Níže uvedené úlohy představují přehled otázek, které se vyskytly v tomto nebo v minulých semestrech ve cvičení nebo v minulých semestrech u zkoušky. Mezi otázkami semestrovými a zkouškovými není žádný rozdíl, předpokládáme, že připravený posluchač dokáže zdárně zodpovědět většinu z nich.

Tento dokument je k dispozici ve variantě převážně s řešením a bez řešení.

Je to pracovní dokument a nebyl soustavně redigován, tým ALG neručí za překlepy a jazykové prohřešky, většina odpovědí a řešení je ale pravděpodobně správně :-).

---------- STACK ----------

1.
Se zásobníkem Z provedeme operace **init(), push(a), pop(), push(b)**. Pak Z obsahuje prvky (počínaje vrcholem zásobníku):

a) a b
b) b a
c) a
d) b
e) žádný prvek

2.
The following operations were performed on a stack S: **init(), push(a), pop(), push(b)**.
The resulting contents of S is then (list starts at the top of S):

a) a b
b) b a
c) a
d) b
e) no element

3.
On the given stack S operations **init(), push(z), push(w), pop()** have been performed. Now S contains the elements (list starts at the stack top):

f) w
g) z
h) w z
i) z w
j) 0 z w

4.
Zásobníkový automat je (zjednodušeně vzato) zařízení, které čte znak po znaku vstupní řetězec a po každém přečteném znaku provede nějakou operaci se zásobníkem. Než automat začne číst řetězec, je zásobník prázdný. Kromě toho jsou specifikována pravidla, jaké operace se zásobníkem se mají provést po přečtení určitého znaku vstupního řetězce.
Předpokládejme, že ve vstupním řetězci mohou být pouze znaky A ... Z a že na zásobník lze ukládat celá čísla. Předpokládejme dále, že zásobník má neomezenou kapacitu a že při pokusu o operaci POP na prázdném zásobníku je ohlášena chyba (samotný prázdný zásobník chybu neznamená).
Pravidla pro práci automatu lze zaznamenat tabulkou

čtený znak	operace
A	PUSH (1)
B	POP()
jiný	žádná

Určete, jaký bude obsah zásobníku po přečtení řetězce
a) AAB
b) ABAB

c) BAA
d) ABAABAAAB

5.
Uvažujme zásobníkový automat **A2** s odlišnou tabulkou operací

čtený znak	operace
A	PUSH (0)
B	PUSH(POP() + 1)
C	POP()
jiný	žádná

Určete, jaký bude obsah zásobníku po přečtení řetězce
a) AABB
b) AABBC
c) AABCB
d) ABABBABBB

6.
Automat **A2** z předchozí úlohy přečetl určitý řetězec *r* a má nyní prázdný zásobník. Co lze říci o počtu znaků A a C v řetězci *r*?

7.
Navrhněte takový vstupní řetězec pro automat A2 z předchozí úlohy, aby po jeho přečtení zásobník obsahoval právě hodnoty 4 3 2 1 (vrchol zásobníku je vpravo). Kolik řešení má tato úloha?

---------- QUEUE ----------

1.
Nad frontou F provedeme operace **init(), ins_last(a), ins_last(b), del_front()**. Potom F obsahuje prvky (počínaje čelem fronty):

a) a b
b) b a
c) a
d) b
e) žádný prvek

2.
On the given queue Q operations **init(), insLast(x), insLast(y), delFront()** have been performed. Now Q contains the elements (list starts at the queue front):

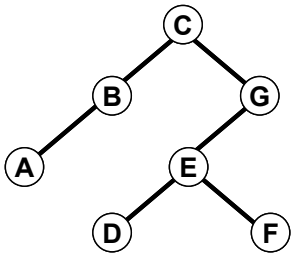
a) x
b) y
c) x y
d) y x
e) 0 x y

3.
Implementujte cyklickou frontu v poli pole[délka]. Dokud není pole zcela naplněno, musí být fronta stále funkční, tzn. musí být možné do ní stále přidávat i z ní ubírat.
Narazí-li konec nebo začátek fronty na konec nebo začátek pole, neposunujte všechny prvky v poli, zvolte výhodnější „cyklickou“ strategii, která zachová konstantní složitost operace Vlož a Vyjmi.

