1 Zásobník s nárazníkom a záznamy

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Vytvorte zásobník s nárazníkom, pomocou ktorého uložíte záznamy do pamäte. Záznamy zapíšte pomocou zásobníka do výstupného súboru *output.txt* v opačnomnom poradí.

Príklad:

input.txt
```
2
Janko Hrasko 15
Ruzenka Sipkova 100
```

output.txt
```
Ruzenka Sipkova
Janko Hrasko
```

2 Zásobník bez nárazníka a záznamy

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Vytvorte zásobník bez nárazníka, pomocou ktorého uložíte záznamy do pamäte. Záznamy zapíšte pomocou zásobníka do výstupného súboru *output.txt* v opačnom poradí.

Príklad:

input.txt
```
2
Janko Hrasko 15
Ruzenka Sipkova 100
```

output.txt
```
Ruzenka Sipkova
Janko Hrasko
```

3 Zásobník jednorozmerným poľom a záznamy

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Implementujte zásobník pomocou jednorozmerného poľa, prostredníctvom neho uložíte záznamy do pamäte. Záznamy zapíšte pomocou zásobníka do výstupného súboru *output.txt* v opačnom poradí.

Príklad:

input.txt
```
2
Janko Hrasko 15
Ruzenka Sipkova 100
```

output.txt
```
Ruzenka Sipkova
Janko Hrasko
```

4 Zásobník s nárazníkom – zátvorky v aritmetickom výraze

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje aritmetické výrazy obsahujúce okrúhle zátvorky. V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet výrazov v súbore. Každý výraz sa nachádza v samostatnom riadku.

Úloha:
Vytvorte zásobník s nárazníkom, pomocou ktorého vyhodnotíte správnosť použitia zátvoriek vo výraze. Do výstupného súboru *output.txt* zapíšte odpoveď *ANO* v prípade, ak sú zátvorky použité správne a *NIE,* ak zátvorky nie sú použité správne.

Príklad:

input.txt	*output.txt*
2	
((a+b)-c)(g)(h-2)	ANO
(()(	NIE

5 Zásobník bez nárazníka – zátvorky v aritmetickom výraze

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje aritmetické výrazy obsahujúce okrúhle zátvorky. V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet výrazov v súbore. Každý výraz sa nachádza v samostatnom riadku.

Úloha:
Vytvorte zásobník bez nárazníka, pomocou ktorého vyhodnotíte správnosť použitia zátvoriek vo výraze. Do výstupného súboru *output.txt* zapíšte odpoveď *ANO* v prípade, ak sú zátvorky použité správne a *NIE*, ak zátvorky nie sú použité správne.

Príklad:

input.txt	*output.txt*
2	
((a+b)-c)(g)(h-2)	ANO
(()(	NIE

6 Zásobník jednorozmerným poľom – zátvorky v aritmetickom výraze

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje aritmetické výrazy obsahujúce okrúhle zátvorky. V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet výrazov v súbore. Každý výraz sa nachádza v samostatnom riadku.

Úloha:
Vytvorte zásobník implementovaný jednorozmerným poľom, pomocou ktorého vyhodnotíte správnosť použitia zátvoriek vo výraze. Do výstupného súboru *output.txt* zapíšte odpoveď *ANO* v prípade, ak sú zátvorky použité správne a *NIE*, ak zátvorky nie sú použité správne.

Príklad:

input.txt	*output.txt*
2	ANO
((a+b)-c)(g)(h-2)	NIE
(()(	

7 Rad bez nárazníka a bez pointera na koniec radu (záznamy)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Vytvorte rad bez nárazníka a bez pointera na koniec radu, pomocou ktorého uložíte záznamy do pamäte. Súbor zatvorte a následne záznamy z radu prepíšte do výstupného súboru *output.txt*.

Príklad:

input.txt
```
2
Janko Hrasko 15
Ruzenka Sipkova 100
```

output.txt
```
Janko Hrasko
Ruzenka Sipkova
```

8 Rad bez nárazníka a s pointerom na koniec radu (záznamy)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Vytvorte rad bez nárazníka a s pointerom na koniec radu, pomocou ktorého uložíte záznamy do pamäte. Súbor zatvorte a následne záznamy z radu prepíšte do výstupného súboru *output.txt*.

Príklad:

input.txt
```
2
Janko Hrasko 15
Ruzenka Sipkova 100
```

output.txt
```
Janko Hrasko
Ruzenka Sipkova
```

9 Rad s nárazníkom a bez pointera na koniec radu (záznamy)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Vytvorte rad s nárazníkom a bez pointera na koniec radu, pomocou ktorého uložíte záznamy do pamäte. Súbor zatvorte a následne záznamy z radu prepíšte do výstupného súboru *output.txt*.

Príklad:

input.txt
```
2
Janko Hrasko 15
Ruzenka Sipkova 100
```

output.txt
```
Janko Hrasko
Ruzenka Sipkova
```

10 Rad s nárazníkom a s pointerom na koniec radu (záznamy)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Vytvorte rad s nárazníkom a s pointerom na koniec radu, pomocou ktorého uložíte záznamy do pamäte. Súbor zatvorte a následne záznamy z radu prepíšte do výstupného súboru *output.txt*.

Príklad:

input.txt
```
2
Janko Hrasko 15
Ruzenka Sipkova 100
```

output.txt
```
Janko Hrasko
Ruzenka Sipkova
```

11 Rad jednorozmerným poľom (záznamy)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Vytvorte rad implementovaný jednorozmerným poľom, pomocou ktorého uložíte záznamy do pamäte. Súbor zatvorte a následne záznamy z radu prepíšte do výstupného súboru *output.txt*.

Príklad:

input.txt
```
2
Janko Hrasko 15
Ruzenka Sipkova 100
```

output.txt
```
Janko Hrasko
Ruzenka Sipkova
```

12 Rad bez nárazníka a bez pointera na koniec radu (generovanie kombinačných čísel)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje kombinačné čísla *C(k,n)* v tvare *k, n*. V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet kombinačných čísel v súbore. Údaje kombinačného čísla sú oddelené čiarkou. Každé kombinačné číslo sa nachádza v samostatnom riadku.

