Zkouškový Test PAL A Jméno ______________________________ sem+bon+pgm: ___

V následujících otázkách je správně někdy jen jedna, někdy více variant odpovědí. Správně zodpovězená otázka je taková, která určí správně všechny varianty odpovědí a je hodnocena uvedeným počtem bodů. Pokud některá varianta je určena chybně, otázka je hodnocena 0 body. Výjimku tvoří otázky 1., 2., 3., 14. — jedna chybně určená varianta znamená hodnocení polovičním počtem bodů.

1. (10 b.)

V jaké třídě složitosti je funkce `push` v následujícím programu vzhledem k velikosti pole `p` (budeme ji značit N)? Předpokládejte, že funkce `malloc` (funkce `malloc` naalokuje datovou strukturu velikosti argumentu a vrací na tuto strukturu ukazatel) a `free` (funkce `free` uvolní z paměti naalokovanou datovou strukturu, kterou dostane v argumentu) pracují v konstantním čase vzhledem k velikosti pole `p`. Dále předpokládejte, že velikost proměnné `N` vždy před a po provedení funkce `push` koresponduje s velikosti pole `p`. Proměnná `i` je vždy v intervalu `-1` až N.

a) $O(1)$
b) $O(\log(N))$
c) $O(N^2)$
d) $\Omega(1)$
e) $\Omega(\log(N))$
f) $\Omega(N^2)$
g) $\Theta(N^2)$
h) $\Theta(N)$

```
int N = 0;
int i = -1;
int * p;
void push (int key)
{
    if (++i >= N) {
         int m = N+1000;
         int * t = malloc(m*sizeof(int));
         int j;
         for(j = N-1; j>=0; j--, t[j] = p[j]);
         if (N != 0) free(p);
         N = m;
         p = t;
    }
    p[i] = key;
}
```

2. (10 b.)

V jaké amortizované třídě složitosti je funkce `push` z předchozího programu vzhledem k velikosti pole `p` (budeme ji značit N)? Předpokládejte, že funkce `malloc` a `free` pracují v konstantním čase vzhledem k velikosti pole `p`.

a) $O(1)$
b) $O(N*\log(N))$
c) $O(N^2)$
d) $\Omega(1)$
e) $\Omega(\log(N))$
f) $\Omega(N^2)$
g) $\Theta(N^2)$
h) $\Theta(N)$

3. (10 b.)

Pro matici incidence *I* úplného neorientovaného grafu s alespoň pěti uzly platí

a) *I* obsahuje řádek se samými nulami
b) *I* obsahuje sloupec se samými nulami
c) *I* obsahuje více nul než jedniček
d) *I* je čtvercová matice
e) *I* není čtvercová matice

4. (5 b.)

Mějme čtvercovou matici A racionálních čísel s rozměry n×n a s hodností n. Rozhodněte, zda platí některá z následujících tvrzení:

a) Matici A lze rozložit pomocí *LUP dekompozice* na matice L, U a P, kde platí, že PA=LU.
b) Pokud má matice A hodnost n, lze k ní najít inverzní matici v čase $O(n^3)$.
c) Pokud je matice A diagonální s hodností n, je po provedení *LUP dekompozice* na A matice P jednotková.
d) Pro matici A takovou, že k ní existuje inverzní matice, lze *LUP dekompozici* provést v čase $O(n^2)$.
e) Předpokládejme, že bychom reprezentovali matici A jako pole čísel s pohyblivou řádovou čárkou dle IEEE 754. Přesnost nalezení inverzní matice k A pomocí *LUP dekompozice* nelze zvýšit permutací některých řádků v A před provedením *LUP dekompozice*.

5. (5 b.)

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

Když má daný souvislý graf n uzlů a m hran, potom asymptotická složitost algoritmu BFS (prohledávání do šířky) je $\Theta(m)$, za předpokladu, že během provádění algoritmu máme přístup v konstantním čase ke každému právě zpracovávanému uzlu a ke každé právě zpracovávané hraně. Jaká bude asymptotická složitost tohoto algoritmu, pokud jediná reprezentace grafu, kterou budeme mít k dispozici, bude matice sousednosti tohoto grafu?

6. (5 b.)

