1.
RL rotace v uzlu *u* lze rozložit na

- o levou rotaci v pravém synovi uzlu *u* následovanou pravou rotací v uzlu *u*
- ✓ pravou rotaci v pravém synovi uzlu *u* následovanou levou rotací v uzlu *u*
- o levou rotaci v levém synovi uzlu *u* následovanou pravou rotací v uzlu *u*
- o pravou rotaci v levém synovi uzlu *u* následovanou levou rotací v uzlu *u*

Namalujte si tři obrázky. Nejprve (dostatečně velký) binarní strom a pojmenujte(!) v něm uzly. Na druhém obrázku bude tento strom po RL rotaci v kořeni. Na třetím obrázku nejprve původní strom překreslete po R rotaci v pravém následníku kořene a tento naposled jmenovaný strom pak ještě po L rotaci v kořeni. Výsledek třetího obrázku je shodný se druhým obrázkem, takže druhá možnost je správně.
Poznámka: U zkoušky se několik respondentů domáhalo vysvětlení, proč je jejich odpověď chybná, nakreslit si však dotyčné stromy zarputile odmítali (-- ??).

2.
LR rotace v uzlu *u* lze rozložit na

- o levou rotaci v pravém synovi uzlu *u* následovanou pravou rotací v uzlu *u*
- o pravou rotaci v pravém synovi uzlu *u* následovanou levou rotací v uzlu *u*
- ✓ levou rotaci v levém synovi uzlu *u* následovanou pravou rotací v uzlu *u*
- o pravou rotaci v levém synovi uzlu *u* následovanou levou rotací v uzlu *u*

Namalujte si tři obrázky. Nejprve (dostatečně velký) binarní strom a pojmenujte(!) v něm uzly. Na druhém obrázku bude tento strom po LR rotaci v kořeni. Na třetím obrázku nejprve původní strom překreslete po L rotaci v levém následníku kořene a tento naposled jmenovaný strom pak ještě po R rotaci v kořeni. Výsledek třetího obrázku je shodný se druhým obrázkem, takže třetí možnost je správně.

3.
Levá rotace v uzlu *u*

a) v podstromu s kořenem *u* přemístí pravého syna u_p uzlu *u* do kořene. Přitom se uzel *u* stane levým synem uzlu u_p a levý podstrom uzlu u_p se stane pravým podstromem uzlu *u*
b) v podstromu s kořenem *u* přemístí levého syna u_l uzlu *u* do kořene. Přitom se uzel *u* stane pravým synem uzlu u_l a levý podstrom uzlu u_l se stane pravým podstromem uzlu *u*
c) v podstromu s kořenem *u* přemístí pravého syna u_p uzlu *u* do kořene. Přitom se uzel *u* stane levým synem uzlu u_p a pravý podstrom uzlu u_p se stane levým podstromem uzlu *u*
d) v podstromu s kořenem *u* přemístí levého syna u_l uzlu *u* do kořene. Přitom se uzel *u* stane pravým synem uzlu u_l a pravý podstrom uzlu u_l se stane levým podstromem uzlu *u*

Zde můžeme jen doporučit nakreslení obrázku, případně nahlédnutí do vhodné publikace, možnost a) se ukáže být jedinou správnou.

4.
Pravá rotace v uzlu *u*

a) v podstromu s kořenem *u* přemístí pravého syna u_p uzlu *u* do kořene. Přitom se uzel *u* stane levým synem uzlu u_p a levý podstrom uzlu u_p se stane pravým podstromem uzlu *u*
b) v podstromu s kořenem *u* přemístí levého syna u_l uzlu *u* do kořene. Přitom se uzel *u* stane pravým synem uzlu u_l a levý podstrom uzlu u_l se stane pravým podstromem uzlu *u*

c) v podstromu s kořenem u přemístí pravého syna u_p uzlu u do kořene. Přitom se uzel u stane levým synem uzlu u_p a pravý podstrom uzlu u_p se stane levým podstromem uzlu u
d) v podstromu s kořenem u přemístí levého syna u_l uzlu u do kořene. Přitom se uzel u stane pravým synem uzlu u_l a pravý podstrom uzlu u_l se stane levým podstromem uzlu u

Zde můžeme – stejně jako v předchozí úloze – jen doporučit nakreslení obrázku, případně nahlédnutí do vhodné publikace, možnost d) se ukáže být jedinou správnou.