---------- BREADTH-FIRST SEARCH ----------

1.

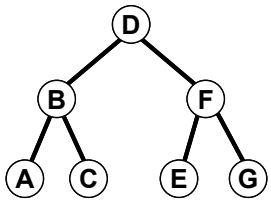
Strom na obrázku procházíme do šířky. V určitém okamžku jsou ve frontě následující uzly (s tím, že čelo fronty je vlevo):



a) BGA
b) GA
c) AG
d) AEG
e) GEA
f) AE

2.

Strom na obrázku procházíme do šířky. V určitém okamžku jsou ve frontě následující uzly (s tím, že čelo fronty je vlevo



a) D
b) DF
c) FB
d) BGE
e) BAC
f) ABCD

3.

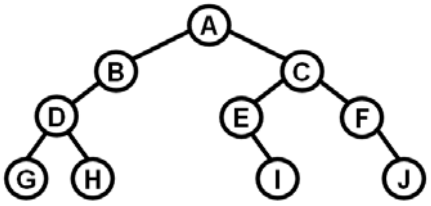
Zopakujme stručně princip procházení do šířky.

Krok 0. Vlož kořen do prázdné fronty

Dokud je fronta neprázdná, dělej

Krok 1. Vyjmi první prvek z fronty, a zpracuj ho.

Krok 2. Vlož do fronty všechny potomky právě vyjmutého listu.

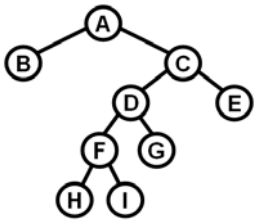


Projděte do šířky daný strom a před každým provedením kroku 1. zaznamenejte obsah fronty.

4.

Úlohou je rekonstruovat tvar pravidelného stromu, pokud známe průběžný obsah fronty. Dejme tomu, že před každým provedením kroku 1. v uvedeném postupu zaregistrujeme aktuální obsah fronty. Získáme posloupnost (předpokládáme, že čelo fronty je vlevo):

A
BC
C
DE
EFG
FG
GHI
HI
I



5.

Formulujte obecný algoritmus, jak z dané posloupnosti obsahů fronty (jako v předchozí úloze) rekonstruovat pravidelný strom.

6.

Přeformulujte předchozí algoritmus, aby na základě posloupnosti obsahů fronty rekonstruoval původní strom, kterým v tomto případě je obecný BVS.

7.

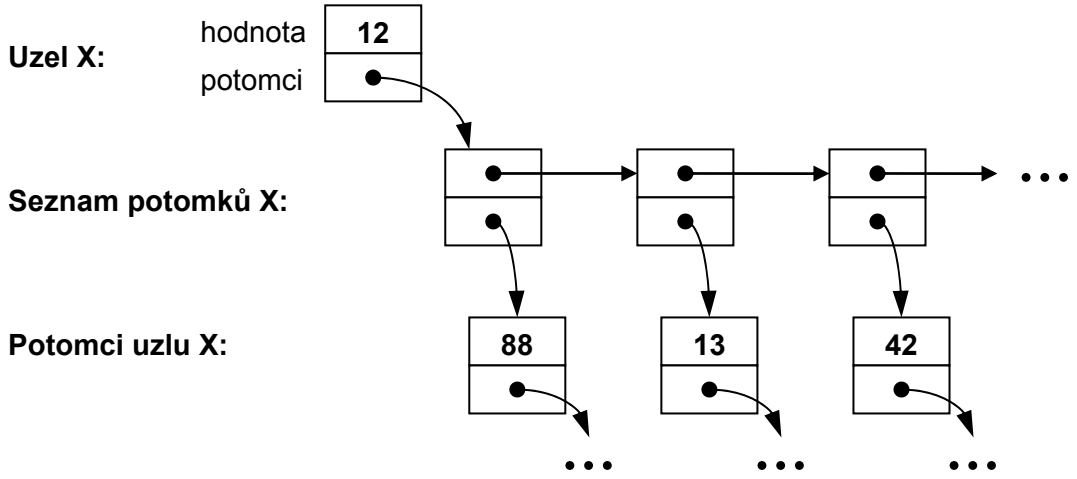
Vypište hodnoty uzlů daného binárního stromu tak, že budete postupovat „po patrech“, tj. nejprve vypíšete hodnoty všech uzlů s hloubkou 0, pak hodnoty všech uzlů s hloubkou 1, pak hodnoty všech uzlů s hloubkou 2, atd. Nápověda: použijte datový typ fronta.