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

Z binární haldy obsahující n prvků odstraníme $n/2$ nejmenších prvků. Asymptotická složitost této akce je

7. (5 b.)

Je dán NKA A_1, který byl převeden na DKA A_2 použitím standardního algoritmu převodu NKA na DKA. Žádné další úpravy se s A_2 neprováděly. Do A_2 se vloudily chyby. Je zapotřebí provést opravy

a) označit stav 0 jako koncový
b) doplnit přechod $\delta(13, c) = 02$
c) označit stav 13 jako koncový
d) doplnit přechod $\delta(02, b) = 13$
e) přidat stav 123 s prázdnými přechody

A_1	a	b	c	
0		1, 3		
1	2			F
2	0, 2		3	F
3		1		

A_2	a	b	c	
0		13		
13	2	1		
2	02		3	F
1	2			F
02	02		3	F
3		1		

8. (5 b.)

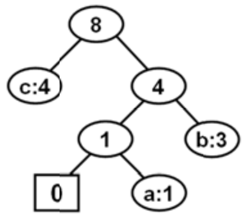
a) A ohlásí výskt R na první pozici v T,
b) A ohlásí výskyt R až na druhé pozici v T
c) A se nezastaví v T(= nenajde R)
d) A pracuje v lineárním čase vzhledem k délce T
e) A má 19 stavů

V textu $T = abababababab$ ... $abab$ o délce 2000 hledáme podřetězec R, který má od vzorku $P = xbbtb$ Levenshteinovu vzdálenost nejvýše 3. Používáme pro to odpovídající automat A. Platí

9. (5 b.)

a) *baa*
b) *aba*
c) *caba*
d) *bb*
e) *babc*

Po zakódování prvních osmi znaků zprávy a příslušných úpravách stromu má strom adaptivního Huffmanova kódu tvar znázorněný na obrázku. Po přečtení posloupnosti dalších níže uvedených znaků bude mít znak *c* dvoubitový kód.



10. (5 b.)

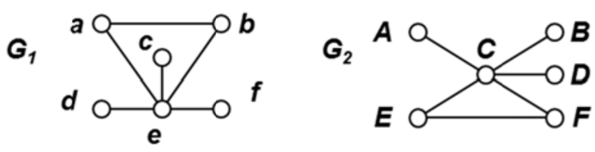
a) $2n^2 - 3n$
b) $3n$
c) $n^2 + n - 3$
d) $n^2 + 4n - 12$
e) $3n^2 - 18$

Graf W_n vznikne tak, že každý uzel kružnice C_n spojíme hranou s dalším přidaným uzlem x (W_n má tedy n+1 uzlů). Uvažujme pouze takové kostry W_n, jejichž právě dvě hrany leží v C_n. Určete, kolik různých (nikoli neizomorfních!) těchto koster existuje ve W_n pro $n \geq 3$.

11. (10 b.)

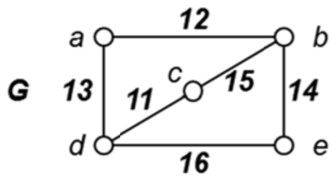
a) 0
b) 6
c) 8
d) 12
e) 24

Počet izomorfizmů mezi grafy G_1 a G_2 je



12. (5 b.)

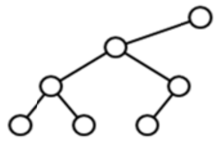
a) 0 b) 1 c) 2 d) 3 e) 4	V Kruskalově algoritmu reprezentujeme datovou strukturu Union-Find pomocí orientovaných stromů. Určete, jaká je hloubka konečného stromu v této struktuře po nalezení minimální kostry grafu G (kořen má hloubku 0). Předpokládáme, že při sjednocování stromů zařazujeme vždy menší strom pod kořen většího a nepoužíváme žádné další heuristiky pro zvýšení efektivity.