5.
Vyvážení BVS některou z operací rotace

a) je nutno provést po každé operaci Insert
b) je nutno provést po každé operaci Delete
c) je nutno provést po každé operaci Insert i Delete
d) snižuje hloubku stromu
e) není pro udržování BVS nezbytné

Binární vyhledávací stromy se vyvažovat mohou a také nemusí. Možnosti a) b) c) tedy opadají a zřejmě platí možnost e). Možnost d) neplatí, protože rotace nemusí snížit hloubku stromu, je-li provedena dostatečně vysoko ve stromu v některé z kratších větví (jednoduché cvičení – najděte na to příklad, budete potřebovat pět pater stromu).

6.
Binární vyhledávací strom:

a) musí splňovat podmínku haldy
b) udržuje v každém uzlu referenci na uzel s nejbližším větším klíčem
c) udržuje v každém uzlu referenci na uzel s nejbližším větším i s nejbližším menším klíčem
d) po každém vložení prvku do BVS musí proběhnout vyvážení stromu
e) po průchodu v pořadí inorder vydá seřazenou posloupnost klíčů

Podmínku haldy musí splňovat pouze halda, která ani není BVS. Reference na jiné uzly než jen bezprostřední potomky nejsou v BVS povinností a vyvažování BVS také není povinné. Zbývá tak jen možnost e), o níž se hovoří i při výkladu pořadí inorder jako takového.

7a.
Jednoduchá levá (nebo pravá) rotace v uzlu u má operační složitost

a) závislou na výšce levého podstromu uzlu u
b) mezi $O(1)$ a $\Omega(n)$
c) závislou na hloubce uzlu u
d) konstantní

Provedení kterékoli rotace nezávisí na poloze uzlu ve stromu a vyžaduje jen změnu ukazatelů na předem známý počet uzlů – nejvýše 3 v jednoduché rotaci (malujte si obrázek) případně o jeden více uvažujeme-li i ukazatele na rodiče uzlu, v němž rotace probíhá. To ovšem znamená, že se jedná o konstantní složitost – varianta d).

7b.
Jednoduchá pravá rotace v uzlu *u* má operační složitost

a) $\Theta(1)$
b) závislou na hloubce uzlu *u*
c) mezi $O(1)$ a $\Omega(\log n)$
d) $\Omega(n)$

Zde je nutno vědět, jak přibližně rotace vypadá. Mění jen několik ukazatelů na uzly stromu. Intuitivně by mělo pak být jasné, že má složitost konstantní. Varianty b) a d) v takovém případě neplatí. Varianta c) tvrdí, že dotyčná složitost je mezi $O(1)$ a $\Omega(\log n)$, což znamená, že sice roste pomaleji než $\log(n)$, ale zároveň rychleji než konstanta (která „neroste vůbec“). Platí ovšem a).

8.
LR (nebo RL) rotace v uzlu *u* má operační složitost

a) závislou na hloubce uzlu *u*
b) $\Theta(\log n)$
c) konstantní
d) lineární

Provedení kterékoli rotace nezávisí na poloze uzlu ve stromu a vyžaduje jen změnu ukazatelů na předem známý počet uzlů – nejvýše 6 ve dvojité rotaci (malujte si obrázek) případně o jeden více uvažujeme-li i ukazatele na rodiče uzlu, v němž rotace probíhá. To ovšem znamená, že se jedná o konstantní složitost – varianta c).