Úloha:
Vytvorte rad bez nárazníka a bez pointera na koniec radu, pomocou ktorého budete generovať čísla Pascalovho trojuholníka a tým vypočítate hodnoty jednotlivých kombinačných čísel. Výsledné hodnoty kombinačných čísel zapíšte do výstupného súboru *output.txt*.

Príklad:

input.txt	*output.txt*
2	10
5,2	7
7,1	

13 Rad bez nárazníka a s pointerom na koniec radu (generovanie kombinačných čísel)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje kombinačné čísla *C(k,n)* v tvare *k, n*. V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet kombinačných čísel v súbore. Údaje kombinačného čísla sú oddelené čiarkou. Každé kombinačné číslo sa nachádza v samostatnom riadku.

Úloha:
Vytvorte rad bez nárazníka a s pointerom na koniec radu, pomocou ktorého budete generovať čísla Pascalovho trojuholníka a tým vypočítate hodnoty jednotlivých kombinačných čísel. Výsledné hodnoty kombinačných čísel zapíšte do výstupného súboru *output.txt*.

Príklad:

input.txt	*output.txt*
2	10
5,2	7
7,1	

14 Rad s nárazníkom a bez pointera na koniec radu (generovanie kombinačných čísel)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje kombinačné čísla *C(k,n)* v tvare *k, n*. V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet kombinačných čísel v súbore. Údaje kombinačného čísla sú oddelené čiarkou. Každé kombinačné číslo sa nachádza v samostatnom riadku.

Úloha:
Vytvorte rad s nárazníkom a bez pointera na koniec radu, pomocou ktorého budete generovať čísla Pascalovho trojuholníka a tým vypočítate hodnoty jednotlivých kombinačných čísel. Výsledné hodnoty kombinačných čísel zapíšte do výstupného súboru *output.txt*.

Príklad:

input.txt	*output.txt*
2	10
5,2	7
7,1	

15 Rad s nárazníkom a s pointerom na koniec radu (generovanie kombinačných čísel)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje kombinačné čísla *C(k,n)* v tvare *k, n*. V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet kombinačných čísel v súbore. Údaje kombinačného čísla sú oddelené čiarkou. Každé kombinačné číslo sa nachádza v samostatnom riadku.

Úloha:
Vytvorte rad s nárazníkom a s pointerom na koniec radu, pomocou ktorého budete generovať čísla Pascalovho trojuholníka a tým vypočítate hodnoty jednotlivých kombinačných čísel. Výsledné hodnoty kombinačných čísel zapíšte do výstupného súboru *output.txt*.

Príklad:

input.txt	*output.txt*
2	10
5,2	7
7,1	

16 Rad jednorozmerným poľom (generovanie kombinačných čísel)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje kombinačné čísla *C(k,n)* v tvare *k, n*. V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet kombinačných čísel v súbore. Údaje kombinačného čísla sú oddelené čiarkou. Každé kombinačné číslo sa nachádza v samostatnom riadku.

Úloha:
Vytvorte rad implementovaný jednorozmerným poľom, pomocou ktorého budete generovať čísla Pascalovho trojuholníka a tým vypočítate hodnoty jednotlivých kombinačných čísel. Výsledné hodnoty kombinačných čísel zapíšte do výstupného súboru *output.txt*.

Príklad:

input.txt	*output.txt*
2	10
5,2	7
7,1	

17 Rad bez nárazníka a bez pointera na koniec radu (LSD)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje reťazce. V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet reťazcov v súbore. Každý reťazec sa nachádza v samostatnom riadku.

Úloha:
Pomocou uvedeného typu radu implementujte triediaci algoritmus LSD. Zoradené reťazce sa vypíšu do súboru *output.txt.*

Príklad:

input.txt
```
5
JAN
ANNA
JOZEF
MARGARETA
```

output.txt
```
ANNA
JAN
JOZEF
MARGARETA
```

18 Rad bez nárazníka a s pointerom na koniec radu (LSD)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje reťazce. V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet reťazcov v súbore. Každý reťazec sa nachádza v samostatnom riadku.

Úloha:
Pomocou uvedeného typu radu implementujte triediaci algoritmus LSD. Zoradené reťazce sa vypíšu do súboru *output.txt.*

Príklad:

input.txt
```
5
JAN
ANNA
JOZEF
MARGARETA
```

output.txt
```
ANNA
JAN
JOZEF
MARGARETA
```

19 Rad s nárazníkom a s pointerom na koniec radu (LSD)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje reťazce. V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet reťazcov v súbore. Každý reťazec sa nachádza v samostatnom riadku.

Úloha:
Pomocou uvedeného typu radu implementujte triediaci algoritmus LSD. Zoradené reťazce sa vypíšu do súboru *output.txt.*

Príklad:

input.txt
```
5
JAN
ANNA
JOZEF
MARGARETA
```

output.txt
```
ANNA
JAN
JOZEF
MARGARETA
```

20 Rad s nárazníkom a bez pointera na koniec radu (LSD)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje reťazce. V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet reťazcov v súbore. Každý reťazec sa nachádza v samostatnom riadku.

Úloha:
Pomocou uvedeného typu radu implementujte triediaci algoritmus LSD. Zoradené reťazce sa vypíšu do súboru *output.txt.*

Príklad:

input.txt	*output.txt*
5	ANNA
JAN	JAN
ANNA	JOZEF
JOZEF	MARGARETA
MARGARETA	

21 Rad jednorozmerným poľom (LSD)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje reťazce. V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet reťazcov v súbore. Každý reťazec sa nachádza v samostatnom riadku.

