Vyvažování a rotace v BVS, všude se předpokládá AVL strom

1.
Jednoduchá levá rotace v uzlu *u* má operační složitost
a) závislou na výšce levého podstromu uzlu *u*
b) mezi O(1) a $\Theta(n)$
c) závislou na hloubce uzlu *u*
d) konstantní

Řešení 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).

2a.
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

Řešení 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.

2b.
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

Řešení 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.

3.
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*

Řešení Provádějí-li se rotace vůbec, pak leda 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 obrázky!). 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 žádné další rotace po operaci Insert neprovedou a platí možnost b).

4.
LR rotace v uzlu *u* má operační složitost
a) závislou na hloubce uzlu *u*
b) Θ (log *n*)
c) konstantní
d) lineární

Řešení 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).

5a.
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

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

5b.
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

Řešení Zde můžeme – stejně jako ve vedlejším oddělení – jen doporučit nakreslení obrázku, případně nahlédnutí do vhodné publikace, možnost d) se ukáže být jedinou správnou.

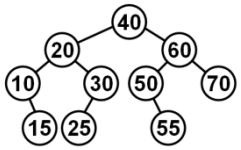
6.
Vyvážení BVS některou z operací rotace

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

Řešení 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).

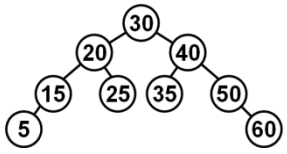
7a.
Na obrázku je uveden BVS, který v průběhu práce vyvažujeme (AVL strom). Ten nyní upravíme tak, že z něj odstraníme operací Delete uzly s klíči 50, 30, 25 v tomto pořadí. Rozhodněte, zda a jaká rotace bude během této úpravy použita:

a) L rotace
b) R rotace
c) LR rotace
d) RL rotace
e) žádná rotace



7b.
Na obrázku je uveden BVS, který v průběhu práce vyvažujeme (AVL strom). Ten nyní upravíme tak, že z něj odstraníme operací Delete uzly s klíči 5, 25, 35 v tomto pořadí. Rozhodněte, zda a jaká rotace bude během této úpravy použita:

a) L rotace
b) R rotace
c) LR rotace
d) RL rotace
e) žádná rotace



8a.
Do prázdného BVS, který v průběhu práce vyvažujeme (AVL strom), vložíme klíče 23, 11, 24, 31, 17, 20, 30, 25. Po prvním rozvážení stromu je třeba provést rotaci, která strom opět vyváží. Tato rotace bude:

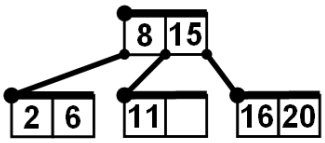
a) L
b) R
c) LR
d) RL
e) žádná, strom se v průběhu vkládání daných klíčů nerozváží

8b.
Do prázdného BVS, který v průběhu práce vyvažujeme (AVL strom), vložíme klíče 25, 30, 20, 17, 31, 24, 11, 12. Po prvním rozvážení stromu je třeba provést rotaci, která strom opět vyváží. Tato rotace bude:

a) L
b) R
c) LR
d) RL
e) žádná, strom se v průběhu vkládání daných klíčů nerozváží

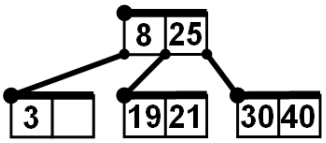
B-strom

9a.
Do B-stromu znázorněného na obrázku vložíme postupně klíče 14, 10. Pak bude kořen obsahovat klíč/klíče



a) 8, 10
b) 8, 11
c) 10
d) 11
e) 11, 14

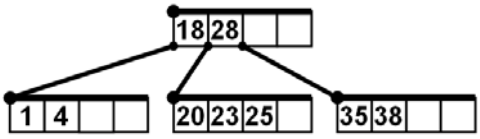
9b.
Do B-stromu znázorněného na obrázku vložíme postupně klíče 7, 5. Pak bude kořen obsahovat klíč/klíče



a) 5
b) 5, 7
c) 7
d) 7, 8
e) 8

10.
Klíče 1, 2, 3, 4, …, 10, 11, 12 v tomto pořadí vložte do prázdného B-stromu řádu 2. Nakreslete tento strom po vložení prvních čtyř, prvních osmi a nakonec po vložení všech daných klíčů.

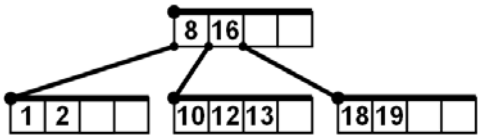
11a.
Z B-stromu znázorněného na obrázku odebereme postupně klíče 4, 35. Pak bude kořen obsahovat klíč/klíče



a) 18
b) 18, 28
c) 20
d) 20, 28
e) 28

11b.
Z B-stromu znázorněného na obrázku odebereme postupně klíče 18, 2. Pak bude kořen obsahovat klíč/klíče

a) 8
b) 8, 16
c) 10
d) 10, 12
e) 13



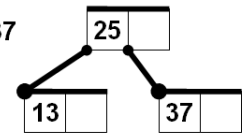
12.
B-strom je řádu k, pokud každý jeho uzel, kromě kořene, musí obsahovat alespoň k klíčů a zároveň může obsahovat nejvýše 2k klíčů. Vybudujte B-strom řádu 1 tak, že do prázdného stromu vložíte v uvedeném pořadí klíče 25, 13, 37, 32, 40, 20, 22.
Dále tento strom zrušte, a to tak, že jednotlivé klíče klíče odstraníte v pořadí
13, 25, 40, 22, 20, 37, 32. Nakreslete strom po každé operaci Insert a Delete.