9.
Kolik jednoduchých rotací se maximálně provede při jedné operaci Insert v AVL stromu s *n* uzly?

a) jedna
b) dvě
c) $\log(n)$
d) n

Provádějí-li se rotace vůbec, pak nanejvýš jednoduché (L, R) nebo dvojité (LR, RL). Každá dvojitá rotace je však složením dvou jednoduchých rotací. Pokud došlo k rozvážení v uzlu X, platí toto: Měl-li podstrom s kořenem v X před operací insert hloubku h, pak po operaci insert stoupla hloubka na h+1 (protože X je najednou rozvážený), ale rotace opět tuto hloubku sníží na původní h (malujte si obrázky1). Protože tedy operace insert a následná rotace v X nezměnila hloubku podstromu s kořenem v X, nemohlo ani dojít k rozvážení uzlů kteří leží „nad“ X, tj. na cestě z X ke kořeni celého stromu, tudíž se žádne další rotace po operaci Insert neprovedou a platí možnost b).

10.
Váha $v(u)$ uzlu *u* ve stromě s ohraničeným vyvážením α je definována takto:
$v(u) = 1/2$, když je *u* listem, a

a) $v(u)$ = ((počet uzlů v levém podstromu uzlu u) + 1) / (|U| + 1), kde |U| je celkový počet uzlů v podstromu s kořenem v uzlu u
b) $v(u)$ = ((počet uzlů podstromu s kořenem v uzlu u) + 1) / 2
c) $v(u)$ = ((počet uzlů v levém podstromu uzlu u) + 1) / (|U| + 1), kde |U| je počet uzlů v pravém podstromu uzlu u
d) $v(u)$ = ((počet uzlů v levém podstromu uzlu u) + 1) / 2*|U| + 1), kde |U| je počet uzlů v podstromu s kořenem v uzlu u

Protože strom vyvyžujeme, musíme znát poměr „velikosti" levého a pravého podstromu každého uzlu (ať už je „velikost" definována jakkoli), tudíž možnost b) odpadá, levý a pravý podstrom nijak celý výraz neovlivňují.
I když o vyvažování nic nevíme, zadání napovídá, že vyvážený uzel (list) má váhu ½. Zkusíme tedy zjistit váhu kořene např. ve stromu se 3 uzly a 2 listy. Podle c) vychází váha kořene (1+1)/(1+1) = 1. Podle d) vychází váha kořene (1+1)/(6+1) = 2/7. Podle a) vychází váha kořene (1+1)/(3+1) = ½. Korektní varianta odpovědi je a).

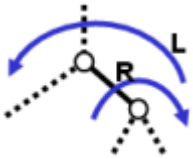
11.
Kolik jednoduchých rotací se maximálně provede při jedné operaci Insert ve vyváženém binárním vyhledávacím stromu s n uzly a s ohraničeným vyvážením?

a) jedna
b) log(n)
c) dvě
d) n

Pro ohraničené vyvážení je podstatný poměr velikostí levého a pravého podstromu každého uzlu. Jakmile provedeme vyvážení (=rotaci) v nějakém uzlu X, zmíněný poměr pro každého předka X se nemůže změnit, neboť rotace nemění počet uzlů v podstromu s kořenem X.
Po vložení prvku operací Insert postupujeme ke kořeni a hledáme rozvážený uzel.
Z řečeného plyne, že takový uzel najdeme nejvýše jednou. Tam však provedeme nanejvýš dvojitou rotaci skládající se ze dvou jednoduchých. Správná varianta odpovědi je c).

12a.
RL rotace v uzlu u lze rozložit na

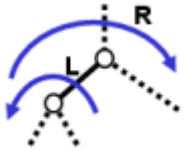
a) levou rotaci v pravém synovi uzlu u následovanou pravou rotací v uzlu u
b) pravou rotaci v pravém synovi uzlu u následovanou levou rotací v uzlu u
c) levou rotaci v levém synovi uzlu u následovanou pravou rotací v uzlu u
d) pravou rotaci v levém synovi uzlu u následovanou levou rotací v uzlu u



