase $\mathcal{O}(1)$. Medián na lichém počtu prvků je definován jako prostřední prvek, na sudém počtu je to průměr prostředních dvou.
- IntAvg(a,b) - vrátí průměr všech hodnot z intervalu $< a, b >$ v čase $\mathcal{O}(\log n)$.

Hint: Pro dotazy fungující v čase $\mathcal{O}(1)$ je potřeba aktuální hodnotu "nějak" průběžně udržovat. Pro intervalové dotazy v čase $\mathcal{O}(\log n)$ je potřeba udržovat aktuální hodnoty v podstromech podobně jako na cvičení (inspiraci můžete najít v kapitole 4.5. v Průvodci).

Předpokládáme, že strom ukládá celá čísla, řazená dle hodnot (bez klíčů). Chcete-li, můžete si představovat konkrétní typ vyhledávacího stromu (AVL, (a,b), RB, ...). Pro jednoduchost mějme hodnoty pouze v listech. Navíc se nám může hodit trik ze cvičení, kde si listy propojíme do spojového seznamu dle pořadí (a seznam udržujeme při Insert a Delete).

Operace chceme implementovat tak, aby byly přesné. Počítače neumí operovat s desetinnými čísly (můžete otestovat, že téměř vždy $0.8 + 0.2 \neq 1$).

Chceme zachovat složitost základních operací.

Popisovat můžete slovně, není vyžadován pseudokód. V řešení popište změny v reprezentaci stromu, implementaci příslušných operací, a změny v základních operacích (Find,Insert,Delete), které je třeba provést. Nezapomeňte také na úpravy ve vyvažování! Stručně zdůvodněte časovou složitost.

Řešení:

Pozn: Řešení níže je popsáno ve velkém detailu. Některé části jsme prováděli na cvičení nebo jsou k nalezení v průvodci, a technicky tedy stačilo se na ně odvolat, např. udřování spojového seznamu, udržování hodnot podstromů a rozklad na intervalu na podstromy.

Úpravy stromu: Výchozí vyhledávací strom upravíme následovně. Uložené prvky propojíme do spojového seznamu (toto není nutné, pokud zvolíme jinou implementaci mediánu). Stranou si budeme pamatovat ukazatel(e) na prvky pro reprezentaci mediánu, a v každém uzlu si budeme pamatovat počet prvků v podstromě daného uzlu, a jejich součet.

Medián: Aktuální hodnotu mediánu budeme reprezentovat jako dvojici ukazatelů na prvky. Ukazatele interpretujeme jako indikující jediný prostřední prvek pokud počet prvků ve stromě je lichý (ukazatele jsou stejné), nebo indikující mezeru mezi dvěma prostředními prvky (ukazatelé ukazují na dva po sobě jdoucí prvky). Ekvivalentně lze mít jediný ukazatel a bit indikující zda reprezentujeme daný prvek, nebo mezeru za ním. Dotaz na medián vydá průměr hodnot na které ukazatele ukazují, v konstantním čase. Pokud ve stromě nejsou žádné prvky, medián není definovaný. Výsledek je polo-celočíselný, a tedy přesně reprezentovatelný.

Udržování mediánu: (pro jednoduchost předpokládáme binární strom) Insert modifikujeme tak, aby udržoval spojový seznam prvků, t.j. po nalezení místa pro zavěšení nového prvku jako listu jeho otec je buď předchůdce nebo následník. Do spojového seznamu tedy můžeme prvek zapojit v konstantním čase. (Pokud nevkládáme jako list, můžeme najít např. nejlevětší list v pravém podstromě, toto vyhledávání zvládneme v rámci složitosti Insertu).

Po vložení zkontrolujeme, zda vložený prvek je vložen za pozici mediánu. Pokud ano, pozici mediánu posuneme o polovinu (tzn. z jediného prvku na následující mezeru, nebo z mezery na následující prvek). Pokud je nový prvek vložen před ukazatele, posouváme symetricky na druhou stranu. Speciální případy nastanou, když je nový prvek vložen do mezery reprezentující medián nebo do prázdného stromu, potom se stává novým mediánem. Udržování po každém Insertu trvá konstantní čas.

Delete upravíme analogicky, s posuny na opačnou stranu (směrem k odstraněnému prvku). Navíc je třeba ošetřit případy, kdy byl smazán jeden (nebo oba) z prvků reprezentující medián. Případy jsou symetrické k Insertu.