Řešení

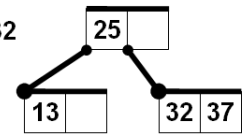
Insert 25

Insert 13

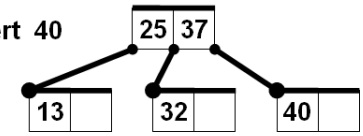
Insert 37



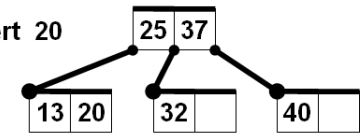
Insert 32



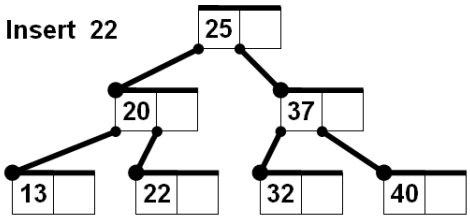
Insert 40



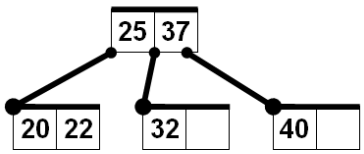
Insert 20



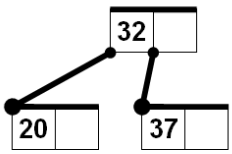
Insert 22



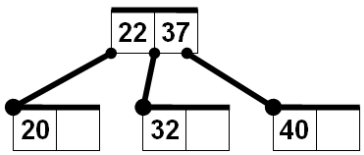
Delete 13



Delete 22



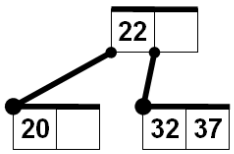
Delete 25



Delete 20

Delete 37

Delete 40



Delete 32 **empty tree**

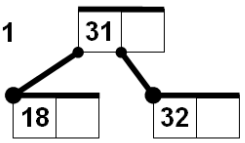
Always replacing deleted inner key by min in right subtree.

13. Vybudujte B-strom řádu 1 tak, že do prázdného stromu vložíte v uvedeném pořadí klíče 32, 18, 31, 59, 20, 23, 24, 36, 60, 58, 15, 57, 51, 17, 16, 26, 42, 21, 43, 12.
Dále tento strom zrušte, a to tak, že jednotlivé klíče klíče odstraníte v pořadí 23, 31, 26, 15, 24, 42, 17, 36, 20, 43, 16, 32, 18, 59, 21, 51, 60, 12, 58, 57.
Nakreslete strom po každé operaci Insert a Delete.
Řešení