Tato otázka se objevila již vloni.
V ní nejspíše nepomůže nic jiného, než vědět zpaměti, jak RL rotace vypadá. LR rotaci si pak není nutno pamatovat, neboť je s ní zrcadlově symetrická. Je tedy zapotřebí udržovat v mysli obrázek podobný uvedenému. Samozřejmě, nemálo přispěje vědomí, že dvojitá! (RL, resp. LR) rotace v uzlu u se aplikuje právě tehdy, když hodnoty klíčů „rozvažujícího, přetíženého" podstromu jsou ve smyslu uspořádání *mezi* hodnotami klíče uzlu u a jeho potomka (pravého, resp. levého). (Na našem obrázku pod druhou tečkovanou čárou zprava).

12b.
LR rotace v uzlu *u* lze rozložit na

a) levou rotaci v pravém synovi uzlu *u* následovanou pravou rotací v uzlu *u*
b) pravou rotaci v pravém synovi uzlu *u* následovanou levou rotací v uzlu *u*
c) levou rotaci v levém synovi uzlu *u* následovanou pravou rotací v uzlu *u*
d) pravou rotaci v levém synovi uzlu *u* následovanou levou rotací v uzlu *u*



Tato otázka se objevila již vloni.
V ní nejspíše nepomůže nic jiného, než vědět zpaměti, jak LR rotace vypadá. RL rotaci si pak není nutno pamatovat, neboť je s ní zrcadlově symetrická. Je tedy zapotřebí udržovat v mysli obrázek podobný uvedenému. Samozřejmě, nemálo přispěje vědomí, že dvojitá! (LR, resp. RL) rotace v uzlu *u* se aplikuje právě tehdy, když hodnoty klíčů „rozvažujícího, přetíženého“ podstromu jsou ve smyslu uspořádání *mezi* hodnotami klíče uzlu u a jeho potomka (levého, resp. pravého). (Na našem obrázku pod druhou tečkovanou čárou zleva).

13a.
Červenočerný strom

a) má maximální výšku rovnou dvojnásobku své červenočerné výšky
b) má ve všech větvích stejný počet uzlů
c) má tři typy uzlů: černé, červené a bílé
d) následníci černého uzlu jsou vždy červení

Červenočerný strom má řadu vlastností, které je nutno si pamatovat, odvozují se špatně.
Citujme z přednášky:
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)

Variantu d) vyvrací strom se dvěma uzly. Jeden je kořenem, druhý je listem. Podle podmínek 2. a 5. Jsou oba černí. Varianta c) je zřejmý nesmysl, variantu b) vyvrací předchozí příklad se dvěma uzly. Zbývá jen správná varianta a). V ní je použit termín červenočerná výška místo termínu černá výška, což je bohužel hloupý překlep, naštěstí jej většina respondentů zaznamenala a nikomu neovlivnil celkový výsledek .

13b.
Červenočerný strom

a) má maximální výšku rovnou 2/3 své červenočerné výšky
b) má červené listy
c) následníci červeného uzlu jsou vždy černí a jsou tři
d) udržuje ve všech větvích stejnou černou výšku

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

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)

Podmínka 3. říká, že v žádné cestě z kořene do listu nemůže být více červených uzlů než černých, výška stromu tedy nepřekročí dvojnásobek jeho černé výšky. Prostřídáme-li ve vyváženém stromu červené a černé vrstvy, vidíme, že lze tohoto maxima téměř dosáhnout. I kdyby ve variantě a) byla napsána černá výška (červenočerná je nesmysl), varianta a) by neplatila. Varianta b) je vyloučena podmínkou 2., varianta nepravdivě c) sugeruje, že strom je ternární (Následník ve smyslu uspořádání zleva doprava může být jen jeden). Platí varianta d) totožná s podmínkou 4.

