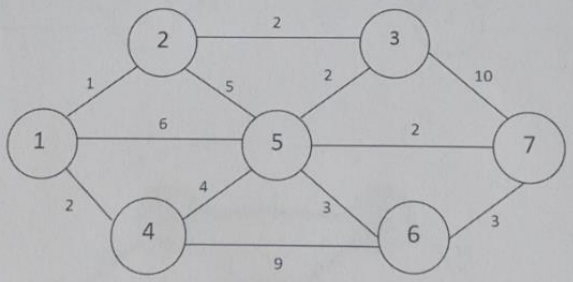
1. Grafové algoritmy **[20 bodů]**

Na následujícím obrázku je znázorněn neorientovaný graf **G**. Určete matici sousednosti **S** a **Prim-Jarnikovým** algoritmem nalezněte "nejlevnější" (minimální) kostru tohoto grafu. V jednotlivých obrázcích zřetelně označte uzly, které se budou přidávat do "mraku" a ohodnocení uzlů, které se v daném kroku změní. V posledním obrázku zřetelně označte nalezenou "nejlevnější" (minimální) kostru.



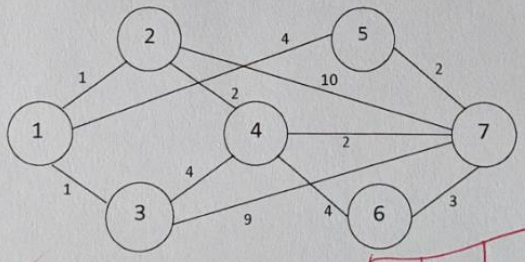
S=

	1	2	3	4	5	6	7
1	0	1	-1	2	6	-1	-1
2	1	0	2	-1	5	-1	-1
3	-1	2	0	-1	2	-1	10
4	2	-1	-1	0	4	9	-1
5	6	5	2	4	0	3	2
6	-1	-1	-1	9	3	0	3
7	-1	-1	10	-1	2	3	0

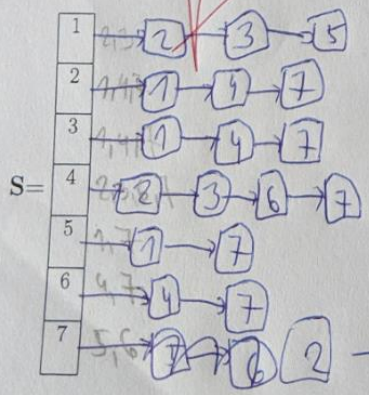
5

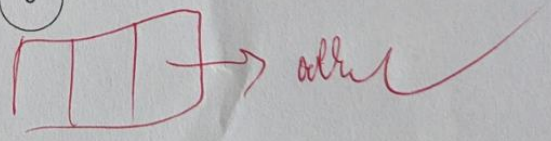
1. Grafové algoritmy **[20 bodů]**

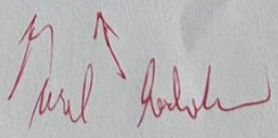
Na následujícím obrázku je znázorněn neorientovaný graf **G**. Určete seznam sousednosti **S** a **Prim-Jarnikovým** algoritmem nalezněte "nejlevnější" (minimální) kostru tohoto grafu. V jednotlivých obrázcích zřetelně označte uzly, které se budou přidávat do "mraku" a ohodnocení uzlů, které se v daném kroku změní. V posledním obrázku zřetelně označte nalezenou "nejlevnější" (minimální) kostru.