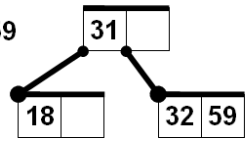
Insert 32

Insert 18

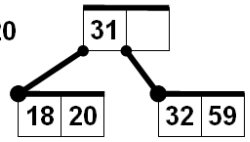
Insert 31



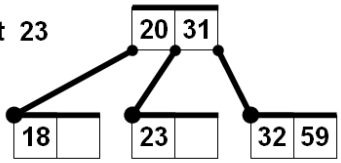
Insert 59



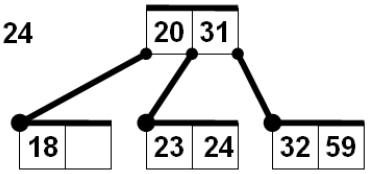
Insert 20



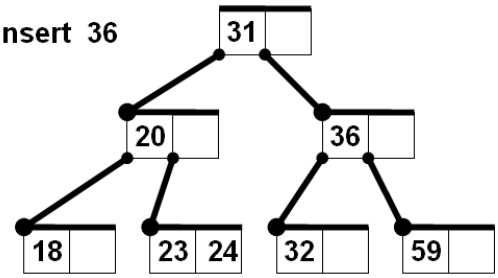
Insert 23



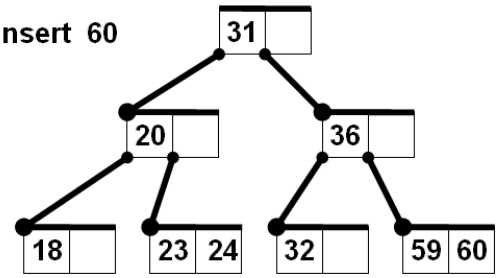
Insert 24



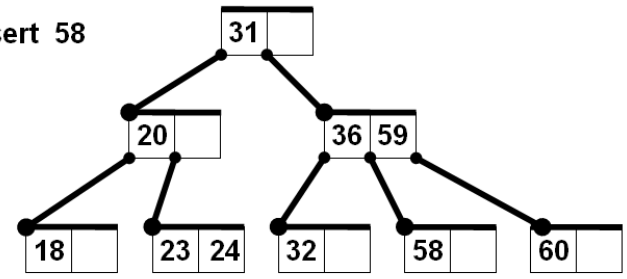
Insert 36



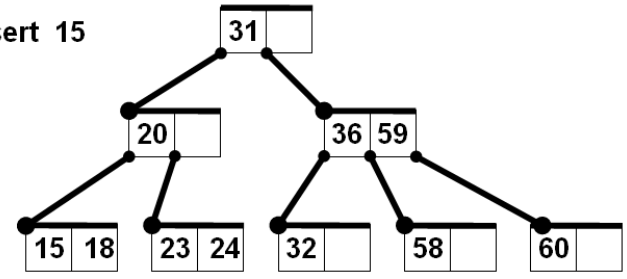
Insert 60



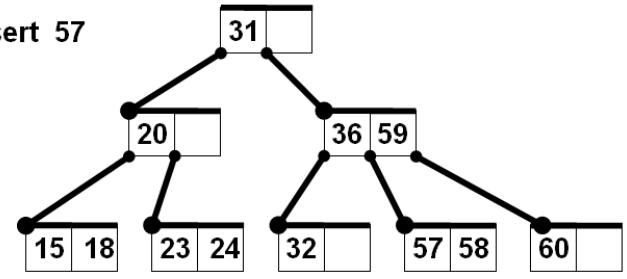
Insert 58



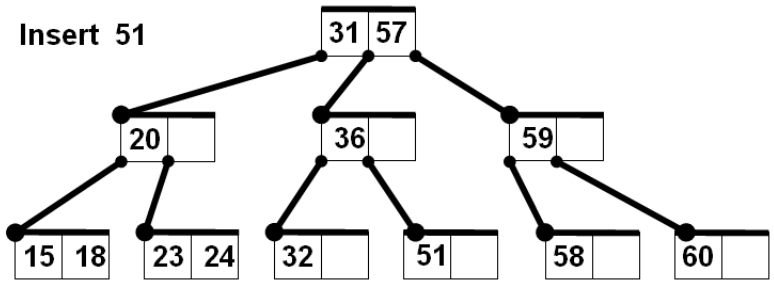
Insert 15



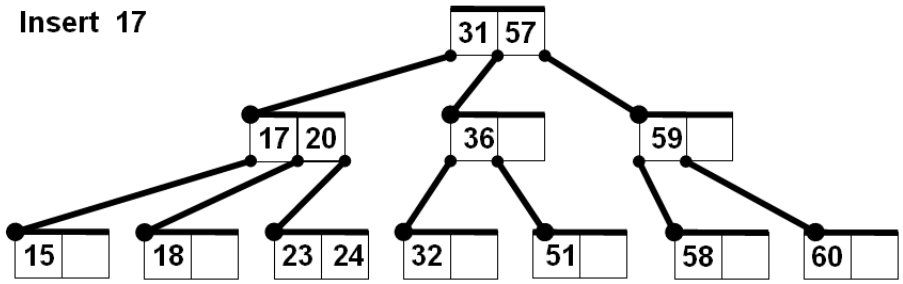