---------- **GENERAL TREE** ----------

1.
Daný obecný strom má *n* uzlů. Kolik má hran? Zdůvodněte.

2.
Napište rekurzívně a nerekurzívně funkci, která projde n-ární strom (každý uzel může mít až n podstromů. Odkazy na podstromy jsou uloženy v každém uzlu v poli délky n). Datová struktura uzlu (1 bod).

3.
Napište funkci, která vytvoří kopii obecného kořenového stromu. Ten je reprezentován takto:
Uzel X stromu obsahuje dvě složky: **hodnota** a **potomci**. Složka **potomci** je ukazatel na zřetězený seznam obsahující jednotlivé ukazatele na potomky uzlu X. Nemá-li uzel X žádné potomky, složka **potomci** ukazuje na null. Každý prvek seznamu potomků obsahuje jen ukazatel na potomka X a ukazatel na další prvek v seznamu potomků. Strom je obecný, potomci libovolného uzlu nejsou nijak uspořádáni podle svých hodnot.



4.
Napište rekurzívně a nerekurzívně funkci, která projde n-ární strom (každý uzel může mít až n podstromů. Odkazy na podstromy jsou uloženy v každém uzlu v poli délky n). Datová struktura uzlu (1 bod). Rekurzívní verze (2b), nerekurzívní (3b.)

---------- **BIN TREE** ----------

1.
Daný pravidelný (kořenový) binární strom má n uzlů. Kolik má listů? Zdůvodněte.

2.
Jsou dány dva kusy papíru. Některý z nich rozstříhneme na dva kusy, některý ze všech kusů rozstříhneme opět na dva kusy, atd. Kolik kusů celkem vznikne, když bylo rozstříženo celkem *k* kusů papíru (nezávisle na jejich velikosti)?

3.
Turnaje ve stolním tenisu, který se hraje vylučovacím způsobem, se zúčastnilo celkem 19 hráčů. Šest vylosovaných hráčů muselo sehrát předkolo, z nějž do turnaje postoupili tři hráči. Nakreslete strom reprezentující možný průběh turnaje a určete počet jeho vnitřních uzlů tj. počet utkání, která byla během turnaje sehrána.
Kolik utkání včetně předkola by bylo nutno sehrát v turnaji, jehož se by se účastnilo147 hráčů?

4.
Při volání rekurzivní funkce f(n) vznikne binární pravidelný ideálně vyvážený strom rekurzivního volání s hloubkou $\log_2(n)$. Asymptotická složitost funkce f(n) je tedy

a) $\Theta(\log_2(n))$
b) $\Theta(n \cdot \log_2(n))$
c) $O(n)$
d) $O(\log_2(n))$
e) $\Omega(n)$

5.
Při volání rekurzivní funkce f(n) vznikne binární pravidelný ideálně vyvážený strom rekurzivního volání s hloubkou n. Asymptotická složitost funkce f(n) je tedy

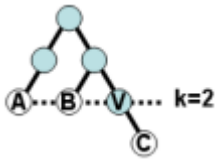
a) $\Theta(n)$
b) $O(n)$
c) $\Theta(\log_2(n))$
d) $\Omega(2^n)$
e) $O(n!)$

6.
V obecném binárním stromu s n uzly, který není vyhledávací, hledáme klíč s maximální hodnotou. Když to uděláme efektivně, bude složitost této operace

a) O(1)
b) $\Theta(n)$
c) $\Theta(n^2)$
d) O(log(n))
e) $\Theta(n \cdot \log(n))$

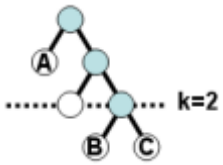
7.
Je dána konstanta k taková, že hloubka každého listu v daném binárním stromu je větší nebo rovna k. Pro hloubku každého vnitřního uzlu platí

a) je menší než k
b) je menší nebo rovna k–2
c) je větší než k/2
d) je menší nebo rovna $\log_2(k)$
e) nic z předchozího