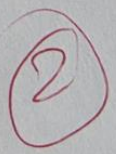


S=

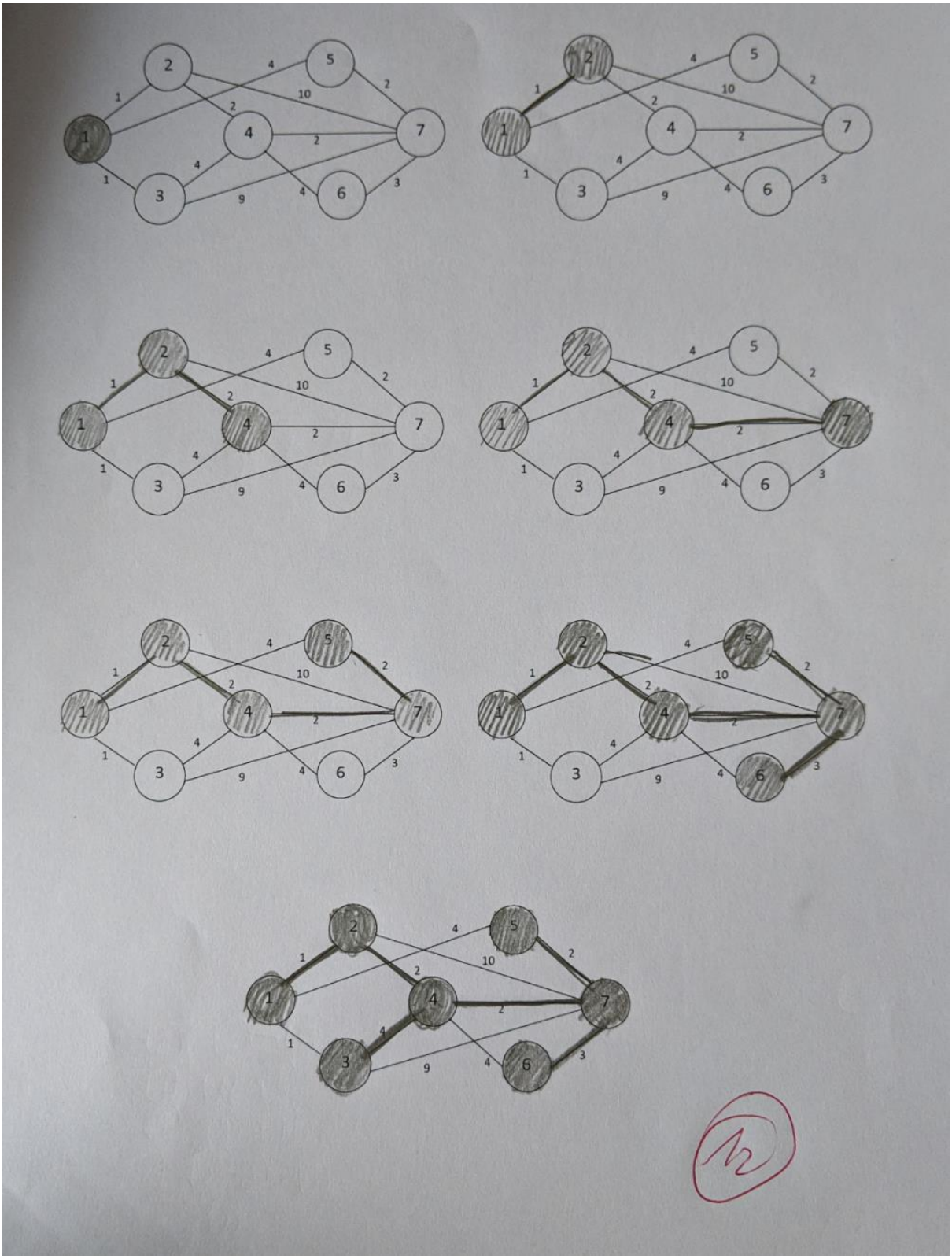








2


1
2
3
4
5
6
7
1
10
2
4
1
2
4
2
4
1
9
4
3

2. Zpracování textů I [20 bodů]

A) Určete, kolik operací porovnávání znaků musí provést BEREME_BERBERY

1. jednoduchý algoritmus přímého vyhledávání (brute-force) [2 body]
2. Boyer - Mooreův algoritmus [5 bodů]
3. Knuth - Moris - Prattův algoritmus [5 bodů]

pro nalezení slova BERBERY ve znakovém řetězci (prohledávaném textu):

BEREME_BERBERY_JAKO_MALÝ_NÁROD

i) 16 porovnání ii) 8 porovnání iii) 14 porovnání

B) Do předtištěného pole doplňte pro uvedené znaky hodnotu funkce **Last(x)** použité u Boayer - Moorova algoritmu [4 body]

P | B | E | R | B | E | R | Y

x		A	B	D	E	J	K	L	M	O	R	Y	Á	Ý
Last(x)		-1	3	-1	4	-1	-1	-1	-1	-1	5	6	-1	-1

C) Do předtištěného pole doplňte hodnoty chybové funkce **F(k)** KMP algoritmu pro hledané slovo BERBERY [4 body]