13. (5 b.)

a) hloubka T bude 3
b) hloubka T bude 4
c) tvar T zůstane beze změny
d) levý podstrom kořene bude mít jeden uzel
e) pravý podstrom kořene bude prázdný

Splay strom T má tvar daný obrázkem. Po vyhledání prvku s druhým nejmenším klíčem v T bude platit, že



14. (10 b.)

a) $T(\mathbf{x}) = f_1(f_2(f_3(f_4(\mathbf{x}))))$

b) $T(\mathbf{x}) = f_4(f_3(f_2(f_1(\mathbf{x}))))$

c) $T(\mathbf{x}) = f_1(f_1(f_3(f_4(\mathbf{x}))))$

d) $T(\mathbf{x}) = f_4(f_2(f_4(f_2(\mathbf{x}))))$

e) $T(\mathbf{x}) = f_3(f_4(f_3(f_4(\mathbf{x}))))$

Pro každé $\mathbf{x} \in \mathbf{R}^2$ ($\mathbf{x}$ považujeme za sloupcový vektor) jsou dána zobrazení do $\mathbf{R}^2$:

$$f_1(\mathbf{x}) = A_1 \cdot \mathbf{x}, \quad f_2(\mathbf{x}) = A_2 \cdot \mathbf{x}, \quad f_3(\mathbf{x}) = \mathbf{x} + \mathbf{z}_1, \quad f_4(\mathbf{x}) = \mathbf{x} + \mathbf{z}_2, \text{ kde}$$

$$A_1 = \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}, \; A_2 = \begin{pmatrix} 0.1 & 0 \\ 0 & 0.1 \end{pmatrix}, \; \mathbf{z}_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \; \mathbf{z}_2 = \begin{pmatrix} 0.5 \\ 1 \end{pmatrix}.$$

Určete, která z uvedených složených zobrazení jsou afinní kontrakce.

15. (5 b.)

a) $n! - 1$
b) $(n-1)!$
c) $n! - (n-1)! - (n-2)! - \ldots - 2! - 1!$
d) $n! - (n-1)!$
e) $(n-1)! - (n-2)!$

Všechny permutace množiny $\{1, 2, 3, \ldots, n\}$ seřadíme v rostoucím lexikografickém uspořádání a vtomto pořadí je očíslujeme celými čísly počínaje nulou.
Pořadové číslo permutace $(n, 1, 2, 3, \ldots, n-1)$ pak bude

Zkouškový Test PAL B Jméno ______________________________ sem+bon+pgm: ___

V následujících otázkách je správně někdy jen jedna, někdy více variant odpovědí.
Správně zodpovězená otázka je taková, která určí správně všechny varianty odpovědí a je hodnocena uvedeným počtem bodů. Pokud některá varianta je určena chybně, otázka je hodnocena 0 body. Výjimku tvoří otázky 1., 2., 3., 14. — jedna chybně určená varianta znamená hodnocení polovičním počtem bodů.

1. (10 b.)

V jaké třídě složitosti je funkce `push` v následujícím programu vzhledem k velikosti pole `p` (budeme ji značit N)? Předpokládejte, že funkce `malloc` (funkce `malloc` naalokuje datovou strukturu velikosti argumentu a vrací na tuto strukturu ukazatel) a `free` (funkce `free` uvolní z paměti naalokovanou datovou strukturu, kterou dostane v argumentu) pracují v konstantním čase vzhledem k velikosti pole `p`. Dále předpokládejte, že velikost proměnné `N` vždy před a po provedení funkce `push` koresponduje s velikostí pole `p`. Proměnná `i` je vždy v intervalu `-1` až `N`.

a) O(1)
b) O(N*log(N))
c) O(N²)
d) Ω(1)
e) Ω(log(N))
f) Ω(N²)
g) Θ(N²)
h) Θ(1)

```
int N = 0;
int i = -1;
int * p;
void push (int key)
{
    if (++i >= N) {
        int m = N*2+1;
        int * t = malloc(m*sizeof(int));
        int j;
        for(j = 0; j<N; j++, t[j] = p[j]);
        if (N != 0) free(p);
        N = m;
        p = t;
    }
    p[i] = key;
}
```

2. (10 b.)

V jaké amortizované třídě složitosti je funkce `push` z předchozího programu vzhledem k velikosti pole `p` (budeme ji značit N)? Předpokládejte, že funkce `malloc` a `free` pracují v konstantním čase vzhledem k velikosti pole `p`.

a) O(1)
b) O(N*log(N))
c) O(N²)
d) Ω(1)
e) Ω(log(N))
f) Ω(N²)
g) Θ(N²)
h) Θ(1)

3. (10 b.)