13c.
Červenočerný strom

a) má černé listy
b) udržuje ve všech větvích stejnou červenou výšku
c) následníci černého uzlu jsou vždy červení
d) má maximální výšku rovnou 2/3 své červené výšky

Červenočerný strom má řadu vlastností, které je nutno si pamatovat, odvozují se špatně. Citujme z přednášky:
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)

Podmínka 3. říká, že v žádné cestě z kořene do listu nemůže být více červených uzlů než černých, výška stromu tedy nepřekročí dvojnásobek jeho černé výšky. Prostřídáme-li ve vyváženém stromu červené a černé vrstvy, vidíme, že lze tohoto maxima téměř dosáhnout. Varianta d) je v rozporu s tímto zjištěním. Varianta c) je také v rozporu s podmínkou 3, podmínka 4 popírá variantu b). Platí a).

13d.
Červenočerný strom

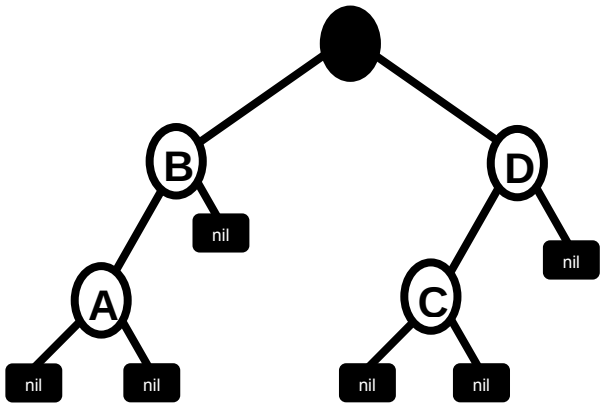
a) má maximální výšku rovnou dvojnásobku své černé výšky
b) má tři typy uzlů: vnitřní černé, vnitřní červené a listy červené
c) následníci červeného uzlu jsou vždy černí a jsou tři
d) má červené listy

Červenočerný strom má řadu vlastností, které je nutno si pamatovat, odvozují se špatně. Citujme z přednášky:
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)

Podmínka 3. říká, že v žádné cestě z kořene do listu nemůže být více červených uzlů než černých, výška stromu tedy nepřekročí dvojnásobek jeho černé výšky. Prostřídáme-li ve vyváženém stromu červené a černé vrstvy, vidíme, že lze tohoto maxima téměř dosáhnout. Varianta b) i d) je vyloučena podmínkou 2. Varianta c) nepravdivě sugeruje, že strom je ternární (Naopak následník ve smyslu uspořádání zleva doprava může být jen jeden).

14.
Červenočerný strom je často používanou strukturou. Jaké barvy mají mít označené uzly, aby strom na obrázku opravdu byl červenočerným stromem?

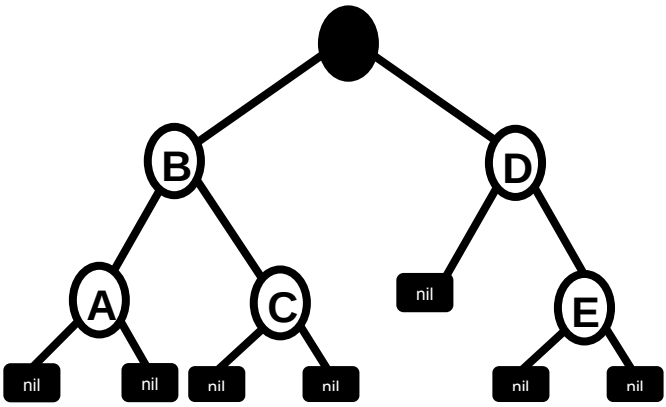
a) A červený, B červený, C černý, D černý
b) A červený, B černý, C červený, D černý
c) A černý, B červený, C černý, D černý
d) A černý, B červený, C černý, D červený



Jedna z vlastností červenočerného stromu je ta, že cesta z kořene do libovolného listu obsahuje stejný počet černých uzlů. Ve větvi DC nemohou být oba uzly červené, další vlastností červenočerného stromu je totiž to, že červený uzel není nikdy bezprostředním potomkem červeného uzlu. Na cestě z kořene do pravého potomka uzlu D však musí být stejný počet černých uzlů jako na cestě z kořene do potomka uzlu C. To lze zajistit pouze tak, že uzel D bude černý a uzel C bude červený. Když analogickou úvahu aplikujeme na větev BA, zjistíme, že uzel B musí být černý a uzel A červený. Celkem tak vychází právě odpověď ve variantě b).