Úloha:
Pomocou uvedeného typu radu implementujte triediaci algoritmus LSD. Zoradené reťazce sa vypíšu do súboru *output.txt.*

Príklad:

input.txt
```
5
JAN
ANNA
JOZEF
MARGARETA
```

output.txt
```
ANNA
JAN
JOZEF
MARGARETA
```

22 Rad bez nárazníka a bez pointera na koniec radu (MSD)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje reťazce. V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet reťazcov v súbore. Každý reťazec sa nachádza v samostatnom riadku.

Úloha:
Pomocou uvedeného typu radu implementujte triediaci algoritmus MSD. Zoradené reťazce sa vypíšu do súboru *output.txt.*

Príklad:

input.txt
```
5
JAN
ANNA
JOZEF
MARGARETA
```

output.txt
```
ANNA
JAN
JOZEF
MARGARETA
```

23 Rad bez nárazníka a s pointerom na koniec radu (MSD)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje reťazce. V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet reťazcov v súbore. Každý reťazec sa nachádza v samostatnom riadku.

Úloha:
Pomocou uvedeného typu radu implementujte triediaci algoritmus MSD. Zoradené reťazce sa vypíšu do súboru *output.txt.*

Príklad:

input.txt
```
5
JAN
ANNA
JOZEF
MARGARETA
```

output.txt
```
ANNA
JAN
JOZEF
MARGARETA
```

24 Rad s nárazníkom a s pointerom na koniec radu (MSD)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje reťazce. V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet reťazcov v súbore. Každý reťazec sa nachádza v samostatnom riadku.

Úloha:
Pomocou uvedeného typu radu implementujte triediaci algoritmus MSD. Zoradené reťazce sa vypíšu do súboru *output.txt.*

Príklad:

input.txt
```
5
JAN
ANNA
JOZEF
MARGARETA
```

output.txt
```
ANNA
JAN
JOZEF
MARGARETA
```

25 Rad s nárazníkom a bez pointera na koniec radu (MSD)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje reťazce. V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet reťazcov v súbore. Každý reťazec sa nachádza v samostatnom riadku.

Úloha:
Pomocou uvedeného typu radu implementujte triediaci algoritmus MSD. Zoradené reťazce sa vypíšu do súboru *output.txt.*

Príklad:

input.txt
```
5
JAN
ANNA
JOZEF
MARGARETA
```

output.txt
```
ANNA
JAN
JOZEF
MARGARETA
```

26 Rad jednorozmerným poľom (MSD)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje reťazce. V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet reťazcov v súbore. Každý reťazec sa nachádza v samostatnom riadku.

Úloha:
Pomocou uvedeného typu radu implementujte triediaci algoritmus MSD. Zoradené reťazce sa vypíšu do súboru *output.txt.*

Príklad:

input.txt
```
5
JAN
ANNA
JOZEF
MARGARETA
```

output.txt
```
ANNA
JAN
JOZEF
MARGARETA
```

27 Lineárny jednosmerný spájaný zoznam bez nárazníkov (záznamy)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Vytvorte lineárny jednosmerný spájaný zoznam bez nárazníkov, pomocou ktorého uložíte záznamy do pamäte. Funkčnosť zoznamu demonštrujte tak, že záznamy budú v zozname uložené podľa atribútu priezvisko vzostupne podľa abecedy. Záznamy vypíšte do súboru *output.txt*.

Príklad:

input.txt
```
3
Duro Trulo 22
Janko Hrasko 15
Ruzenka Sipkova 100
```

output.txt
```
Janko Hrasko 22
Ruzenka Sipkova 100
Duro Trulo 15
```

28 Lineárny jednosmerný spájaný zoznam s nárazníkmi (záznamy)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Vytvorte lineárny jednosmerný spájaný zoznam s nárazníkmi, pomocou ktorého uložíte záznamy do pamäte. Funkčnosť zoznamu demonštrujte tak, že záznamy budú v zozname uložené podľa atribútu priezvisko vzostupne podľa abecedy. Záznamy vypíšte do súboru *output.txt*.

Príklad:

input.txt
```
3
Duro Trulo 22
Janko Hrasko 15
Ruzenka Sipkova 100
```

output.txt
```
Janko Hrasko 22
Ruzenka Sipkova 100
Duro Trulo 15
```

29 Cyklický jednosmerný spájaný zoznam (záznamy)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Vytvorte cyklický jednosmerný spájaný zoznam, pomocou ktorého uložíte záznamy do pamäte. Funkčnosť zoznamu demonštrujte tak, že záznamy budú v zozname uložené podľa atribútu priezvisko vzostupne podľa abecedy. Záznamy vypíšte do súboru *output.txt*.

Príklad:

input.txt
```
3
Duro Trulo 22
Janko Hrasko 15
Ruzenka Sipkova 100
```

output.txt
```
Janko Hrasko 22
Ruzenka Sipkova 100
Duro Trulo 15
```

30 Lineárny obojsmerný spájaný zoznam bez nárazníkov (záznamy)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Vytvorte lineárny obojsmerný spájaný zoznam bez nárazníkov, pomocou ktorého uložíte záznamy do pamäte. Funkčnosť zoznamu demonštrujte tak, že záznamy budú v zozname uložené podľa atribútu priezvisko vzostupne podľa abecedy. Záznamy vypíšte do súboru *output.txt*.

Príklad:

input.txt
```
3
Duro Trulo 22
Janko Hrasko 15
Ruzenka Sipkova 100
```

output.txt
```
Janko Hrasko 22
Ruzenka Sipkova 100
Duro Trulo 15
```

31 Lineárny obojsmerný spájaný zoznam s nárazníkmi (záznamy)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Vytvorte lineárny obojsmerný spájaný zoznam s nárazníkmi, pomocou ktorého uložíte záznamy do pamäte. Funkčnosť zoznamu demonštrujte tak, že záznamy budú v zozname uložené podľa atribútu priezvisko vzostupne podľa abecedy. Záznamy vypíšte do súboru *output.txt*.