Insert 57



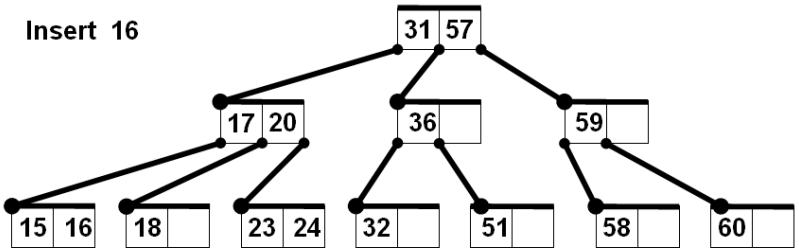
Insert 51



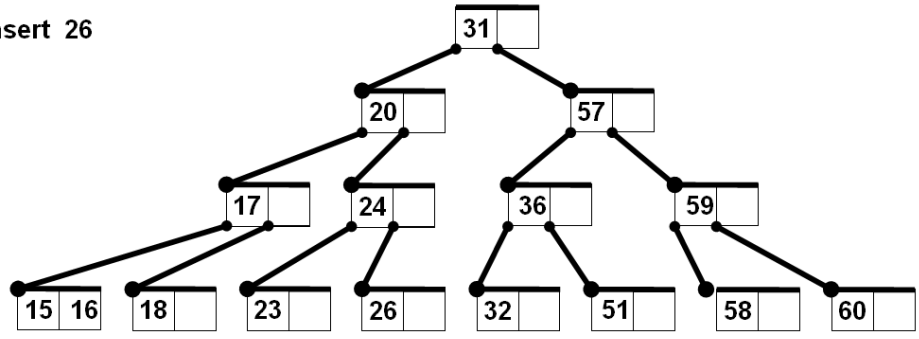
Insert 17



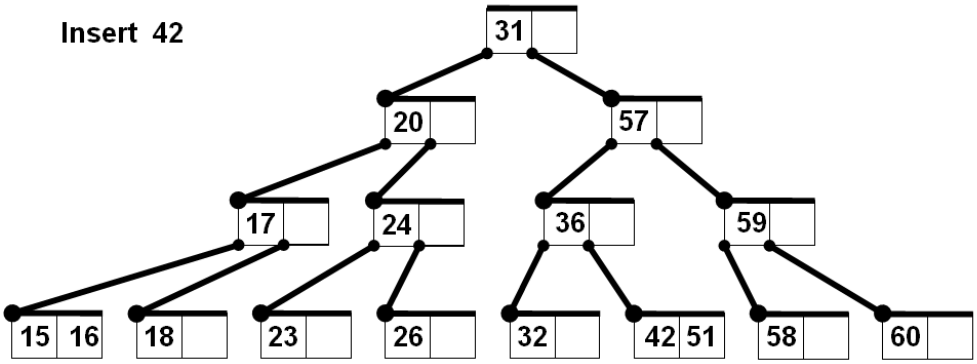
Insert 16



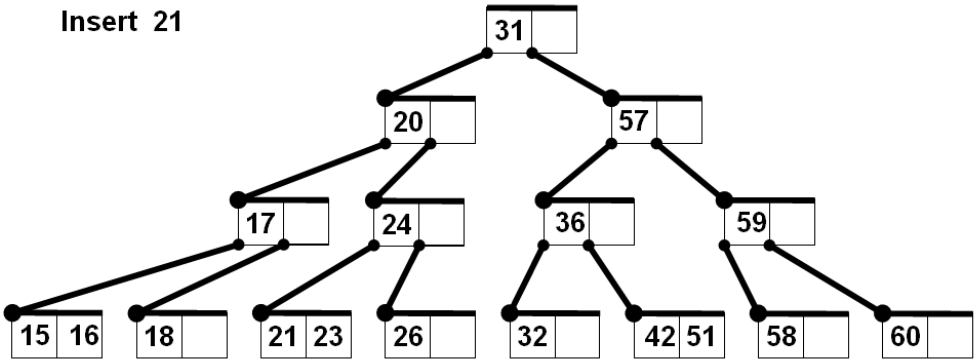
Insert 26



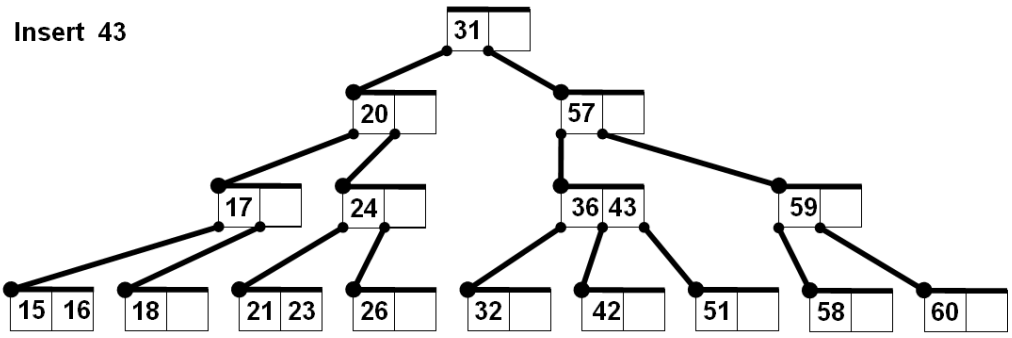
Insert 42



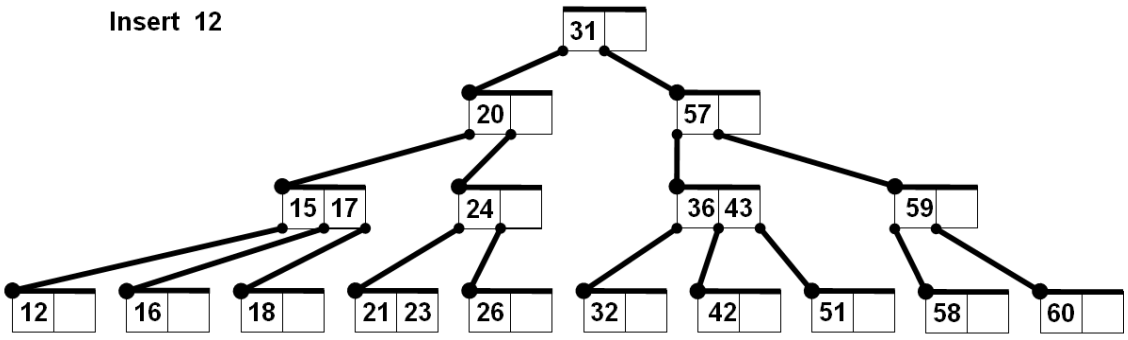
Insert 21



Insert 43

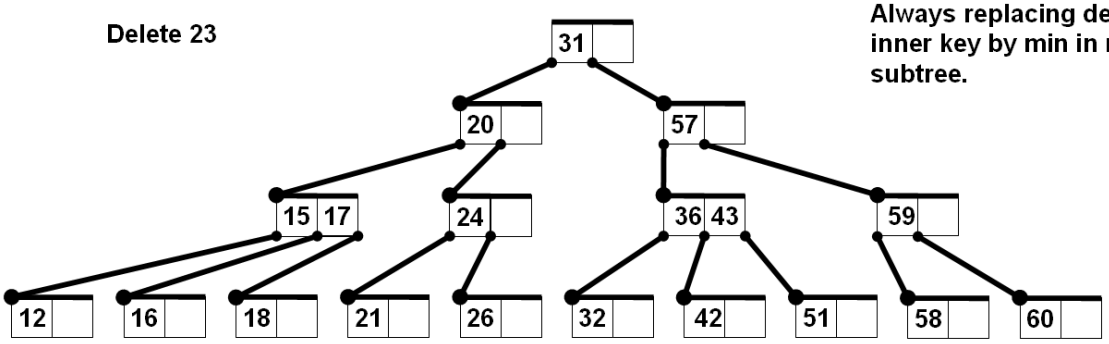


Insert 12

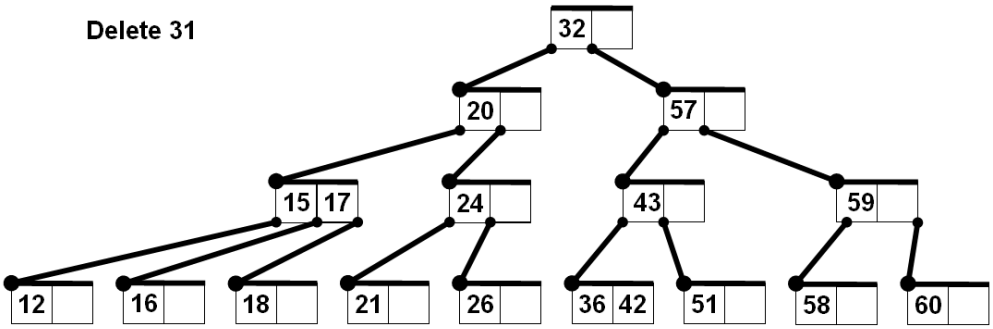


Delete 23

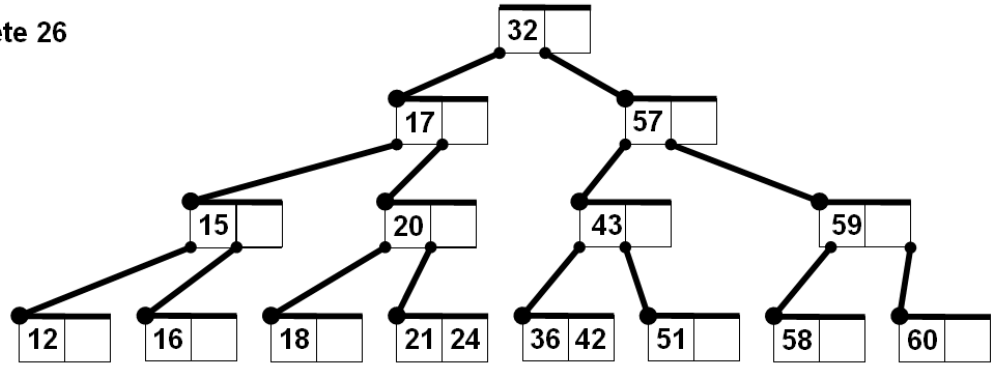
Always replacing deleted inner key by min in right subtree.