15.
Červenočerný strom je často používanou strukturou. Jaké barvy mají označené uzly, aby strom na obrázku opravdu byl červenočerným stromem?

a) A červený, B červený, C černý, D černý, E černý
b) A černý, B červený, C černý, D černý, E černý
c) A červený, B černý, C červený, D černý, E červený
d) A černý, B červený, C černý, D červený, E černý



Jedna z vlastností červenočerného stromu je ta, že cesta z kořene do libovolného listu obsahuje stejný počet černých uzlů. Ve větvi DC nemohou být oba uzly červené, další vlastností červenočerného stromu je totiž to, že červený uzel není nikdy bezprostředním potomkem červeného uzlu. Na cestě z kořene do levého potomka uzlu D však musí být stejný počet černých uzlů jako na cestě z kořene do potomka uzlu E. To lze zajistit pouze tak, že uzel D bude černý a uzel E bude červený. Když si nyní prohlédneme nabízené varianty, vidíme, že tuto kombinaci obsahuje pouze varianta c). Ještě ji ověřme. Na cestě

z kořene do libovolného listu tedy máme mít (podle rozboru větve DE) vždy jen jediný černý uzel (nepočítaje kořen a list). To lze v levém podstromu kořene zajistit buď tak, že uzly A a C budou černé a uzel B červený, nebo naopak tak, že B bude černý a A a C červené. Tato poslední možnost je vskutku obsažena i ve variantě c).

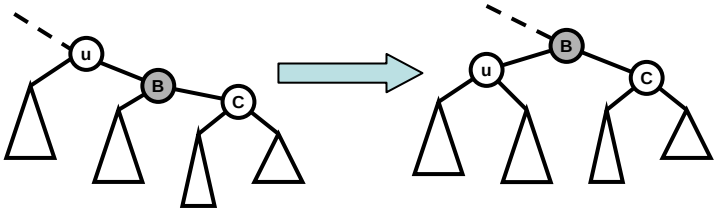
16.
Stromy nad danou množinou n klíčů:

a) AVL strom je vždy ideálně vyvážený
b) váhově vyvážený strom má stejnou hloubku všech listů
c) AVL strom n uzly se vyvažuje v nejhorším případě za použití $\log_2(n)$ rotací
d) RB strom s pouze černými uzly je pravidelný

AVL strom se běžně vyvažuje rotacemi, takže zřejmě není pokaždé ideálně vyvážený, což znamená, že varianta a) neplatí. Obrázek 92 ze 7. přednášky ukazuje váhově vyvážený strom, v němž listy nemají stejnou hloubku. Varianta b) tedy také neplatí. Vyvažování stromu se děje v čase úměrném konstantě, tudíž i varianta c) odpadá. Zbývá tak jen poslední varianta d). RB strom je vždy pravidelný, nezávisle na barvě uzlů.

17a.
Jednoduchá levá rotace v uzlu *u*

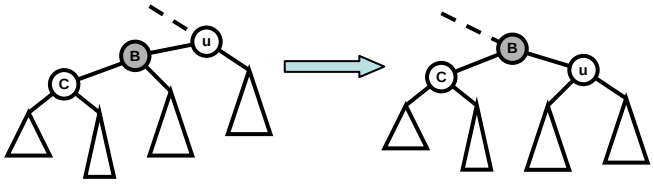
a) sníží hloubku levého syna uzlu *u*
b) vzájemně zamění obsah uzlu *u* s obsahem jeho levého syna
c) přemístí vnitřní uzel nalevo od pravého syna uzlu *u* do kořene
d) sníží hloubku pravého syna uzlu *u*



Pravým synem uzlu *u* na obrázku je uzel B (zvýrazněný) . Tento se po levé rotaci octne na místě sveho rodiče ve stromu, tedy ve větší výšce, tedy v menší hloubce, než byl předtím. Jeho hloubka se snížila. To je obsahem varianty d)
Vidíme také, že hloubka levého syna uzlu u se zvýšila (a), levý syn uzlu B se v kořeni neoctl (c). Varianta b) odpadá sama, rotace nikdy nemanipulují s obsahem uzlů.