Matice incidence *I* neorientovaného grafu *G* s alespoň šesti uzly obsahuje v každém řádku právě dvě jedničky. Lze s jistotou tvrdit

a) *I* obsahuje v každém soupci právě dvě jedničky
b) *G* má sudý počet uzlů
c) *G* je určitě souvislý
d) *G* může být nesouvislý
e) všechny uzly *G* mají stupeň 2

4. (5 b.)

Mějme čtvercovou matici A racionálních čísel s rozměry n×n a s hodností n. Rozhodněte, zda platí některá z následujících tvrzení:

a) Pokud je matice A diagonální, je po provedení *LUP dekompozice* na A matice P jednotková.
b) Pro matici A takovou, že k ní existuje inverzní matice, lze *LUP dekompozici* provést v čase $O(n^2)$.
c) Matici A lze rozložit pomocí LUP *dekompozice* na matice L, U a P, kde platí, že PA=LU.
d) Předpokládejme, že bychom reprezentovali matici A jako pole čísel s pohyblivou řádovou čárkou dle IEEE 754. Přesnost nalezení inverzní matice k A pomocí *LUP dekompozice* lze zvýšit permutací některých řádků v A před provedením *LUP dekompozice*.
e) K matici A lze najít inverzní matici v čase $O(n^3)$.

5. (5 b.)

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

Když má daný souvislý graf n uzlů a m hran, potom asymptotická složitost algoritmu DFS (prohledávání do hloubky) je $\Theta(m)$, za předpokladu, že během provádění algoritmu máme přístup v konstantním čase ke každému právě zpracovávanému uzlu a ke každé právě zpracovávané hraně. Jaká bude asymptotická složitost tohoto algoritmu, pokud jediná reprezentace grafu, kterou budeme mít k dispozici, bude matice sousednosti tohoto grafu?

6. (5 b.)

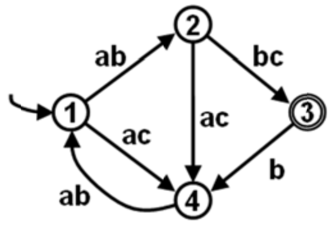
a) $O(n)$
b) $O(\log(n))$
b) $O(n + \log(n))$
c) $O(n \cdot \log(n))$
e) $O(n^2)$

Je dáno n $(n \geq 2)$ navzájem různých celočíselných klíčů a prázdná binární halda.
Všechny klíče vložíme jeden po druhém v náhodném pořadí do dané haldy.
Asymptotická složitost tohoto procesu je

7. (5 b.)

a) 1 | 23 | 3 | 3 | F
b) 34 | 1 | 14 | |
c) 34 | 1 | 14 | | F
d) 14 | 124 | 12 | 14
e) 23 | 4 | 34 | 34 | F

K danému automatu je sestrojen s použitím standardního algoritmu převodu NKA na DKA odpovídající deterministický automat D. Přechodová tabulka D obsahuje řádek



8. (5 b.)

a) má více než 10 stavů,
b) má 4 koncové stavy,
c) nutně obsahuje ε-přechody,
d) v každém stavu, který není koncový, má smyčku (= přechod do téhož stavu),
e) po odstranění smyček z přechodového diagramu A_1 vznikne acyklický graf.

Pro vzorek $P = accac$ nad abecedou $\{a, b, c\}$ je sestrojen nedeterministický automat A_1, pomocí nějž lze v textu vyhledávat řetězce, které mají od P Hammingovu vzdálenost rovnu nejvýše 3. Pro A_1 platí

9. (5 b.)

a) a: 10, b: 40, c: 40, d: 40, e: 40
b) a: 80, b: 10, c: 10, d: 10, e: 10
c) a: 60, b: 10, c: 20, d: 30, e: 40
d a: 90, b: 10, c: 20, d: 30, e: 30
e) a: 50, b: 10, c: 20, d: 30, e: 30

Zpráva nad abecedou $\{a, b, c, d, e\}$ je zakódována pomocí statického Huffmanova kódování.
Znaky b, c, d, e jsou kódovány 3 bity, znak a je kódován jedním bitem.
Možné frekvence jednotlivých znaků tedy jsou:

10. (5 b.)