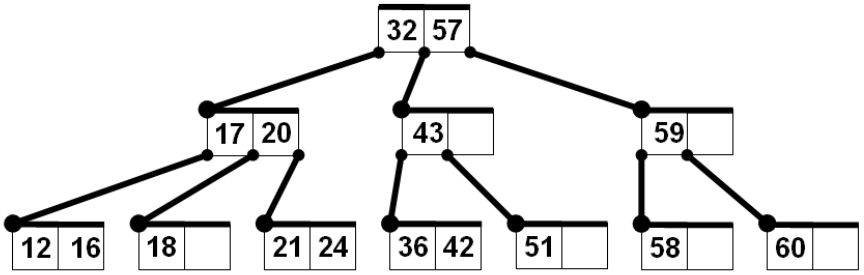
Delete 31



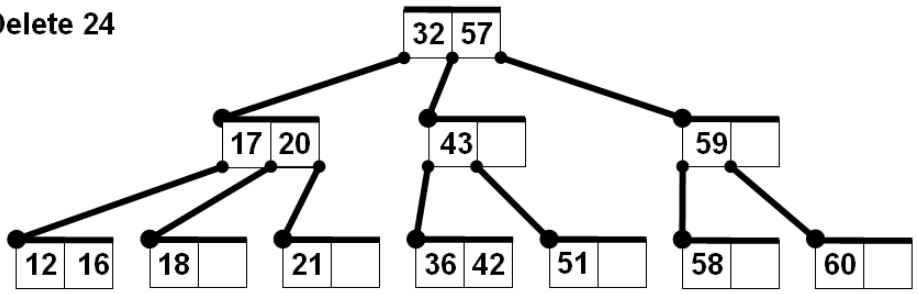
Delete 26



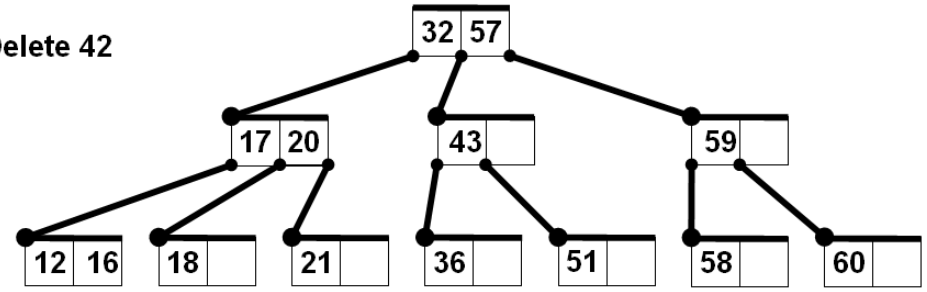
Delete 15



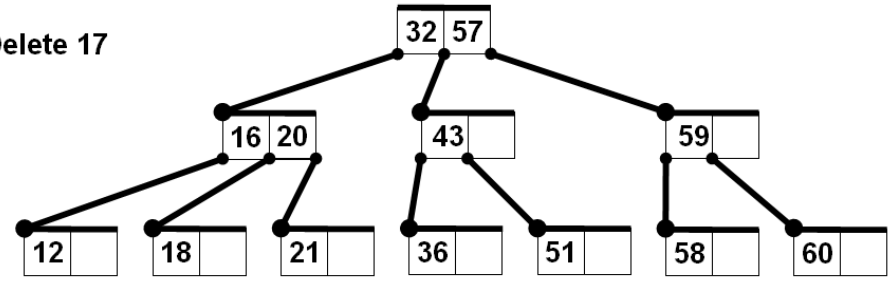
Delete 24



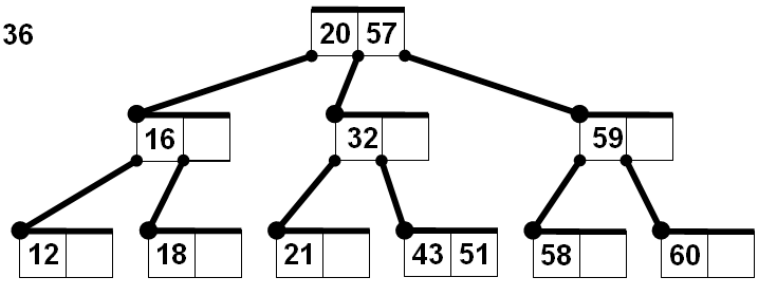
Delete 42



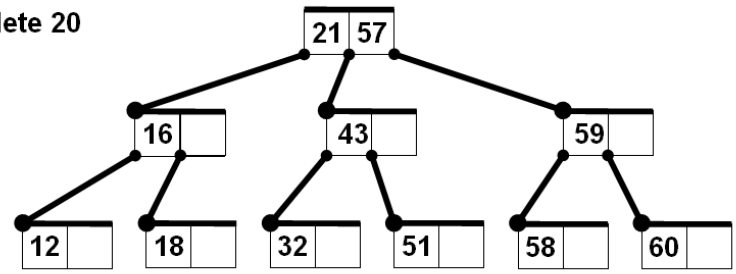
Delete 17



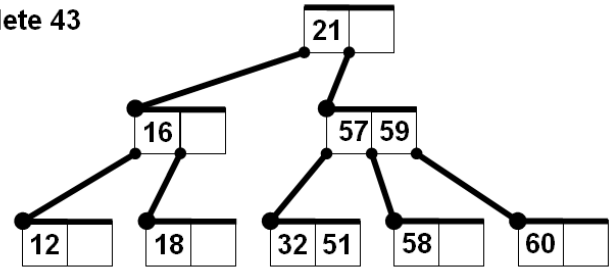
Delete 36



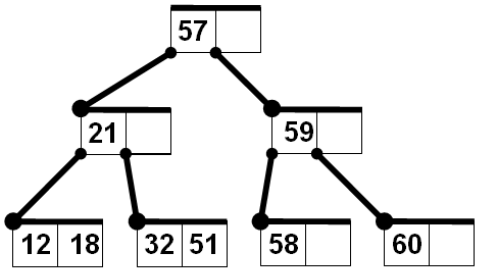
Delete 20



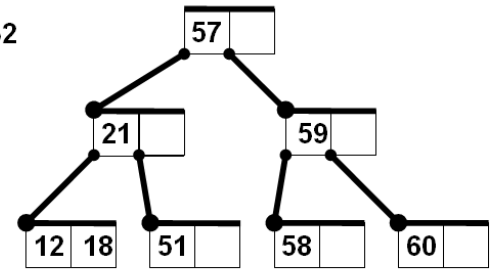
Delete 43



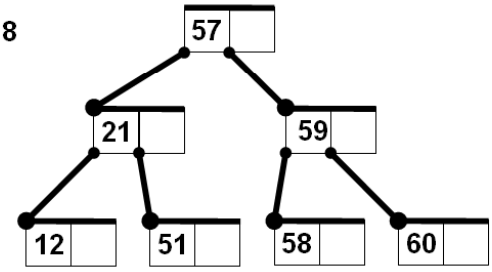
Delete 16



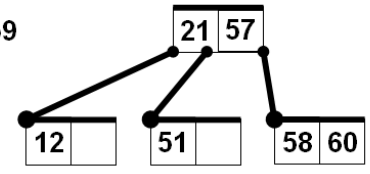
Delete 32



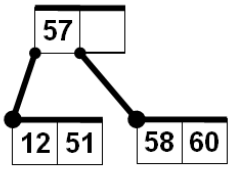
Delete 18



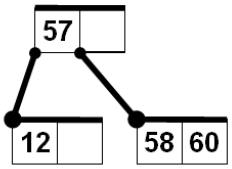
Delete 59



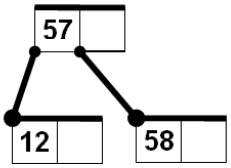
Delete 21



Delete 51



Delete 60



Delete 12

Delete 58

Delete 57 empty tree

14.
B-strom je řádu 5 a máme do něj umístit 1 000 000 klíčů. Jaký je maximální a minimální možný počet uzlů tohoto stromu? Jaká je maximální a minimální možná hloubka tohoto stromu?

Řešení Pokud jsou všechny uzly maximálně naplněny klíči, má každý uzel 10 klíčů a 11 potomků.
Počet klíčů v jednotlivých patrech je pak 10, 11*10, 11*11*10, 11*11*11*10 atd. Když má tento strom hloubku *h*, jeho počet klíčů je roven 10 * (1 + 11 +11*11 + ... + 11^*h*) = 10 * (11^(*h*+1) −1) /(11 −1) = 11^(*h*+1) −1.
Nejmenší *h*, pro které platí 11^(*h*+1) −1 ≥ 1000000, je *h*=5, strom tedy bude mít 6 pater včetně kořene.