Alternativní implementace: Možnou alternativou je využít uložené počty hodnot v podstromech, a se znalostí celkového počtu prvků po každém Insert a Delete najít prostřední prvek, resp. prvky. Vyhledávání zvládneme v $\mathcal{O}(logn)$, a tedy nedojde ke zhoršení časové složitosti Insert ani Delete. Pokud je počet prvků sudý, můžeme najít oba prostřední prvky, a není poté potřeba spojový seznam prvků. V praxi toto řešení nebude vhodné pokud náš strom ještě neobsahuje počty prvků v podstromech, a způsobí zbytečné přístupy do paměti.

Intervalový průměr:

Pro výpočet průměru určíme počet a součet prvků na zadaném intervalu. K tomu si interval rozložíme na podintervaly reprezentované podstromy struktury.

Pro rozklad stačí nahlédnout, že pokud vyhledáváme a (levý okraj intervalu), všechny prvky uložené v pravých podstromech cesty z kořene do a jsou větší než a. Pravými podstromy myslíme podstromy pod ukazateli vedoucími více doprava než kam vede cesta z kořene do a. Pro ostrost nerovnosti tady využíváme předpoklad neexistence duplicitních hodnot ve stromě. Analogicky levé podstromy cesty do b obsahují pouze prvky menší než b. Najdeme tedy všechny podstromy, které splňují obě podmínky.

Nejprve najdeme nejnižšího společného předchůdce P uzlů s hodnotami a a b, třeba tak, že budeme vyhledávat paralelně a i b a když se rozejdou, vyhledávání se nachází v P a ukončíme ho. Následně pokračujeme ve vyhledávání hodnoty a z P. Všechny pravé podstromy jsou součástí intervalu (splňují obě podmínky). Symetricky provedeme vyhledání b z P. V průběhu obou vyhledávání budeme akumulovat celkový součet a počet prvků v podstromech. Poznamenejme, že také musíme zahrnout některé hodnoty na samotných cestách mezi P a uzly kam doje vyhledávání a a b pokud nepředpokládáme, že hodnoty jsou pouze v listech (u vyhledávání a zahrnujeme každý uzel ze kterého vyhledávání pokračuje doleva, a samotný uzel s a, pokud existuje; pro b symetricky). Také musíme zahrnou hodnotu v P. Celkově zahrneme nejvýše $\mathcal{O}(logn)$ podstromů a $\mathcal{O}(logn)$ jednotlivých uzlů. Vydáme podíl akumulovaného součtu a počtu prvků.

Pro korektnost lze nahlédnout, že každý prvek na intervalu $< a, b >$ musí ležet napravo od cesty z kořene do a nebo přímo na ní (z pozorování výše), pro b symetricky naopak. Navíc všechny takové prvky býtnejvýše v P, protože cesty z kořene do a a b se rozejdou na prvním prvku, který padne do intervalu. Všechny takové prvky jsou zahrnuty. Poznamenejme, že tato pozorování platí i v případě, že hodnota v P je přímo a nebo b.

Výsledný průměr se získá jako jediný podíl celých čísel. V případě potřeby absolutní přesnoti můžeme vydat přímo součet a počet na výstup.

V případě, že interval neobsahuje žádné prvky, průměr je nedefinovaný.

Udržování:

Pro udržování je třeba upravit Insert a Delete, včetně vyvažování, tak, aby udržovany součty a počty prvků v podstromech. Pro Insert stačí po cestě na místo vložení zvyšovat počet prvků ve všech projitých uzlech o 1, a součet o hodnotu vkládaného prvku. U Delete analogicky odčítáme. Musíme ošetřit případy, kdy vkládaný prvek je duplicitní (a tedy možná nebude vložen v závislosti na definici stromu), nebo mazaný prvek ve stromě neexistuje. Je tedy vhodné nejprve prověst Insert/Delete a až poté (pokud operace skutečně proběhne) upravovat hodnoty projitím nahoru ke kořeni (společně s vyvažováím). Pro udržování hodnot během vyvažování si všimneme, že hodnoty v každém uzlu lze v konstantním čase rekonstruovat z hodnot jeho synů. Každá individuální vyvažovací operace upraví pouze konstantně mnoho uzlů, jejich hodnoty tak můžeme ze spoda nahoru rekonstruovat také v konstantním čase. Složitost vyvažování se tak asymptoticky nezvýší.