8.
Je dána konstanta k taková, že hloubka každého vnitřního uzlu v daném binárním stromu je menší nebo rovna k. Pro hloubku každého listu platí

a) je rovna k+1
b) je také menší nebo rovna k
c) je menší nebo rovna k+1
d) je větší než k–1
e) je menší nebo rovna $\log_2(k)$



9.
Obecný binární strom

a) má vždy více listů než vnitřních uzlů
b) má vždy více listů než vnitřních uzlů, jen pokud je pravidelný
c) má vždy méně listů než vnitřních uzlů
d) může mít více kořenů

e) může mít mnoho listů a žádné vnitřní uzly

10.
Binární strom má n uzlů. Šířka jednoho „patra“ (tj. počet uzlů se stejnou hloubkou) je tedy
a) nejvýše $\log(n)$
b) nejvýše $n/2$
c) alespoň $\log(n)$
d) alespoň $n/2$
e) nejvýše n

11.
Daný binární strom má tři listy. Tudíž
a) má nejvýše dva vnitřní uzly
b) počet vnitřních uzlů není omezen
c) všechny listy mají stejnou hloubku
d) všechny listy nemohou mít stejnou hloubku
e) strom je pravidelný

12.
Binární strom má hloubku 2 (hloubka kořene je 0). Počet listů je
a) minimálně 0 a maximálně 2
b) minimálně 1 a maximálně 3
c) minimálně 1 a maximálně 4
d) minimálně 2 a maximálně 4

13.
Binární strom má 2 vnitřní uzly. Má tedy
a) minimálně 0 a maximálně 2 listy
b) minimálně 1 a maximálně 3 listy
c) minimálně 1 a maximálně 4 listy
d) minimálně 2 a maximálně 4 listy

14.
Algoritmus A provádí průchod v pořadí inorder binárním vyváženým stromem s n uzly a v každém uzlu provádí navíc další (nám neznámou) akci, jejíž složitost je $\Theta(n^2)$.
Celková asymptotická složitost algoritmu A je tedy
a) $\Theta(n)$
b) $\Theta(n^2)$
c) $\Theta(n^3)$
d) $\Theta(n^2+\log_2(n))$
e) $\Theta(n^2 \cdot \log_2(n))$

15.
Algoritmus A provede jeden průchod binárním stromem s hloubkou n. Při zpracování celého k-tého „patra“ (=všech uzlů s hloubkou k) provede $k+n$ operací. Operační (=asymptotická) složitost algoritmu A je tedy
a) $\Theta(k+n)$
b) $\Theta((k+n)\cdot n)$
c) $\Theta(k^2+n)$
d) $\Theta(n^2)$
e) $\Theta(n^3)$

16.
Navrhněte strukturu binárního stromu s n prvky tak, aby hloubka stromu nebyla vyjádřena výrazem ani $\Theta(\log(n))$ ani $\Theta(n)$, ale výrazem $\Theta(\sqrt{n})$.

17.
Představme si, že máme navštívit všechny uzly v ideálně vyváženém stromu se 7 uzly. Můžeme chodit podle každé hrany tam i zpět, pohyb podél jedné hrany mezi dvěma uzly trvá právě jednu sekundu.

a) Jak dlouho se budeme ve stromu pohybovat, máme-li (celkem přirozeně) začít i skončit v kořeni?
b) Změní se nějak doba průchodu stromem v případě že strom bude obsahovat stejný počet uzlů ale bude maximálně nevyvážený?
c) Jaká bude doba průchodu obecně v případě, že strom bude obsahovat n uzlů?

18.
Uvažujme opět ideálně vyvážený strom s n uzly. Naší úlohou bude opět navštívit všechny uzly, tentokrát ale bude strategie odlišná. Smíme se pohybovat pouze vpřed, z libovolného uzlu ale můžeme kdykoli skočit zpátky do kořene. Cesta podél jedné hrany trvá jednu sekundu, skok do kořenu je okamžitý, jeho doba trvání je zanedbatelná.

19.
Máme k dispozici 1024 sekundy tj. něco málo přes 1/4 hodiny. Jaká je maximální možný počet uzlů ve vyváženém stromu z předchozí úlohy, abychom jej stihli za tuto dobu projít? Jaká je maximální velikost maximálně nevyváženého stromu pro stejnou úlohu?