Príklad:

input.txt
```
3
Duro Trulo 22
Janko Hrasko 15
Ruzenka Sipkova 100
```

output.txt
```
Janko Hrasko 22
Ruzenka Sipkova 100
Duro Trulo 15
```

32 Cyklický jednosmerný spájaný zoznam (záznamy)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Vytvorte cyklický obojsmerný spájaný zoznam, pomocou ktorého uložíte záznamy do pamäte. Funkčnosť zoznamu demonštrujte tak, že záznamy budú v zozname uložené podľa atribútu priezvisko vzostupne podľa abecedy. Záznamy vypíšte do súboru *output.txt*.

Príklad:

input.txt
```
3
Duro Trulo 22
Janko Hrasko 15
Ruzenka Sipkova 100
```

output.txt
```
Janko Hrasko 22
Ruzenka Sipkova 100
Duro Trulo 15
```

33 Binárny strom (preklad Morzeovho kódu)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje reťazce morzeovho kódu. V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet reťazcov v súbore. Každý reťazec sa nachádza v samostatnom riadku.

Úloha:
Vytvorte binárny strom pomocou rekurzívneho údajového typu, pomocou ktorého dokážete prekladať reťazce morzeovho kódu do latinky. Preklady reťazcov vypíšte do súboru *output.txt*.

Príklad:

input.txt	*output.txt*
4	SOS
… --- …	INFORMATIKA
.. .- ..-. --- .- .-- .- - .. -.- .-	znak neexistuje
-----....----	obsahuje nepripustny znak
--* -.. .. - --	

A .-, B -..., C -.-., D -.., E . , F ..-., G --., H ...., I .., J .---, K -.-, L .-.., M --, N -., O ---, P .--., Q --.-, R .-., S …, T -, U ..-, V ...-, W .--, X -..-, Y -.--, Z --..

34 Binárny strom jednorozmerným poľom (preklad Morzeovho kódu)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje reťazce morzeovho kódu. V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet reťazcov v súbore. Každý reťazec sa nachádza v samostatnom riadku.

Úloha:
Vytvorte binárny strom implementovaný jednrozmerným poľom, pomocou ktorého dokážete prekladať reťazce morzeovho kódu do latinky. Preklady reťazcov vypíšte do súboru *output.txt*.

Príklad:

input.txt	*output.txt*
4	SOS
… --- …	INFORMATIKA
.. .- ..-. --- .- .-- .- - .. -.- .-	znak neexistuje
-----....----	obsahuje nepripustny znak
--* -.. .. - --	

A .-, B -..., C -.-., D -.., E . , F ..-., G --., H ...., I .., J .---, K -.-, L .-.., M --, N -., O ---, P .--., Q --.-, R .-., S …, T -, U ..-, V ...-, W .--, X -..-, Y -.--, Z --..

35 Binárny triediaci strom (záznamy)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Vytvorte binárny triediaci strom pomocou rekurzívneho údajového typu, pomocou ktorého uložíte záznamy do pamäte. Funkčnosť stromu demonštrujte tak, že budete v strome vyhľadávať záznamy podľa atribútu priezvisko. Hľadané priezviská sa nachádzajú v súbore *priezviska.txt*. Stav vyhľadania vypíšte do súboru *output.txt*.

Príklad:

input.txt
```
3
Duro Trulo 22
Janko Hrasko 15
Ruzenka Sipkova 100
```

priezviska.txt
```
Jaga
Sulejman
Trulo
```

output.txt
```
Zaznam s priezviskom Jaga sa v strome nenachadza
Zaznam s priezviskom Sulejman sa v strome nenachadza
Duro Trulo 15
```

36 Binárny triediaci strom jednorozmerným poľom (záznamy)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Vytvorte binárny triediaci strom implementovaný jednorozmerným poľom, pomocou ktorého uložíte záznamy do pamäte. Funkčnosť stromu demonštrujte tak, že budete v strome vyhľadávať záznamy podľa atribútu priezvisko. Hľadané priezviská sa nachádzajú v súbore *priezviska.txt*. Stav vyhľadania vypíšte do súboru *output.txt*.

Príklad:

input.txt
```
3
Duro Trulo 22
Janko Hrasko 15
Ruzenka Sipkova 100
```

priezviska.txt
```
Jaga
Sulejman
Trulo
```

output.txt
```
Zaznam s priezviskom Jaga sa v strome nenachadza
Zaznam s priezviskom Sulejman sa v strome nenachadza
Duro Trulo 15
```

37 Lexikografický strom (preklad Angličtina –> Slovenčina)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje dvojice slov. Prvé slovo je anglické, druhé je jeho alternatíva v Slovenčine. Dvojice slov sú oddelené medzerou. Každá dvojica sa nachádza v samostatnom riadku.

Úloha:
Vytvorte lexikografický strom strom pomocou rekurzívneho údajového typu, pomocou ktorého dokážete prekladať anglické výrazy do Slovenčiny. Funkčnosť stromu demonštrujte tak, že budete v strome vyhľadávať preklady anglických slov. Prekladané slová sa nachádzajú v súbore *prelozit.txt.* Stav prekladu vypíšte do súboru *output.txt*. Preklady sa ukladajú v uzloch. V programe nepredpokladáme výskyt viacvýznamových slov.

Príklad:

input.txt
dog pes
door dvere
keep drzat

output.txt
dog -> pes
keep ->drzat
do -> nie je v slovniku

prelozit.txt
dog
keep
do

38 Lexikografický strom jednorozmerným poľom (preklad Angličtina –> Slovenčina)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje dvojice slov. Prvé slovo je anglické, druhé je jeho alternatíva v Slovenčine. Dvojice slov sú oddelené medzerou. Každá dvojica sa nachádza v samostatnom riadku.

Úloha:
Vytvorte lexikografický strom implementovaný jednorozmerným poľom, pomocou ktorého dokážete prekladať anglické výrazy do Slovenčiny. Funkčnosť stromu demonštrujte tak, že budete v strome vyhľadávať preklady anglických slov. Prekladané slová sa nachádzajú v súbore *prelozit.txt.* Stav prekladu vypíšte do súboru *output.txt*. Preklady sa ukladajú v uzloch. V programe nepredpokladáme výskyt viacvýznamových slov.