B E R B E R Y

k	0	1	2	3	4	5	6
F(k)	0	0	0	1	2	3	0

3. Zpracování textů II [12 bodů]

Mějme zadané dva řetězce X="TGAT" a Y="TCAGT". Pro tyto řetězce vyplňte odpovídající tabulky a určete:

a) nejdelší společnou sekvenci znaků (LCS) která se v obou řetězcích vyskytuje, [6 bodů]

X_i \ Y_j	T	C	A	G	T
T	0	0	0	0	0
G	0	1↖	1←	1←	1↖
A	0	1↑	1↑	1↑	2←
T	0	1↖	1↑	2↑	2↑
	0	1↖	1↑	2↑	3↖

nejdelší společná sekvence: TGT

b) vzdálenost mezi řetězci, za předpokladu, že všechny chybové transformace (vložení, náhrada, zrušení) mají váhu **1** [6 bodů]

2. Zpracování textů I

a) Určete, kolik operací porovnávání znaků musí provést

1. jednoduchý algoritmus přímého vyhledávání (brute-force) [2 body]
2. Boyer - Mooreův algoritmus [5 bodů]
3. Knuth - Moris - Prattův algoritmus [5 bodů]

pro nalezení slova BEREME ve znakovém řetězci (prohledávaném textu

BARONY_A_BARBARY_BEREME_JAKO_PROTIKLADY

i) 26 porovnání ii) 10 porovnání iii) 24 26 porovnání

7

b) Do předtištěného pole doplňte pro uvedené znaky hodnotu funkce **Last(x)** použité u Boayer - Moorova algoritmu [4 body]

BEREME
0 1 2 3 4 5

x	-	A	B	D	E	I	J	K	L	M	N	O	P	R	T	Y
Last(x)	-1	-1	0	-1	5	-1	-1	-1	-1	~~4~~	-1	-1	-1	2	-1	-1

4

c) Do předtištěného pole doplňte hodnoty chybové funkce **F(k)** KMP algoritmu pro hledané slovo BEREME [4 body]

B E R E M E

k	0	1	2	3	4	5
F(k)	0	0	0	1	0	1

0 0

4,5

3. Zpracování textů II [12 bodů]

Mějme zadané dva řetězce X="ADCA" a Y="ABCDA". Pro tyto řetězce vyplňte odpovídající tabulky a určete:

a) nejdelší společnou sekvenci znaků (LCS) která se v obou řetězcích vyskytuje, [6 bodů]

X_i \ Y_j	A	B	C	D	A
A	0	1	1	1	1
D	0	1	1	1	2
C	0	1	1	2	2
A	0	1	1	2	3

6

nejdelší společná sekvence: ADA

b) vzdálenost mezi řetězci, za předpokladu, že všechny chybové transformace (vložení, náhrada, zrušení) mají váhu 1 [6 bodů]

X_i \ Y_j	A	B	C	D	A
A	1	0	1	2	3
D	2	1	1	2	3
C	3	2	2	1	2
A	4	3	3	2	2

6

vzdálenost mezi řetězci: 2

4. Stromové struktury

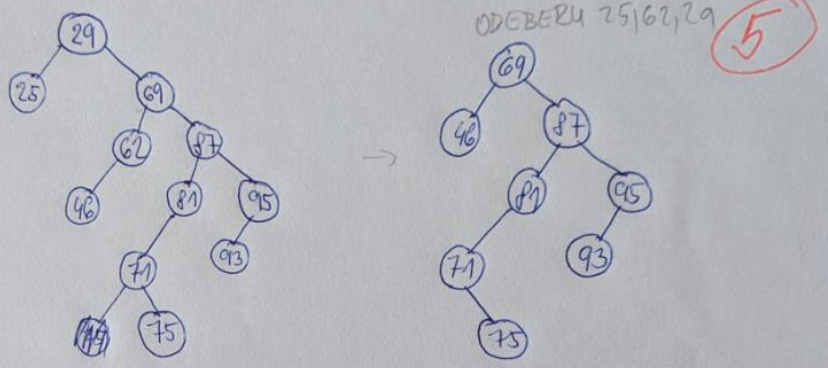
[20 bodů]

Mějte dánu následující posloupnost datových položek identifikovatelných celočíselnými klíči:

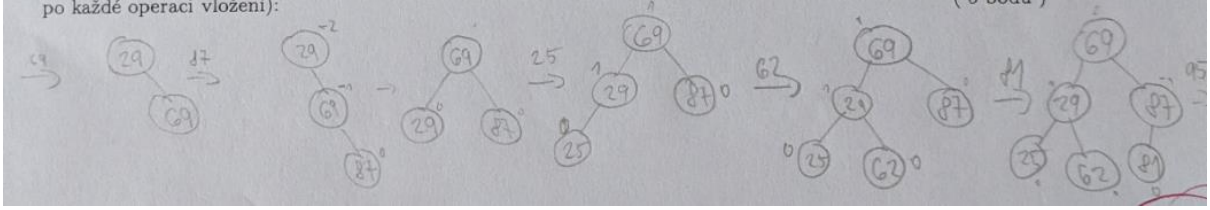
29, 69, 87, 25, 62, 81, 95, 71, 46, 75, 93,

Posloupnost položek reprezentujte zadanými typy stromů (pro jednoduchost znázornění reprezentujte položky jen uvedenými hodnotami klíčů, tam, kde je to nutné, dodržte výše zadané pořadí položek). Po vytvoření stromu odeberte v uvedeném pořadí prvky s klíči **25, 62, 29**. (Při rušení prvku ve vnitřním uzlu BVS a AVL stromu nahrazujte rušený prvek jeho symetrickým předchůdcem - pokud to jde).

a) Reprezentace **BVS** stromem: (5 bodů)



b) Reprezentace **AVL** stromem (prvky vkládejte postupně a pokud je to nutné provádějte vyvážení stromu po každé operaci vložení): (5 bodů)

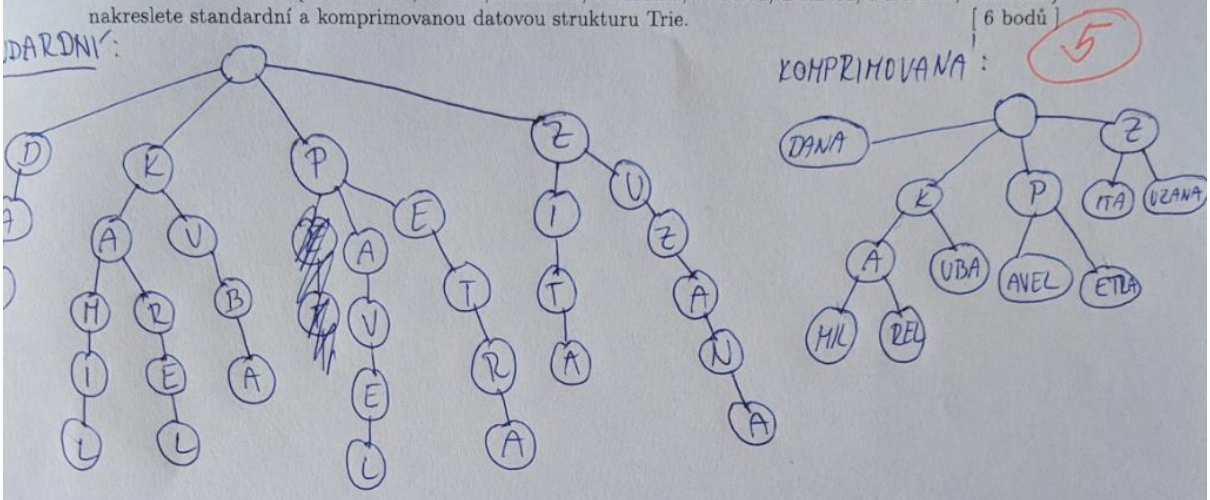


c) Reprezentace **B** stromem řádu m=3 (uzel obsahuje 3 ukazatele): (5 bodů)

d) Reprezentace **B** stromem řádu m=5 (uzel obsahuje 5 ukazatelů): (5 bodů)

5. **Datová struktura Trie** [12 bodů]

- Pro množinu řetězců {**ZUZANA**, **ZITA**, **KAMIL**, **KAREL**, **KUBA**, **DANA**, **PETRA**, **PAVEL**} nakreslete standardní a komprimovanou datovou strukturu Trie. [6 bodů]



- Pro slovo **BARBAROSSA** nakreslete suffixovou datovou strukturu Trie. [6 bodů]

6. **Komprese dat**

a) Pro znakový řetězec [16 bodů]

NENECHAVY_NENECHAVEC_NENECHAL_NIC

kde znak _ značí mezeru, vytvořte Huffmanův kódovací strom; aby Vámi vytvořený strom byl jednoznačný, důsledně dodržujte řazení znaků vkládaných do stromu podle četností jejich výskytu v řetězci a uvnitř tříd znaků se stejnou četností pak dodržujte abecední řazení znaků s tím, že _ (mezeru) řaďte **na začátek abecedy**, jak je dáno kódovou tabulkou ASCII kódu: (10 bodů)

b) Vámi vytvořeným Huffmanovým kódovacím stromem zakódujte slovo HELENA : (2 body)