Pokud by měly být všechny uzly minimálně zaplněny, tj. počet uzlů by se maximalizoval, pak zopakujeme předchozí výpočet s jinými parametry, jmenovitě:
Počet klíčů v jednotlivých patrech je pak 1, 6*5, 6*6*5, 6*6*6*5 atd. Když má tento strom hloubku *h*, jeho počet klíčů je roven 5 * (1 + 6 +6*6 + ... + 6^*h*) − 4 = 5 * (6^(*h*+1) −1) /(6 −1) − 4 = 6^(*h*+1) −5.
Nejmenší *h*, pro které platí 6^(*h*+1) −5 ≥ 1000000, je *h*=7, strom tedy bude mít 8 pater včetně kořene.

15.
Řešte předchozí úlohu pro obecnou hodnotu řádu B-stromu *k*. a pro daný počet klíčů *n*.

Insert sort

16.
Nahradíme-li Selection sort Insert sort-em, pak asymptotická složitost řazení
a) nutně klesne pro každá data
b) nutně vzroste pro každá data
c) může klesnout pro některá data
d) může vzrůst pro některá data
e) zůstane beze změny pro libovolná data

Řešení Tady je patrně nutné si pamatovat, že asymptotická složitost Selection sortu je $\Theta(n^2)$, zatímco asymptotická složitost Insert sortu je $O(n^2)$. To znamená, že existují případy, kdy Insert sort svá data seřadí v čase asymptoticky kratším než n^2. Na těchto datech však asymptotická složitost Selection sortu zůstane stejná, protože ta je rovna $\Theta(n^2)$ vždy. To znamená, že při přechodu od Selection sortu k Insert sortu nad týmiž daty může asymptotická složitost někdy klesnout, což odpovídá variantě c).

17.
Čtyři prvky řadíme pomocí Insert Sortu. Celkový počet vzájemných porovnání prvků je
a) je alespoň 4
b) je nejvýše 4
c) je alespoň 3
d) může být až 10
e) je vždy roven 4*(4−1)/2 = 6.