Príklad:

input.txt
dog pes
door dvere
keep drzat

output.txt
dog -> pes
keep ->drzat
do -> nie je v slovniku

prelozit.txt
dog
keep
do

39 Lexikografický strom pre jazyk s určenou množinou symbolov jazyka.

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje dvojice slov. Prvé slovo je v jazyku mimozemšťanov, ktorý používa len podmnožinu symbolov latinky, druhé je jeho alternatíva v Slovenčine. Dvojice slov sú oddelené medzerou. Každá dvojica sa nachádza v samostatnom riadku.

Úloha:
Vytvorte lexikografický strom implementovaný rekurzívnym údajovým typom, pomocou ktorého dokážete prekladať výrazy mimozemšťanov do Slovenčiny. Funkčnosť stromu demonštrujte tak, že budete v strome vyhľadávať preklady slov mimozemšťanov. Prekladané slová sa nachádzajú v súbore *prelozit.txt.* Stav prekladu vypíšte do súboru *output.txt*. Nepredpokladá sa výskyt viacerých významov jedného slova.

Príklad:

input.txt
xxy pes
axg% dvere
@P8 drzat

output.txt
xxy -> pes
axg% ->dvere
##! -> nie je v slovniku

prelozit.txt
xxy
axg%
##!

40 Lexikografický strom pre jazyk s určenou množinou symbolov jazyka – jednorozmerným poľom.

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje dvojice slov. Prvé slovo je v jazyku mimozemšťanov, ktorý používa len podmnožinu symbolov latinky, druhé je jeho alternatíva v Slovenčine. Dvojice slov sú oddelené medzerou. Každá dvojica sa nachádza v samostatnom riadku.

Úloha:
Vytvorte lexikografický strom implementovaný jednorozmerným poľom, pomocou ktorého dokážete prekladať výrazy mimozemšťanov do Slovenčiny. Funkčnosť stromu demonštrujte tak, že budete v strome vyhľadávať preklady slov mimozemšťanov. Prekladané slová sa nachádzajú v súbore *prelozit.txt.* Stav prekladu vypíšte do súboru *output.txt*. Nepredpokladá sa výskyt viacerých významov jedného slova.

Príklad:

input.txt
xxy pes
axg% dvere
@P8 drzat

output.txt
xxy -> pes
axg% ->dvere
##! -> nie je v slovniku

prelozit.txt
xxy
axg%
##!

41 Dynamické programovanie – výpočet člena rekurentnej postupnosti.

Vyučujúci zadá konkrétnu postupnosť.

Úloha:
Napísať program, pomocou ktorého sa nájde *n*-tý člen rekurentnej postupnosti. Porovná sa čas výpočtu, ak sa úloha rieši nerekurzívne, rekurzívne a s použitím dynamického programovania.

42 Backtracking – Robot v bludisku

Na základe parametrov programu (argv) sa vygeneruje bludisko veľkosti argv[1] s argv[2] s náhodne rozmiestnenými prekážkami, pričom platí, že argv[1]>>argv[2]. Default nastavenie pre rozmer bludiska je 8 × 8 a pre počet prekážok je 20. V ľavom hornom rohu je umiestnený robot, ktorý má nájsť cestu von z bludiska. Miesto východu sa generuje náhodne.

Úloha:
Vytvorte program s použitím algoritmu vyhľadávania so spätným návratom. Priebeh hľadania cesty sa ukladá do súboru *output.txt*.

43 Backtracking – 8 dám na šachovnici

Dáma položená na šachové políčko ohrozuje všetky políčka vo vodorovnom smere, zvislom smere, diagonálnych smeroch (8 smerov). Máme šachovnicu veľkosti 8 × 8 a 8 dám. Koľkými spôsobmi sa dajú tieto dámy rozložiť na šachovnici tak, aby sa navzájom neohrozovali.

Úloha:
Vytvorte program s použitím algoritmu vyhľadávania so spätným návratom. Nájdené riešenia program ukladá do súboru *output.txt*.

44 Backtracking – Jazdcová prechádzka

Kôň položený na šachové políčko môže skákať8 smermi. Máme šachovnicu veľkosti 8 × 8. Je možné a koľkými spôsobmi, skákať po šachovnici koňom tak, aby na každé políčko skočil práve raz a zložitejšie zadanie – skončil tam kde začal.

Úloha:
Vytvorte program s použitím algoritmu vyhľadávania so spätným návratom na hľadanie odpovede. Nájdené riešenia program ukladá do súboru *output.txt*.

45 Úlohy na grafoch – Značkovací algoritmus (matica susednosti)

Na vstupe je súbor *input.txt*, v ktorom je zadaný graf (orientovaný alebo neorientovaný) vo forme matice susednosti.

Úloha:
Vytvorte program s použitím značkovacieho algoritmu, ktorý zistí všetky dostupné uzly zo zvoleného uzla *r*, ktorý sa zadá ako parameter programu. Dostupné uzly zapíše do súboru *output.txt*.

Príklad:
input.txt
5
0 1 0 0 0
0 0 0 0 1
0 0 0 1 0
0 1 0 0 0
1 0 1 0 0

output.txt (pre r = 3)
3 1 4 2 0

46 Úlohy na grafoch – Značkovací algoritmus (matica incidencie)

Na vstupe je súbor *input.txt*, v ktorom je zadaný orientovaný graf vo forme matice incidencie.

Úloha:
Vytvorte program s použitím značkovacieho algoritmu, ktorý zistí všetky dostupné uzly zo zvoleného uzla *r*, ktorý sa zadá ako parameter programu. Dostupné uzly zapíše do súboru *output.txt*.

Príklad:
input.txt
5
–1 1 0 0 0 0
1 0 1 –1 0 0
0 0 0 0 1 –1
0 0 –1 0 0 1
0 –1 0 1 –1 0

*output.txt (*pre *r = 3)*
3 1 4 2 0

47 Úlohy na grafoch – Algoritmus prehľadávania do hĺbky (matica susednosti)

Na vstupe je súbor *input.txt*, v ktorom je zadaný orientovaný graf vo forme matice susednosti.

