Programiranje 2
Argumenti komandne linije. Razni zadaci - datoteke, strukture

1 ZADACI

Zadatak 1 *Napisati program koji ispisuje broj navedenih argumenata komadne linije, a zatim i same argumente sa rednim brojevima.*

Primer 1:
```
./a.out danas 63

3
1. ./a.out
2. danas
3. 63
```

Primer 2:
```
./a.out

1
1. ./a.out
```

Primer 3:
```
./a.out -abc -f input.txt

4
1. ./a.out
2. -abc
3. -f
4. input.txt
```

Zadatak 2 *Ako su celi brojevi* `a` *i* `b` *argumenti komandne linije napraviti niz* `A[0] = a, A[1] = a+1, A[2] = a+2, ..., A[b-a] = b` *i ispisati ga. Pretpostaviti da je maksimalna dužina niza 200 elemenata. Proveriti da li* $a < b$ *i* $b - a < 200$ *i ako ovi uslovi nisu ispunjeni ispisati poruku da je došlo do greške. U slučaju da je dato manje ili više argumenata komandne linije ispisati poruku o grešci.*

Primer 1:
```
./a.out 34

greska
```

Primer 2:
```
./a.out 12 20

12 13 14 15 16 17 18 19 20
```

Primer 3:
```
./a.out 30 8

greska
```

Primer 4:
```
./a.out -4 -1

-4 -3 -2 -1
```

Zadatak 3 *Uobičajena praksa na UNIX sistemima je da se argumenti komandne linije dele na opcije i argumente u užem smislu. Opcije počinju znakom - nakon čega obično sledi jedan ili više karaktera koji označavaju koja je opcija u pitanju. Ovim se najčešće upravlja funkcionisanjem programa i neke mogućnosti se uključuju ili isključuju. Argumenti načšće predstavljaju opisne informacije poput na primer imena datoteka. Napisati program koji ispisuje sve opcije koje su navedene u komandnoj liniji.*

Primer 1:
```
./a.out -abc input.txt -d -Fg output

a b c d F g
```

Primer 2:
```
./a.out
```

Primer 3:
```
./a.out ulaz.txt
```

Zadatak 4 *Kao argumenti komandne linje zadate su dimenzije matrice* `A` *(*`m` *i* `n`*). Element matrice se naziva sedlo ako je istovremeno najmanji u svojoj vrsti, a najveći u svojoj koloni. Ispisati indekse i vrednosti onih elemenata matrice koji su sedlo. Pretpostaviti da je maksimalna dimenzija matrice* 50×50*. Ukoliko nisu zadati svi potrebni argumenti komadne linije ispisati poruku da je došlo do greške. Ukoliko su dimenzije van opsega ispisati poruku o grešci.*

```
Primer 1:                  Primer 2:                  Primer 3:          Primer 4:
./a.out 2 3                ./a.out 3 3                ./a.out 3          ./a.out 200 3

1 2 3                      10  3 20                   greska             greska
0 5 6                      15  5 100
                           30 -1 200
0 0 1
                           1 1 5
```

Zadatak 5 *Napisati program koji poredi dva fajla i ispisuje redni broj linija u kojima se fajlovi razlikuju. Imena fajlova se zadaju kao argumenti komandne linije. U slučaju neuspešnog otvaranja datoteka ispisati poruku o grešci. Pretpostaviti da je maksimalna dužina reda u datoteci 200 karaktera. Ukoliko nisu zadati potrebni argumenti komadne linije ispisati poruku o grešci. Linije brojati pov cevši od* `1`.

```
Primer 1:                          Primer 2:                          Primer 3:
./a.out ulaz.txt izlaz.txt         ./a.out primer1.dat primer2.dat    ./a.out prva.dat

ulaz.txt:                          primer1.dat:                       greska
danas vezbamo                      danas vezbamo
programiranje                      analizu
ovo je primer kad su               ovo je primer kad
datoteke iste                      su datoteke razlicite

izlaz.txt:                         primer2.dat:
danas vezbamo                      danas vezbamo
programiranje                      programiranje
ovo je primer kad su               ovo je primer kad su
datoteke iste                      datoteke razlicite

                                   2 3 4
-------------------------------------------------------------------------------------------------
Primer 4:
./a.out prva.dat druga.dat

prva.dat:                            druga.dat:
ovo je primer                        ovaj primer
kada su                              kada su
datoteke                             datoteke
razlicite duzine                     razlicite
                                     duzine
                                     i kada treba ispisati broj
                                     tih redova

1 4 5 6 7
```

Zadatak 6 *Definisati strukturu*

```
typedef struct{
    unsigned int a, b;
    char ime[5];
}_pravougaonik;
```

kojom se opisuje pravougaonik dužinama svojih stranica i imenom. Napisati program koji iz datoteke čije ime se zadaje kao argument komandne linije učitava pravougaonike (nepoznato koliko), a zatim ispisuje imena onih pravougaonika koji su kvadrati i vrednost najveće površine medju pravougaonicima koji nisu kvadrati. U slučaju unosa nekorektnih dužina stranica pravougaonika ili nekorektne vrednosti broja `n`*, ispisati -1 i odmah prekinuti izvršavanje programa. Maksimalan broj pravougaonika je 200.*

```
Primer 1:                  Primer 2:          Primer 3:          Primer 4:          Primer 5:
./a.out pravougaonici.dat  ./a.out dva.dat    ./a.out tri.dat    ./a.out primerx.dat  ./a.out prazna.dat
pravougaonici.dat:         dva.dat:           tri.dat:           primerx.dat:       prazna.dat:
2 4 p1                     5 2 pm             5 5 m              9 7 p
3 3 p2                     4 7 pv             3 3 s
1 6 p3                                        8 8 xl

p2 8                       28                 m s xl             63
```