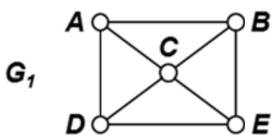
a) $1/2\,(n^2 + n - 2)$
b) $1/2\,(3n^2 - 15n + 28)$
c) $1/2\,(6n - 8)$
d) $1/2\,(n^2 + 1)$
e) $1/2\,(n^2 - n + 4)$

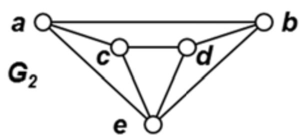
Graf W_n vznikne tak, že každý uzel kružnice C_n spojíme hranou s dalším přidaným uzlem x (W_n má tedy n+1 uzlů). Určete, kolik cest v grafu W_n vede mezi dvěma uzly, které sousedí v C_n, pokud $n \geq 3$.

11. (10 b.)

a) 0
b) 6
c) 8
d) 12
e) 24

Počet izomorfizmů mezi grafy G_1 a G_2 je

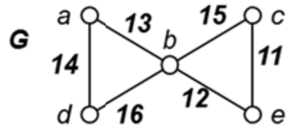




12. (5 b.)

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

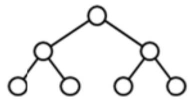
V Kruskalově algoritmu reprezentujeme datovou strukturu Union-Find pomocí orientovaných stromů. Určete, jaká je hloubka konečného stromu v této struktuře po nalezení minimální kostry grafu G (kořen má hloubku 0). Předpokládáme, že při sjednocování stromů zařazujeme vždy menší strom pod kořen většího a nepoužíváme žádné další heuristiky pro zvýšení efektivity.



13. (5 b.)

a) hloubka T bude 2
b) hloubka T bude 3
c) tvar T zůstane beze změny
d) pravý podstrom kořene bude mít jeden uzel
e) pravý podstrom kořene bude prázdný

Splay strom T má tvar daný obrázkem. Po vyhledání prvku s největším klíčem v T bude platit, že



14. (10 b.)

a) $T(\mathbf{x}) = f_1(f_2(f_3(f_4(\mathbf{x}))))$
b) $T(\mathbf{x}) = f_4(f_3(f_2(f_1(\mathbf{x}))))$
c) $T(\mathbf{x}) = f_1(f_1(f_3(f_4(\mathbf{x}))))$
d) $T(\mathbf{x}) = f_4(f_2(f_4(f_2(\mathbf{x}))))$
e) $T(\mathbf{x}) = f_1(f_2(f_3(f_1(\mathbf{x}))))$

Pro každé $\mathbf{x} \in \mathbf{R}^2$ ($\mathbf{x}$ považujeme za sloupcový vektor) jsou dána zobrazení do $\mathbf{R}^2$:

$$f_1(\mathbf{x}) = A_1 \cdot \mathbf{x}, \quad f_2(\mathbf{x}) = A_2 \cdot \mathbf{x}, \quad f_3(\mathbf{x}) = \mathbf{x} + \mathbf{z_1}, \quad f_4(\mathbf{x}) = \mathbf{x} + \mathbf{z_2}, \text{ kde}$$

$$A_1 = \begin{pmatrix} 0.5 & 0 \\ 0 & 0.5 \end{pmatrix}, \quad A_2 = \begin{pmatrix} 3 & 0 \\ 0 & 3 \end{pmatrix}, \quad \mathbf{z_1} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \quad \mathbf{z_2} = \begin{pmatrix} 1.5 \\ 2 \end{pmatrix}.$$

Určete, která z uvedených složených zobrazení jsou afinní kontrakce.

15. (5 b.)

a) $(n-2)! + 1$
b) $(n-1)!$
c) $1! + 3! + 4! + ... + (n-1)! - 2!$
d) $n! - (n-1)!$
e) $(n-2)! + (n-3)!$

Všechny permutace množiny $\{1, 2, 3, ..., n\}$ seřadíme v rostoucím lexikografickém uspořádání a vtomto pořadí je očíslujeme celými čísly počínaje nulou.
Pořadové číslo permutace $(2, 1, 3, 4, 5, 6, ..., n-1, n)$ pak bude