17b.
Jednoduchá pravá rotace v uzlu *u*

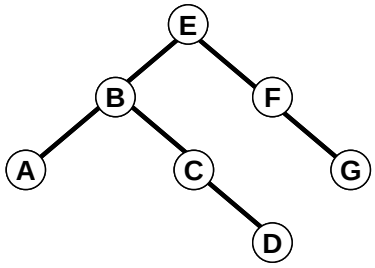
a) sníží hloubku pravého syna uzlu *u*
b) vzájemně zamění obsah uzlu *u* s obsahem jeho levého syna
c) sníží hloubku levého syna uzlu *u*
d) přemístí vnitřní uzel nalevo od pravého syna uzlu *u* do kořene



Úloha je analogická předchozí úloze.
Levým synem uzlu *u* na obrázku je uzel

B (zvýrazněný) . Tento se po pravé rotaci octne na místě sveho rodiče ve stromu, tedy ve větší výšce, tedy v menší hloubce, než byl předtím. Jeho hloubka se snížila. To je obsahem varianty c) Vidíme také, že hloubka pravého syna uzlu u se zvýšila (a), levý syn pravého syna uzlu *u* se v kořeni neoctl (d). Varianta b) odpadá sama, rotace nikdy nemanipulují s obsahem uzlů.

18a.

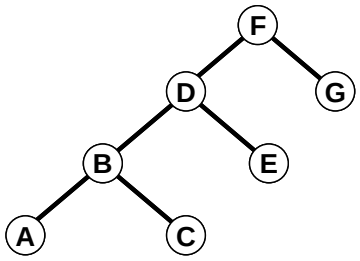


Aby byl strom na obrázku červenočerným stromem červenočerné výšky bh=2, musíme obarvit uzly

a) A,C,D,G na červeno a ostatní na černo
b) B,D,G na červeno a ostatní na černo
c) B,F,D na červeno a ostatní na černo
d) A,C,G na červeno a ostatní na černo

Nezapomínejme, že v červenočerném stromu nemusíme malovat listy, které jsou vždy černé a které neobsahují data. Navíc černá hloubka každého z nich musí být stejná. Kdyby byly uzly A – G všechny černé, měly by právě zmíněné listy různou hloubku. Některé uzly tedy musí být červené. Kořen to být nesmí (definice). Kdyby byl červený uzel F, nastala by potíž: Černá hloubka levého syna F (který je listem a není na obrázku) by byla 2. To by ale znamenalo, že černá hloubka synů D by také byla 2, ale to by dále znamenalo, že žádný další uzel na cestě E B C D nesmí být černý, tudíž by jistě byly B a C červené, ale to by odporovalo pravidlu, že syn červeného uzlu nesmí být červený. Uzel F tedy je černý a černá hloubka všech listů je 3. Uvážíme–li nyní opět syny uzlu D a cestu E B C D, musí být mezi uzly B C D již jen jeden černý (E již černý je), aby měli synové D černou hloubku rovnou 3. To lze splnit jen tak, že černý bude C (jinak bychom měli nedovoleného červeného syna červeného uzlu) a červené B a D. V tomto okamžiku se již rozhodujeme jen mezi variantou b) a c). Řekli jsme ale dříve, že F musí být černý, takže varianta c) odpadá a zbývá jen b).

18b.