Úloha:
Vytvorte program s použitím algoritmu prehľadávania do hĺbky, ktorý zistí všetky dostupné uzly zo zvoleného uzla *r*, ktorý sa zadá ako parameter programu. Dostupné uzly zapíše do súboru *output.txt*.

Príklad:
input.txt
5
0 1 0 0 0
0 0 0 0 1
0 0 0 1 0
0 1 0 0 0
1 0 1 0 0

*output.txt (*pre *r = 3)*
3 1 4 2 0

48 Úlohy na grafoch – Algoritmus prehľadávania do hĺbky (matica incidencie)

Na vstupe je súbor *input.txt*, v ktorom je zadaný orientovaný graf vo forme matice incidencie.

Úloha:
Vytvorte program s použitím algoritmu prehľadávania do hĺbky, ktorý zistí všetky dostupné uzly zo zvoleného uzla *r*, ktorý sa zadá ako parameter programu. Dostupné uzly zapíše do súboru *output.txt*.

Príklad:
input.txt
5
–1 1 0 0 0 0
1 0 1 –1 0 0
0 0 0 0 1 –1
0 0 –1 0 0 1
0 –1 0 1 –1 0

output.txt (pre r = 3)
3 1 4 2 0

49 Úlohy na grafoch – Algoritmus prehľadávania do šírky (matica susednosti)

Na vstupe je súbor *input.txt*, v ktorom je zadaný orientovaný graf vo forme matice susednosti. V prvom riadku súboru sa nachádza hodnota určujúca počet uzlov v grafe.

Úloha:
Vytvorte program s použitím algoritmu prehľadávania do šírky, ktorý zistí všetky dostupné uzly zo zvoleného uzla *r*, ktorý sa zadá ako parameter programu. Dostupné uzly a ich vzdialenosť od uzla *r* zapíše do súboru *output.txt*.

Príklad:
input.txt
5
0 1 0 0 0
0 0 0 0 1
0 0 0 1 0
0 1 0 0 0
1 0 1 0 0

output.txt (pre r = 3)
3 1 4 2 0
0 1 2 3 3

50 Úlohy na grafoch – Algoritmus prehľadávania do šírky (matica incidencie)

Na vstupe je súbor *input.txt*, v ktorom je zadaný orientovaný graf vo forme matice incidencie. V prvom riadku súboru sa nachádza hodnota určujúca počet uzlov v grafe.

Úloha:
Vytvorte program s použitím algoritmu prehľadávania do šírky, ktorý zistí všetky dostupné uzly zo zvoleného uzla *r.* Dostupné uzly a ich vzdialenosť od uzla *r* zapíše do súboru *output.txt*.

Príklad:
input.txt
5
–1 1 0 0 0 0
1 0 1 –1 0 0
0 0 0 0 1 –1
0 0 –1 0 0 1
0 –1 0 1 –1 0

output.txt (pre r = 3)
3 1 4 2 0
0 1 2 3 3

51 Triediaci algoritmus QuickSort (pole záznamov)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Uložte záznamy do poľa alebo spájaného zoznamu a potom toto pole zoraďte algoritmom QuickSort podľa kľúča *priezvisko* vzostupne.

Príklad:

input.txt
```
2
Janko Hrasko 15
Ruzenka Sipkova 100
```

output.txt
```
Janko Hrasko 15
Ruzenka Sipkova 100
```

52 Triediaci algoritmus MergeSort (pole záznamov)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Uložte záznamy do poľa alebo spájaného zoznamu a potom toto pole zoraďte algoritmom MergeSort podľa kľúča *priezvisko* vzostupne. V zdrojovom kóde uveďte princíp činnosti algoritmu.

Príklad:

input.txt
```
2
Janko Hrasko 15
Ruzenka Sipkova 100
```

output.txt
```
Janko Hrasko 15
Ruzenka Sipkova 100
```

53 Triediaci algoritmus HeapSort (pole záznamov)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Uložte záznamy do poľa alebo spájaného zoznamu a potom toto pole zoraďte algoritmom HeapSort podľa kľúča *priezvisko* vzostupne. V zdrojovom kóde uveďte princíp činnosti algoritmu.

Príklad:

input.txt
```
2
Janko Hrasko 15
Ruzenka Sipkova 100
```

output.txt
```
Janko Hrasko 15
Ruzenka Sipkova 100
```

54 Vyhľadávacie algoritmy – Binárne vyhľadávanie v utriedenom poli (pole záznamov)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Ukladajte záznamy do poľa usporiadané algoritmom zakladania a potom použite binárne vyhľadávanie podľa kľúča *priezvisko*. Vyhľadávanie riešte rekurzívne aj nerekurzívne. Hľadané priezviská sa zadáva z klávesnice.

Príklad:

input.txt
```
3
Duro Trulo 22
Janko Hrasko 15
Ruzenka Sipkova 100
```

priezviska.txt
```
Jaga
Sulejman
Trulo
```

output.txt
```
Zaznam s priezviskom Jaga sa v poli nenachadza
Zaznam s priezviskom Sulejman sa v poli nenachadza
Duro Trulo 15
```

55 Vyhľadávacie algoritmy – Binárny vyhľadávací strom (pole záznamov) – jednorozmerným poľom

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Ukladajte záznamy do binárneho vyhľadávacieho stromu implementovaného jednorozmerným poľom a použite vyhľadávanie podľa kľúča *priezvisko*. Hľadané priezviská sa zadávajú zo súboru *priezviska.txt*.