Řešení Nejmenší možný počet testů je:
při zařazení 2. prvku - 1,
při zařazení 3. prvku - 1,
při zařazení 4. prvku - 1.
Tedy celkem 3 porovnání, pokud je posloupnost prvků rostoucí již před seřazením.
Největší možný počet testů je
při zařazení 2. prvku - 1,
při zařazení 3. prvku - 2,
při zařazení 4. prvku - 3.
Tedy celkem 6 porovnání, pokud je posloupnost prvků klesající před seřazením. Pro 4 prvky tedy Insert sort vykoná 3 až 6 porovnání prvků, podle toho, jaká má data. S tímto zjištěním koresponduje pouze varianta c).

18.
V určitém problému je velikost zpracovávaného pole s daty rovna rovna $3n^2 \cdot \log(n^2)$ kde *n* charakterizuje velikost problému. Pole se řadí pomocí Insert sort-u.

Asymptotická složitost tohoto algoritmu nad uvedeným polem je tedy

a) $O(n^2 \cdot \log(n))$
b) $O(n^2 \cdot \log(n^2))$
c) $O(n^4 \cdot \log^2(n))$
d) $O(n^2 \cdot \log(n) + n^2)$
e) $O(n^2)$

Řešení Máme-li velikost pole rovnu k, Insert sort jej zpracuje v čase $O(k^2)$. Velikost našeho pole je $k = 3n^2 \cdot \log(n^2)$, takže bude zpracováno v čase $O(k^2) = O((3n^2 \cdot \log(n^2))^2) = = O(9n^4 \cdot \log^2(n^2)) = O(9n^4 \cdot 4 \cdot \log^2(n)) = O(n^4 \cdot \log^2(n))$. Platí tedy možnost c).

19.

Insert sort řadí (do neklesajícího pořadí) pole o *n* prvcích, kde jsou stejné všechny hodnoty kromě poslední, která je menší. Jediné nesprávné označení asymptotické složitosti výpočtu je

a) $O(n)$
b) $\Theta(n)$
c) $\Omega(n)$
d) $O(n^2)$
e) $\Omega(n^2)$

Řešení Zpracování prvních (n–1) prvků pole proběhne v čase úměrném (n–1), protože žádný prvek se nebude přesouvat a zjištění toho, že se nebude přesouvat, proběhne v konstantním čase. Poslední prvek se pak přesune na začátek, což opět proběhne v čase úměrném *n*. Celkem tedy veškeré řazení proběhne v čase úměrném *n*. Platí ovšem

$konst \cdot n \in O(n)$
$konst \cdot n \in \Theta(n)$
$konst \cdot n \in \Omega(n)$
$konst \cdot n \in O(n^2)$

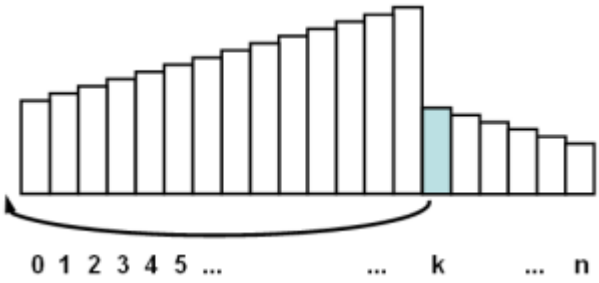
Jediná nepravdivá varianta je e).

20.

Insert sort řadí pole seřazené sestupně. Počet porovnání prvků za celý průběh řazení bude

a) n
b) n–1
c) n*(n–1)/2
d) n*n
e) $\log_2(n)$

Řešení V k-tém kroku řazení je nutno prvek na pozici k zařadit na jemu odpovídající místo v seřazeném úseku nalevo od k (předpokládáme pole indexované 0.. n–1). Tento prvek je však díky původnímu sestupnému seřazení menší než všecny prvky nalevo od něj, tudíž musí být zařazen na úplný začátek, což znamená, že bude porovnán se všemi prvky nalevo od něj jichž je právě k. Viz obrázek.
V k tém kroku tedy bude provedeno k porovnání.



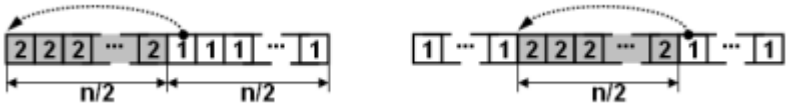
Celkový počet porovnání tedy je 0 + 1 + 2 + 3 + ... + (n–1) = n*(n–1)/2. Platí varianta c).

21.

Insert sort řadí (do neklesajícího pořadí) pole o *n* prvcích, kde v první polovině pole jsou pouze dvojky, ve druhé polovině jsou jen jedničky. Jediné nesprávné označení asymptotické složitosti výpočtu je

a) $\Theta(n)$
b) $\Omega(n)$
c) $O(n^2)$
d) $\Theta(n^2)$
e) $\Omega(n^2)$

Řešení Na obrázku máme znázorněnou situaci před prvním přesunem jedničky první a pak situaci před některým dalším přesunem jedničky.



V každém kroku je nutno jednu jedničku přesunout o $n/2$ pozic doleva a těchto kroků bude $n/2$ (protože jedniček je $n/2$). Počet provedených operací bude tedy úměrný hodnotě výrazu $n/2 \cdot n/2 = n^2/4 \in \Theta(n^2)$. Z toho, že $n^2/4 \in \Theta(n^2)$, vyplývá bezprostředně také, že $n^2/4 \in O(n^2)$, $n^2/4 \in \Omega(n^2)$, $n^2/4 \in \Omega(n)$ a navíc $n^2/4 \notin O(n)$. Pravdivé jsou tedy varianty b) – d), nepravdivá je jedině varianta a).