20.
Vyzkoušejme ještě jednu variantu předchozí úlohy: Co když traverzování jedné hrany trvá 10^(-8) sec a my máme na projití stromu 1/10 sec?

21.
Aritmetický výraz obsahující celá čísla, závorky a operace +,–,*,/ (celočíselné dělení) může být reprezentován jako pravidelný binární strom. Popište, jak takový strom obecně vypadá, navrhněte implementaci uzlu a napište funkci, jejímž vstupem bude ukazatel na kořen stromu a výstupem hodnota odpovídajícího aritmetického výrazu.

22.
Napište funkci, která z binárního stromu odstraní všechny listy. Předpokládejte, že každý uzel obsahuje ukazatel na svého rodiče.

23.
Deklarujte uzel binárního stromu, který bude obsahovat celočíselné složky
výška a **hloubka**. Ve složce **hloubka** bude uložena hloubka daného uzlu ve stromu, ve složce **výška** jeho výška. Výška uzlu X je definována jako vzdálenost od
jeho nejvzdálenějšího potomka (= počet hran mezi uzlem X a jeho nejvzdálenějším potomkem).
Napište funkci, která každému uzlu ve stromu přiřadí korektně hodnotu jeho hloubky a výšky.

---------- POST IN PREORDER ----------

1.
Všechny klíče uložené v uzlech binárního stromu vypíšeme v pořadí postorder. Strom má k listů. Ve vypsané posloupnosti se objeví

a) všechny klíče listů na prvních k pozicích
b) všechny klíče listů na posledních k pozicích
c) polovina klíčů listů na prvních k/2 pozicích a polovina na posledních k/2 pozicích
d) polovina klíčů vnitřních uzlů na prvních k/2 pozicích a polovina na posledních k/2 pozicích
e) klíče uzlů v žádné z předchozích konfigurací

2.
Klíče daného binárního vyhledávacího stromu vypíšeme v pořadí postorder. Vznikne posloupnost 2 9 4 5 1. Celkový počet uzlů v pravém podstromu kořene tohoto BVS je:

a) 0
b) 1
c) 2
d) 3
e) 4

3.
Klíče daného binárního vyhledávacího stromu vypíšeme v pořadí preorder.
Vznikne posloupnost 4 1 8 5 9. Celkový počet uzlů v pravém podstromu kořene tohoto BVS je:

a) 0
b) 1
c) 2
d) 3
e) 4

4.
Klíče daného binárního vyhledávacího stromu vypíšeme v pořadí postorder.
Vznikne posloupnost 4 3 7 9 6. Celkový počet uzlů v levém podstromu kořene tohoto BVS je:

a) 0
b) 1
c) 2
d) 3
e) 4

5.
The keys of the given BST are written out using the postorder scheme. The printed sequence is 4 3 7 9 6. The total number of nodes in the left subtree of the root is:

f) 0
g) 1
h) 2
i) 3
j) 4

6.
Klíče daného binárního vyhledávacího stromu vypíšeme v pořadí preorder. Vznikne posloupnost 2 9 4 5 1. Celkový počet uzlů v pravém podstromu kořene tohoto BVS je:

f) 0
g) 1
h) 2
i) 3
j) 4

7.
The keys of a binary search tree are printed out in using the postorder traversal scheme. The printed sequence is 2 9 4 5 1. Total number of the nodes in the right subtree of the tree root is

a) 0
b) 1
c) 2
d) 3
e) 4

8.
Všechny klíče uložené v uzlech binárního stromu vypíšeme v pořadí preorder. Strom má k listů. Ve vypsané posloupnosti budou

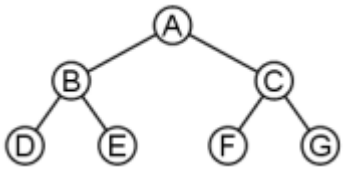
a) všechny klíče listů na prvních k pozicích
b) všechny klíče listů na posledních k pozicích
c) polovina klíčů listů na prvních k/2 pozicích a polovina na posledních k/2 pozicích
d) polovina klíčů vnitřních uzlů na prvních k/2 pozicích a polovina na posledních k/2 pozicích
e) klíče uzlů v žádné z předchozích konfigurací