Príklad:

input.txt
```
3
Duro Trulo 22
Janko Hrasko 15
Ruzenka Sipkova 100
```

priezviska.txt
```
Jaga
Sulejman
Trulo
```

output.txt
```
Zaznam s priezviskom Jaga sa v poli nenachadza
Zaznam s priezviskom Sulejman sa v poli nenachadza
Duro Trulo 15
```

56 Vyhľadávacie algoritmy – Binárny vyhľadávací strom (pole záznamov)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Ukladajte záznamy do binárneho vyhľadávacieho stromu implementovaného rekurzívnym údajovým typom a použite vyhľadávanie podľa kľúča *priezvisko*. Hľadané priezviská sa zadávajú zo súboru *priezviska.txt*.

Príklad:

input.txt
3
Duro Trulo 22
Janko Hrasko 15
Ruzenka Sipkova 100

priezviska.txt
Jaga
Sulejman
Trulo

output.txt
Zaznam s priezviskom Jaga sa v poli nenachadza
Zaznam s priezviskom Sulejman sa v poli nenachadza
Duro Trulo 15

57 Vyhľadávacie algoritmy – Hašovanie (pole záznamov) – kľúč typu int

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek, výška v metroch). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Vytvorte hašovaciu tabuľku, do ktorej vložíte záznamy. Hašujte kľúč *vek*. Hašujte štyrmi spôsobmi (prevodom na FP delením, prevodom na FP bitovými operáciami, modulárne, zlatým rezom). Výstupné hašovacie tabuľky sa vypíšu do súboru *output.txt*.

Príklad:

input.txt
```
2
Janko Hrasko 15 1.25
Ruzenka Sipkova 100 1.75
```

output.txt
```
0 Ruzenka Sipkova 100 1.75
1
2
3 Janko Hrasko 15 1.25
```

58 Vyhľadávacie algoritmy – Hašovanie (pole záznamov) – kľúč typu FP

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek, výška v metroch). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Vytvorte hašovaciu tabuľku, do ktorej vložíte záznamy. Hašujte kľúč *výška*. Hašujte dvomi spôsobmi (bez a s použitím zlatého rezu). Výstupné hašovacie tabuľky sa vypíšu do súboru *output.txt*.
Zlatý rez $\varphi = 0{,}618033988$.

Príklad:

input.txt
```
2
Janko Hrasko 15 1.25
Ruzenka Sipkova 100 1.75
```

output.txt
```
0 Ruzenka Sipkova 100 1.75
1
2
3 Janko Hrasko 15 1.25
```

59 Vyhľadávacie algoritmy – Hašovanie (pole záznamov) – kľúč typu reťazec – reťazenie bitov

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek, výška v metroch). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Vytvorte hašovaciu tabuľku, do ktorej vložíte záznamy. Hašujte kľúč *priezvisko* použitím reťazenia bitov. Výstupnú hašovaciu tabuľku vypíšte do súboru *output.txt*. Nepredpokladáme výskyt kolízií. V reťazcoch sa predpokladá nesúvislá oblasť použitých ASCII znakov!!! V prípade, že to nezakomponujete do programu a budete pracovať len s veľkými písmenami, znižuje sa hodnotenie o jeden stupeň.

Príklad:

input.txt
```
2
Janko Hrasko 15 1.25
Ruzenka Sipkova 100 1.75
```

output.txt
```
0 Ruzenka Sipkova 100 1.75
1
2
3 Janko Hrasko 15 1.25
```

60 Vyhľadávacie algoritmy – Hašovanie (pole záznamov) – kľúč typu reťazec – funkcia mod 2 (XOR)

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek, výška v metroch). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Vytvorte hašovaciu tabuľku, do ktorej vložíte záznamy. Hašujte kľúč *priezvisko* použitím operácie xor. Výstupnú hašovaciu tabuľku vypíšte do súboru *output.txt*. Nepredpokladáme výskyt kolízií. V reťazcoch sa predpokladá nesúvislá oblasť použitých ASCII znakov!!! V prípade, že to nezakomponujete do programu a budete pracovať len s veľkými písmenami, znižuje sa hodnotenie o jeden stupeň.

Príklad:

input.txt
```
2
Janko Hrasko 15 1.25
Ruzenka Sipkova 100 1.75
```

output.txt
```
0 Ruzenka Sipkova 100 1.75
1
2
3 Janko Hrasko 15 1.25
```

61 Vyhľadávacie algoritmy – Hašovanie (pole záznamov) – kľúč typu reťazec – posun bitov

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje záznamy o osobách (meno, priezvisko, vek, výška v metroch). V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet záznamov v súbore. Údaje v zázname sú oddelené medzerou. Každý záznam sa nachádza v samostatnom riadku.

Úloha:
Vytvorte hašovaciu tabuľku, do ktorej vložíte záznamy. Hašujte kľúč *priezvisko* použitím operácie xor a posunom bitov. Výstupnú hašovaciu tabuľku vypíšte do súboru *output.txt*. Nepredpokladáme výskyt kolízií. V reťazcoch sa predpokladá nesúvislá oblasť použitých ASCII znakov!!! V prípade, že to nezakomponujete do programu a budete pracovať len s veľkými písmenami, znižuje sa hodnotenie o jeden stupeň.

Príklad:

input.txt
```
2
Janko Hrasko 15 1.25
Ruzenka Sipkova 100 1.75
```

output.txt
```
0 Ruzenka Sipkova 100 1.75
1
2
3 Janko Hrasko 15 1.25
```

62 Hašovanie – riešenie kolízií – oddelené reťazenie

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje FP čísla. V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet FP čísel v súbore. Každé FP číslo sa nachádza v samostatnom riadku.

Úloha:
Vytvorte hašovaciu tabuľku, do ktorej FP čísla. Na hašovanie použite ľubovoľnú metódu. Kolízie riešte použitím oddeleného zreťazenia. Do výstupného súboru *output.txt* sa vypíšu do riadku čísla, ktoré majú rovnaké haše.

Príklad:

input.txt	*output.txt*
5	0 1.0
1.0	1 3.1 74.1
3.1	2
74.1	3
14.5	4
13.5	5 14.5 13.5

63 Hašovanie – riešenie kolízií – lineárna sondáž

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje FP čísla. V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet FP čísel v súbore. Každé FP číslo sa nachádza v samostatnom riadku.