22.
V poli velikosti n mají všechny prvky stejnou hodnotu kromě posledního, který je menší.
Zatrhněte všechny možnosti, které pravdivě charakterizují asymptotickou složitost Insert Sortu (třídění vkládáním) pracujícího nad tímto konkrétním polem.
$O(n)$ $\Omega(n)$ $\Theta(n)$ $O(n^2)$ $\Omega(n^2)$ $\Theta(n^2)$

Řešení Insert Sort postupuje od začátku pole a každý prvek zařadí do již seřazeného počátečního úseku pole. protože ale jsou všechny prvky stejně velké, zařazení každého prvku se redukuje na to, že bude ponechán na místě. To je operaces konstantní časovou složitostí, takže prvních n–1 prvků bude zpracováno v čase $\Theta(n)$. Poslední prvek v poli je sice menší, takže patrně nebude zpracován v konstantním čase. Zpracování každého jednotlivého prvku při řazení Insert Sort-em má však složitost $O(n)$, takže celková složitost řazení bude $\Theta(n) + O(n) = \Theta(n)$. V zadání je tedy nutno zatrhnout možnosti $O(n)$, $\Omega(n)$, $\Theta(n)$, $O(n^2)$.

23.
Pole délky n obsahuje hodnoty n, 1, 2, 3, 4, ... , n–2, n–1, v uvedeném pořadí. Počet porovnání dvojic čísel, které provede Insert sort při řazení tohoto pole, je

a) $2 \cdot \log_2(n) - 1$
b) $n+3$
c) $2n - 3$
d) $n(n-1)/2 + 2$
e) $2n^2 - n - 1$

24a.
Pole délky n obsahuje všechna čísla 1, 2, ..., n, přitom první polovina pole obsahuje všechna lichá čísla seřazená vzestupně, druhá polovina obsahuje všechna sudá čísla také seřazená vzestupně. Pomocí Insert sortu řadíme toto pole do vzestupného pořadí. Čas potřebný na seřazení tohoto pole je

a) konstantní
b) úměrný n
c) úměrný $n \cdot \log(n)$
d) úměrný n^2
e) úměrný n^3

24b.
Pomocí Insert sortu řadíme pole délky n o do neklesajícího pořadí. První třetina a třetí třetina pole obsahují čísla v rozmezí 100 až 1000, druhá třetina pole obsahuje čísla v rozmezí 20 až 50. Čas potřebný na seřazení tohoto pole je

a) konstantní
b) úměrný n
c) úměrný $n \cdot \log(n)$
d) úměrný n^2
e) úměrný n^3

25.
Insert sort řadí do neklesajícího uspořádání posloupnost n různých kladných celých čísel. Na vstupu je posloupnost
2, 4, 6, 8, 10, …, n–6, n–4, n–2, n, n–1, n–3, n–5, n–7, …, 7, 5, 3, 1.
Určete přesně, kolik porovnání dvojic čísel provede Insert sort při řazení těchto dat a také vyjádřete asymptotickou složitost tohoto procesu pomocí $O/\Theta/\Omega$ symboliky.

Selection sort

26.
V určitém problému je velikost zpracovávaného pole s daty rovna rovna $2n^3 \cdot \log(n)$ kde n charakterizuje velikost problému. Pole se řadí pomocí Selection sort-u. Asymptotická složitost tohoto algoritmu nad uvedeným polem je tedy

a) $\Theta(n^2)$
b) $\Theta(n^6 \cdot \log^2(n))$
c) $\Theta(n^3 \cdot \log(n))$
d) $\Theta(n^3 \cdot \log(n) + n^2)$
e) $\Theta(n^5 \cdot \log(n))$

Řešení Máme-li velikost pole rovnu k, Selection sort jej zpracuje v čase $\Theta(k^2)$. Velikost našeho pole je $k = 2n^3 \cdot \log(n)$, takže bude zpracováno v čase $\Theta(k^2) = \Theta((2n^3 \cdot \log(n))^2) =$
$= \Theta(4n^6 \cdot \log^2(n)) = \Theta(n^6 \cdot \log^2(n))$.
Platí tedy možnost b).

27.
Pole n různých prvků je uspořádáno od druhého prvku sestupně, první prvek má nejmenší hodnotu ze všech prvků v poli.
Zatrhněte všechny možnosti, které pravdivě charakterizují asymptotickou složitost Selection Sortu (třídění výběrem) pracujícího nad tímto konkrétním polem.
$O(n)$ $\Omega(n)$ $\Theta(n)$ $O(n^2)$ $\Omega(n^2)$ $\Theta(n^2)$

Řešení Pro výběr aktuálního minima musí Select Sort pokaždé zkontrolovat všechny prvky od aktuálního až do konce pole, takže jeho složitost je je vždy právě $\Theta(n^2)$ nezávisle na datech v poli. V zadání je tedy nutno zatrhnout možnosti $\Omega(n)$, $O(n^2)$, $\Omega(n^2)$, $\Theta(n^2)$

28.
Společná vlastnost Insert sort-u, Select sort-u a Bubble sort-u je následující:

a) všechny tyto algoritmy jsou vždy stabilní
b) všechny tyto algoritmy jsou vždy nestabilní
c) všechny tyto algoritmy mají složitost $O(n \cdot \log_2(n))$
d) všechny tyto algoritmy mají složitost $O(n^2)$
e) všechny tyto algoritmy mají složitost $\Theta(n^2)$

Řešení První z uvedených řazení — Insert sort — lze pouhou záměnou ostré a neostré nerovnosti v testu vnitřního cyklu učinit stabilním či nestabilním. Tvrzení variant a) a b) o stabilitě/nestabilitě za všech okolností tak patrně neplatí. Všechny zmíněné algoritmy v nejhorším případě řadí v čase úměrném n^2, funkce $n \cdot \log_2(n)$ roste asymptoticky pomaleji než funkce n^2, tudíž varianta c) rovněž neplatí. Insert sort může řadit v čase úměrném n, má-li vhodná data, takže tvrdit o něm, jako ve variantě e), že pokaždé řadí v čase úměrném n^2, není korektní. Zbývá tak jen správná varianta d).