9.
Výpis prvků binárního stromu v pořadí postorder provede následující:
a) vypíše prvky v opačném pořadí, než v jakém byly do stromu vloženy
b) pro každý podstrom vypíše nejprve kořen, pak obsah jeho levého a pak pravého podstromu
c) pro každý podstrom vypíše nejprve obsah levého podstromu kořene, pak obsah pravého podstromu a pak kořen
d) pro každý podstrom vypíše nejprve obsah pravého podstromu kořene, pak obsah levého podstromu a pak kořen
e) vypíše prvky stromu v uspořádání zprava doleva

10.
Výpis prvků binárního stromu v pořadí preorder provede následující:
a) vypíše prvky stromu ve stejném pořadí, v jakém byly do stromu vloženy
b) vypíše prvky stromu v uspořádání zleva doprava
c) pro každý podstrom vypíše nejprve kořen, pak obsah jeho levého a pak pravého podstromu
d) pro každý podstrom vypíše nejprve obsah levého podstromu kořene, pak obsah pravého podstromu a pak kořen
e) vypíše prvky stromu seřazené vzestupně podle velikosti

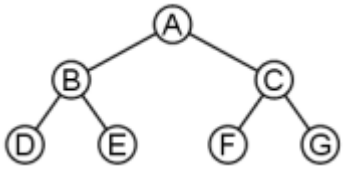
11.
Obsah uzlů daného stromu vypíšeme v pořadí postorder. Vznikne posloupnost

a) G F E D C B A
b) F C G D B E A
c) G C F A E B D
d) D E B F G C A



12.
Obsah uzlů daného stromu vypíšeme v pořadí preorder. Vznikne posloupnost

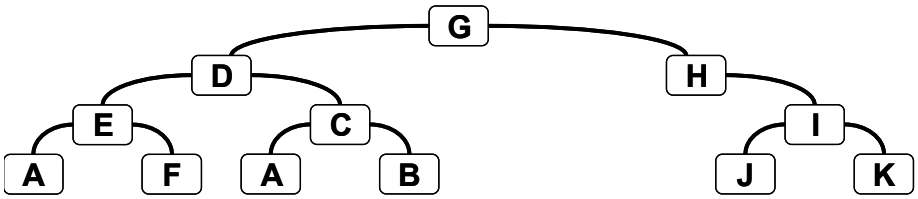
a) A B C D E F G H
b) D B E A F C G
c) A B D E C F G
d) D E F G B C A



13.

Vypište obsah jednotlivých uzlů daného stromu při průchodu v pořadí

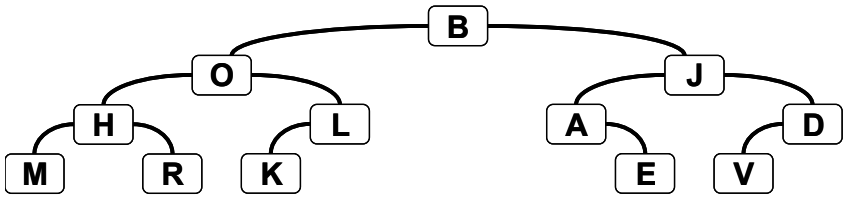
a) preorder
b) postorder



14.

Vypište obsah jednotlivých uzlů daného stromu při průchodu v pořadí

a) preorder
b) postorder



15.

Jaký musí mít binární strom tvar, aby
a) průchody In- Pre- a Postorder zpracovaly uzly ve stejném pořadí?
b) průchody In- Preorder zpracovaly uzly ve stejném pořadí?
c) průchody In- a Postorder zpracovaly uzly ve stejném pořadí?

16.

Při pruchodu daným stromem pořadí Inorder a pak Preorder získáme následující posloupnosti hodnot uložených v jeho jednotlivých (celkem devíti) uzlech:
Inorder: 45 71 98 47 50 62 87 3 79
Preorder: 50 47 71 45 98 62 3 87 79
a) Rekonstruujte tvar stromu.
b) Navrhněte a formulujte algoritmus, který z uvedených dvou posloupností pro libovolný strom rekonstruuje jeho podobu.

17.

Projděte binárním stromem a vypište obsah jeho uzlů v pořadí preorder bez použití rekurze, zato s využitím vlastního zásobníku.

18.

Řešte předchozí úlohu pro pořadí inorder a postorder.