Aby byl strom na obrázku červenočerným stromem červenočerné výšky bh=2, musíme obarvit uzly

a) A,C,G na červeno a ostatní na černo
b) B,D,G na červeno a ostatní na černo
c) A,C,D na červeno a ostatní na černo
d) B,F,D na červeno a ostatní na černo

Úloha je předchozí úloze A. Nezapomínejme, že v červenočerném stromu nemusíme malovat listy, které jsou vždy černé a které neobsahují data. Navíc černá hloubka každého z nich musí být stejná. Kdyby byly uzly A – G všechny černé, měly by právě zmíněné listy různou hloubku. Některé uzly tedy musí být červené. Kořen to být nesmí (definice). Kdyby byl červený uzel G, nastala by potíž: Černá hloubka libovolného syna G (který je listem a není na obrázku) by byla 2. To by ale znamenalo, že černá hloubka synů A by také byla 2, ale to by dále znamenalo, že žádný další uzel na cestě F D B A nesmí být černý, tudíž by jistě byly D a B červené, ale to by odporovalo pravidlu, že syn červeného uzlu nesmí být červený. Uzel G tedy je černý a černá hloubka všech listů je 3. Uvážíme–li nyní opět syny uzlu A a cestu F D B A, musí být mezi uzly D B A již jen jeden černý (F již černý je), aby měli

synové A černou hloubku rovnou 3. To lze splnit jen tak, že černý bude B (jinak bychom měli nedovoleného červeného syna červeného uzlu) a červené D a A. V tomto okamžiku našemu rozboru odpovídá již jen varianta c).

19.

Napište proceduru popisující XX rotaci binárního stromu. Nemusíte testovat nenulovost ukazatelů.

Při psaní této procedury je důležité si uvědomit, v jakém pořadí se mají jednotlivá přiřazení provádět. Špatné pořadí může způsobit rozpojení struktury BVS.

Procedury používají následující datovou strukturu pro uzel BVS:

```
typedef struct _uzel {
      int klic; //hodnota klice, ktera bude ulozena v uzlu;
      _uzel *l; //ukazatel na levy podstrom
      _uzel *p; //ukazatel na pravy podstrom
      _uzel *r; //ukazatel na rodice
} uzel;
```

XX=Levá rotace

```
uzel* Lrotace(uzel *k) //vstupem je koren podstromu, ve kterem se rotuje. Vystupem
je novy koren

{
 uzel *p;

 p = k->p;          //do p se ulozi koren praveho podstromu
 k->p = p->l;       //p se stane novym korenem, takze je potreba jeho levy
                    // podstrom presunout. Stane se pravym podstromem puvodniho //
                    korene.
 p->l = k;          //puvodni koren se posune do leveho podstromu
                    //nyni je potreba spravne nastavit ukazatele na rodice
 p->r = k->r;

//potrebujeme zjistit, jestli byl puvodni koren v levem, nebo pravem podstromu
//rodice a podle toho upravit odkaz na novy koren
 if( p->r->l->klic == k->klic )
         p->r->l = p;
      else
         p->r->p = p;
 k->r = p; //puvodni koren je pod p, takze se p stava jeho rodicem
 return p;
}
```

XX=LR rotace (dvojitá)

```
uzel* LRrotace(uzel* k)
{
 uzel *p1,*p2;

 p1 = k->l;
 p2 = p1->p;  //p2 bude novym korenem
 p1->p = p2->l;
 p2->l = p1;
 k->l = p2->p;
 p2->p = k;

 //nyni je potreba spravne nastavit reference na rodice
 p1->r = p2;
 p2->r = k->r;
 k->r = p2;
```

```
//potrebujeme zjistit, jestli byl puvodni koren v levem, nebo pravem podstromu
rodice a podle toho upravit odkaz na novy koren
 if( p2->r->l->klic == k->klic )
       p2->r->l = p2;
      else
       p2->r->p = p2;

 return p2;
}
```

Všechny přesuny během rotací musí zachovat relativní pozici všech uzlů. Pokud byl uzel A vlevo od uzlu B (třeba i nepřímo tak, že byl součástí levého podstromu B), musí zůstat vlevo od B i po rotaci. Pokud by se A dostal v BVS po provedení rotace nad B, musí být B v pravém podstromu A.