Úloha:
Vytvorte hašovaciu tabuľku, do ktorej FP čísla. Na hašovanie použite ľubovoľnú metódu. Kolízie riešte použitím lineárnej sondáže. Do výstupného súboru *output.txt* sa vypíšu do riadku čísla, ktoré majú rovnaké haše.

Príklad:

input.txt	*output.txt*
5	0 1.0
1.0	1 3.1 74.1
3.1	2
74.1	3
14.5	4
13.5	5 14.5 13.5

64 Hašovanie (pole záznamov) – riešenie kolízií – dvojité hašovanie

Na vstupe je daný textový súbor *input.txt*, ktorý obsahuje FP čísla. V prvom riadku súboru je zapísané celé číslo, ktoré udáva počet FP čísel v súbore. Každé FP číslo sa nachádza v samostatnom riadku.

Úloha:
Vytvorte hašovaciu tabuľku, do ktorej FP čísla. Na hašovanie použite ľubovoľnú metódu. Kolízie riešte použitím dvojitého hašovania. Do výstupného súboru *output.txt* sa vypíšu do riadku čísla, ktoré majú rovnaké haše.

Príklad:

input.txt	*output.txt*
5	0 1.0
1.0	1 3.1 74.1
3.1	2
74.1	3
14.5	4
13.5	5 14.5 13.5

65 Vyhľadávanie podreťazcov v reťazcoch – algoritmus hrubej sily a algoritmus KMP

Na vstupe je textový súbor *input.txt* obsahujúci prípustné znaky z množiny symbolov ('A' – 'Z', 'a' – 'z', '.', ',', '?' '0' – '9'). V súbore *search.txt* sa nachádzajú hľadané podreťazce.

Úloha:
Napíšte program, ktorý pomocou algoritmov hrubej sily a algoritmu KMP zistí pozíciu prvého výskytu podreťazca vo vstupnom súbore. Do výstupného súboru napíše túto pozíciu alebo, ak sa podreťazec v texte nenachádza napíše *–1*.

Príklad:
input.txt
abcdefghasdfgn vbiultcionenc iewcnriouwq cirmnm,wer jkwejqkjrkc jkwervwqkjLKJKLJk kjkljsdkfjjkljsdfjkm,nmn,gnmdf,g.

search.txt
aha
cde

output.txt
–1
2

66 Vyhľadávanie podreťazcov v reťazcoch – algoritmus Boyer–Moore

Na vstupe je textový súbor *input.txt* obsahujúci prípustné znaky z množiny symbolov ('A' – 'Z', 'a' – 'z', '.', ',', '?' '0' – '9'). V súbore *search.txt* sa nachádzajú hľadané podreťazce.

Úloha:
Napíšte program, ktorý pomocou algoritmu Boyer–Moore zistí pozíciu prvého výskytu podreťazca vo vstupnom súbore. Do výstupného súboru napíše túto pozíciu alebo, ak sa podreťazec v texte nenachádza napíše *–1*.

Príklad:
input.txt
abcdefghasdfgn vbiultcionenc iewcnriouwq cirmnm,wer jkwejqkjrkc jkwervwqkjLKJKLJk kjkljsdkfjjkljsdfjkm,nmn,gnmdf,g.

search.txt
aha
cde

output.txt
–1
2

67 Vyhľadávanie podreťazcov v reťazcoch – algoritmus hrubej sily a algoritmus Karp–Rabin.

Na vstupe je textový súbor *input.txt* obsahujúci prípustné znaky z množiny symbolov ('A' – 'Z', 'a' – 'z', '.', ',', '?' '0' – '9'). V súbore *search.txt* sa nachádzajú hľadané podreťazce.

Úloha:
Napíšte program, ktorý pomocou algoritmu Karp–Rabin zistí pozíciu prvého výskytu podreťazca vo vstupnom súbore. Do výstupného súboru napíše túto pozíciu alebo, ak sa podreťazec v texte nenachádza napíše *–1*.

Príklad:
input.txt
abcdefghasdfgn vbiultcionenc iewcnriouwq cirmnm,wer jkwejqkjrkc jkwervwqkjLKJKLJk kjkljsdkfjjkljsdfjkm,nmn,gnmdf,g.

search.txt
aha
cde

output.txt
–1
2

68 Úlohy na grafoch – Dijkstrov algoritmus

Na vstupe je súbor *input.txt*, v ktorom je zadaný ohodnotený orientovaný graf vo forme matice susednosti.

Úloha:
Vytvorte program, v ktorom naprogramujete všetky spomínané algoritmy. V prípade, že sa jedná o ten istý algoritmus s rôznym pomenovaním, uveďte to a naprogramujte ho len raz. Výsledky výpočtu zapíše program do súboru *output.txt*.

Príklad:
input.txt
5
0 4 ∞ ∞ ∞
∞ 0 ∞ ∞ 8
∞ ∞ 0 6 ∞
∞ 1 ∞ 0 ∞
7 ∞ 2 ∞ 0

output.txt
0 4 14 20 12
15 0 10 16 8
22 5 0 6 15
16 1 11 0 9
7 9 2 8 0

69 Úlohy na grafoch – Floydov algoritmus

Na vstupe je súbor *input.txt*, v ktorom je zadaný ohodnotený orientovaný graf vo forme matice susednosti.

Úloha:
Vytvorte program, v ktorom naprogramujete všetky spomínané algoritmy. V prípade, že sa jedná o ten istý algoritmus s rôznym pomenovaním, uveďte to a naprogramujte ho len raz. Výsledky výpočtu zapíše program do súboru *output.txt*.

Príklad:
input.txt
5
0 4 ∞ ∞ ∞
∞ 0 ∞ ∞ 8
∞ ∞ 0 6 ∞
∞ 1 ∞ 0 ∞
7 ∞ 2 ∞ 0

output.txt
0 4 14 20 12
15 0 10 16 8
22 5 0 6 15
16 1 11 0 9
7 9 2 8 0