Zakódovaná posloupnost: . . . . . .

c) Tímtéž stromem dekódujte komprimovaný řetězec (2 body)

10011111011111111

5. Tabulky s rozptýlenými položkami [12 bodů]

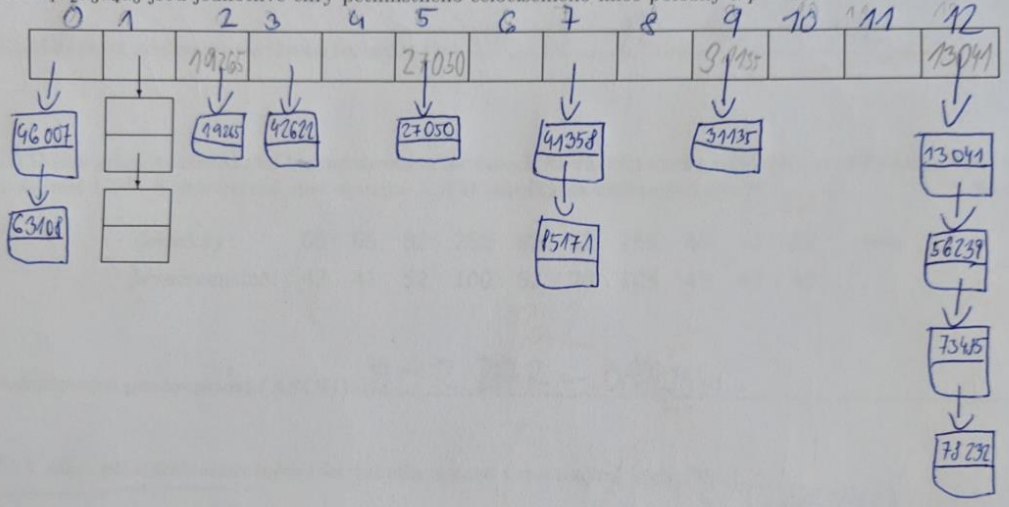
a) Do níže vyobrazené struktury, která implementuje hash-tabulku s vnějším zřetězením s ukládáním synonymických položek do seznamů zřetězených prvků (metoda "scattered index"), zařaďte (zakreslete naznačeným způsobem) položky s níže uvedenými celočíselnými klíči v pořadí: (10 bodů)

13041 19265 27050 31135 41358 42622 46007 56239 63108 73485 78232 85171

Hodnotu rozptylové funkce pro prvotní přístup do jednotlivých seznamů určujte podle vztahu

$$h(k) = (a_1 * a_2 * a_4)\%p$$

kde $a_1a_2a_3a_4a_5$ jsou jednotlivé cifry pětimístného celočíselného klíče položky a $p = 13$.



b) vyčíslete střední algoritmickou složitost vyhledávání položek (v počtu operací porovnání klíčů potřebných pro vyhledání položky) v takto vytvořené implemetaci tabulky:

střední počet operací porovnání klíčů: . . . . . (2 body)

6. **Komprese dat** [16 bodů]

a) Metodou LZW komprese zakódujte (komprimujte) posloupnost znaků (8 bodů)

KURKUMA

Zakódovanou posloupnost zapište hexadecimálně s využitím přiložené ACSII tabulky znaků.
POZOR !!!! Na následující straně je uvedena pouze dolní polovina ASCII tabulky. Předpokládejte, že kompletní tabulka obsahuje kódy znaků od 0 do 255 a nové kódy vytvářené algoritmem komprese začínají dekadickou hodnotou 256.

6

Zakódovaná posloupnost (hexadecimálně): 48 55 52 100 102

b) Dekomprimujte následující komprimovanou informaci, která byla zkomprimována standardním algoritmem komprese LZW. K dekompresi opět využijte ASCII tabulku na následující straně. (8 bodů)

dekadicky:	66	65	82	256	82	32	259	66	73	69	, resp.
hexadecimálně:	42	41	52	100	52	20	103	42	49	45	

4

Dekódovaná posloupnost (ASCII):

!!!!! V obou případech znázorněte část tabulky s nově vytvořenými kódy !!!!!