Zadatak 7 *Ime datoteke dato je kao argument komandne linije. U datoteci se nalaze otvorene i zatvorene zagrade i još nekakav tekst. Proveriti da li su zagrade pravilno uparene. Npr.* `ab( cd) ..` *odgovor je* `jesu`*, a* `..)ba()` *odgovor je* `nisu`*. Ukoliko nisu zadati svi argumenti komadne linije ispisati poruku o grešci.*

```
Primer 1:
./a.out zagrade.txt
zagrade.txt:
ab( cd) ..
((3+4)*5+1)*9

jesu

Primer 2:
./a.out primer2.dat
primer2.dat:
(7+8
nisu(
uparene

nisu

Primer 3:
./a.out primer3.dat
primer3.dat:
)) 7 + 6 ((

nisu

Primer 4:
./a.out

greska
```

Zadatak 8 *Napraviti strukturu* `STUDENT` *koja sadrži:*

- `ime` *(u polju se čuva ime i prezime studenta, napr. "Marko Markovic", maksimalna dužina polja je 100 karaktera),*
- `oc` *(sadrži najviše 10 ocena studenta)*
- `br_ocena` *(ukupan broj ocena za studenata)*
- `pr_oc` *(prosečna ocena)*

U datoteci se nalaze podaci o studentima. Za svakog studenta unosi se ime i prezime razdvojeno razmakom (uputstvo: može se koristiti `strcat` *da spoji ime i prezime koji se mogu pročitati sa specifikatorom* `%s`*), a potom ocene koje se završavaju sa 0. Pronaći studenta koji ima najveći prosek i ispisati sve njegove podatke (prosek ispisati na 2 decimale). Maksimalan broj studenta je 100. Ime datoteke se zadaje kao argument komandne linije.*

```
Primer 1:
./a.out studenti.txt
studenti.txt:
Marko Markovic 5 6 7 8 9 0
Jelena Jankovic 10 10 10 0
Filip Viskovic 10 9 8 7 6 0
Jana Peric 10 10 9 9 8 8 7 7 0

Jelena Jankovic 10 10 10 0 10.00

Primer 2:
./a.out

greska

Primer 3:
./a.out prazna.dat
prazna.dat:
```

Zadatak 9 *a) Napisati C funkciju* `int unesiSkup(char *s, FILE* f)` *kojom se unosi skup elemenata iz datoteke F. Skup se predstavlja kao niz karaktera, pri čemu su dozvoljeni elementi skupa mala i velika slova abecede, kao i cifre. Unos se prekida kada se naiđe na znak za novi red ili nedozvoljeni karakter za skup (maksimalan broj elemenata skupa je 1000). Funkcija vraća broj elemenata skupa koji su uspesno učitani.*

b) Napisati funkciju `void prebroj(char *s, int *br_slova,int *br_cifara)` *kojom se određuje broj slovnih elemenata skupa (velikih ili malih slova) kao i broj cifara u skupu.*

c) Napisati glavni program gde se unose podaci o skupu elemenata. Ime datoteke se zadaje kao argument komandne linije. Na stadardni izlaz ispisati informacije o broju slova i cifara (koristiti funkcije pod a) i b)).

```
Primer 1:
./a.out skup.txt
skup.txt:
abc56ighj9012hjFGHH

broj slova: 13
broj cifara: 6

Primer 2:
./a.out

greska

Primer 3:
./a.out skup2.txt
skup2.txt:
ovdeimamo$dolar

broj slova: 9
broj cifara: 0

Primer 4:
./a.out skup3.txt
skup3.txt:
broJ3
broj5

broj slova: 4
broj cifara: 1
```

Zadatak 10 *Definisati strukturu*

```
typedef struct{
    int x;
    int y;
    int z;
} vektor;
```

kojom se opisuje trodimenzioni vektor. U datoteci `vektori.txt` *nalazi se nepoznati broj vektora (maksimalno ih može biti 200). Učitati ih u niz i ispisuje na standardnom izlazu koordinate vektora sa najvećom dužinom. Dužina vektora se izračunava po formuli:*

$$|v| = \sqrt{x^2 + y^2 + z^2}$$

U slučaju greške ispisati -1 i prekinuti izvršavanje programa.

Primer 1:	***Primer 2:***	***Primer 3:***	***Primer 4:***
2	*67*	*3*	*4*
4 -1 7		*0 0 0*	*3 0 1*
3 1 2		*0 1 0*	*4 5 2*
		1 0 0	*1 0 0*
			2 -1 2
4 -1 7	*-1*	*0 1 0*	*4 5 2*