Příklad první. Připomeňte si definici B-stromů. Kolik je (poměrně) klíčů v každém vrcholu B-stromu?

Příklad druhý. Zkusme B-stromy „zahustit“ – budeme nyní chtít, aby každý* nekořenový vrchol byl zaplněn alespoň na 2/3. Vymyslete, jak to zaručit při vkládání a mazání klíčů. Nedají se klíče u daného vrcholu nějak půjčovat odjinud?

**Poznámka:** Rozhodně nemůžeme chtít při prvním štěpení z 1 vrcholu na 3, abychom všechny tři zaplnili na alespoň 2/3. Toto je jediná situace, kdy to není možné provést.

Příklad třetí. $(a, b)$-strom je proměnlivěji zadaný B-strom: všechny listy jsou opět ve stejné hloubce, každý nekořenový vrchol má alespoň $a - 1$ a nejvýše $b - 1$ položek, kořen má nejvýše $b - 1$ položek, ale minimum nemá. $(a, b)$ tedy určuje počet synů.

A nyní ta opravdu zajímavá věc: $(2, 4)$-stromy jsou strukturálně isomorfní červeno-černým stromům. Můžeme tedy najít zobrazení klíčů RB stromu do klíčů $(2, 4)$-stromu tak, že se axiomy RB stromů přesně přeloží na axiomy $(2, 4)$-stromů, a dokonce i úpravy $(2, 4)$ stromů (které se chovají jako B-stromy) přesně popisují rotace v RB stromech.

Vymyslete přesně, jak se tenhle isomorfismus dá nahlédnout, ukažte, že axiomy sedí, a popište operace RB stromu pomocí toho, co se děje ve $(2, 4)$ stromu. Nemusíte všechny, ale alespoň insert.

A když to dokážete, tak už si RB stromy a jejich operace vždycky snadno odvodíte.

Příklad čtvrtý. Jak se v BVS hledá následník, tj. nejmenší větší prvek? Jak dlouho trvá, když začneme v minimu a budeme strom procházet opakovaným hledáním následníků?

Příklad pátý. Vymyslete, jde-li BVS upravit, aby uměl vyhledat $k$-tý nejmenší prvek.

Příklad šestý. $(2, 3)$-stromy jsou zřejmě neisomorfní RB stromům, protože nejsou isomorfní ani $(2, 4)$-stromům. Zkuste ale přidat do RB stromů nový axiom, který je udělá ekvivalentní $(2, 3)$-stromům.

---

Pokusím se do každého dalšího zadání vložit alespoň jednu úlohu, která používá něco z předchozích cvičení. Na pomoc si často vezmu starší ročníky KSP.

Příklad sedmý – KSP 16-3-1. Newtoon trénuje svoji blechu na bleší turnaj. Ten probíhá na svislé stěně, na které jsou vodorovné plošinky. Cílem blechy je slézt ze startovací polohy co nejdříve na podlahu. Pohyb blechy závisí na tom, zda je blecha na nějaké plošince, či padá. Pokud padá, klesne za jednu blechovteřinu o jeden blechometr. Pokud je na plošince, posune se za jednu blechovteřinu o jeden blechometr vlevo či vpravo.

Závod tedy probíhá tak, že blecha padá, padá, až dopadne na plošinku. Pak se rozhodne (nebo jí její majitel přikáže), zda půjde doleva nebo doprava, a jde, dokud nedojde na konec plošinky. Z ní pak seskočí a zase padá, dokud se nedostane na podlahu. Vítězí blecha, která přistane na podlaze jako první. Ovšem je nutné, aby žádná blecha nespadla z větší výšky než v blechometrů, jinak se totiž po dopadu urazí a odmítne pokračovat v závodě.

Na vstupu dostanete jednak počáteční souřadnice blechy (v celých blechometrech), v, což je největší výška, ze které může blecha spadnout, aby se neurazila, a N, což je počet plošinek na stěně. Dále dostanete popis N plošinek, u každé plošinky souřadnice jejího horního dolního rohu a její šířku (vše opět v celých blechometrech). Všechny plošinky jsou vysoké jeden blechometr a žádné dvě se nedotýkají. Blecha je na podlaze, pokud se nachází na souřadnicích [x;0], kde x je libovolné celé číslo.

Výstupem Vašeho programu je nejmenší počet blechovteřin, které bude blecha potřebovat, aby se dostala na podlahu. Kromě tohoto počtu vypište i počet plošinek, na které blecha dopadne, a u každé plošinky (v pořadí, jak na ně blecha dopadá) rozhodněte, zda má blecha jít vlevo či vpravo. Pokud úloha nemá řešení (moc malé v), vypište odpovídající zprávu.