Piate cvičenie

Základné požiadavky:

- JEDEN súbor obsahujúci celý zdrojový kód, v jazyku C (ANSI C podľa prednášok), s názvom a v štruktúre podľa zverejnených inštrukcií (MSTeams).
- Programy musia komunikovať. Ak program očakáva vstup, musí oznamovať aký vstup sa očakáva. Ak vypisuje výsledok, musí vypisovať zrozumiteľný oznam (napr. čo za hodnotu to vypisuje).
- Formátovanie zdrojového kódu by malo zodpovedať približne príkladom z prednášok. Odsadzovanie textov je základ. Príklad dobrého a zlého formátovania sú v prednáške číslo dva na konci.

Úloha prvá: Píšeme do súboru po znaku.

Napíšte procedúru, ktorá bude čítať z klávesnice znaky. Použite **getchar()** ! Znaky číta, kým nie je stlačený znak bodka. Uvedomte si, že reálne sa celé spracovanie vstupu z klávesnice začína až stlačením enteru. Ak teda napíšete riadok bez bodky, začne sa spracovanie, ale cyklus neskončí. Ak naopak stlačíte bodku, program hneď neskončí, pretože stále sa zadávanie znakov musí ukončiť enterom. Použitie getchar() nám umožní zapisovať do súboru aj entery!

Prečítané znaky program upravuje a zapisuje DO SÚBORU **vystup.txt**. Úpravy znakov sú nasledovné:

a. veľké písmená zmení na malé,
b. malé písmená zmení za bodky,
c. číselné znaky (0-9) zmení za pomlčku
d. entery ponechá
e. všetky ostatné znaky zmení za hviezdičku

.

Ukážkový vstup z klávesnice:

```
abc4DEF↵
ab=/4/C.fffff↵
```

Ukážkový výstup v súbore:

```
...-def↵
..**-*c
```

Ako z príkladu vidíte, ak by som aj za bodkou (v druhom riadku) niečo zadal (čo môžem), to sa už neprenesie do súboru. Neprenesie sa tam ani koncový enter toho riadku za bodkou.

Úloha druhá: Čítame zo súboru do poľa.

Napíšte procedúru, ktorá číta čísla zo súboru **cisla.txt** do poľa celých čísel. Predpokladajte, že čísel nebude viac ako 100 (použite statické pole aj v prípade, že viete používať dynamické). Pomôcka pre čítanie – fscanf – nie že niekoho napadne čítať po znaku a nejak z toho dekódovať čísla.
Program následne vypíše **!!!!párne!!!!** čísla poľa **za sebou** na obrazovku v obrátenom poradí ako boli v súbore, ale bez enterov.

Príklad súboru **cisla.txt**:
1 3 2 10↵
6 7↵
16 7 4↵

Výstup na obrazovke
4 16 6 10 2↵

Ďalšie zaujímavé príklady na precvičenie !!!

3. Napíšte program, ktorý načíta reálne číslo **X** nasledované koncom riadku. Do súboru `nasobky.txt` zapíše 1, 2, ..., 10- násobky čísel **X, 2*X a X*X**. Súbor má obsahovať 10 riadkov s nasledujúcim formátovaním: v `i`-tom riadku vypíšte
   `i:i*x,i*2*x,i*x*x`↵

   kde `i` je číslo riadku zarovnané na 2 miesta, nasledujú príslušné násobky `x` zarovnané na celkovú dĺžku 8 vypísane na na 2 desatinné miesta (použite formátovacie reťazce!!). Každý riadok je ukončený znakom konca riadku.

   Ukážkový vstup:
   `Zadajte realne cislo: 2.5`↵

   Ukážka súboru `nasobky.txt`:
```
 1:    2.50,    5.00,    6.25↵
 2:    5.00,   10.00,   12.50↵
 3:    7.50,   15.00,   18.75↵
 4:   10.00,   20.00,   25.00↵
 5:   12.50,   25.00,   31.25↵
 6:   15.00,   30.00,   37.50↵
 7:   17.50,   35.00,   43.75↵
 8:   20.00,   40.00,   50.00↵
 9:   22.50,   45.00,   56.25↵
10:   25.00,   50.00,   62.50↵
```

4. Napíšte program, ktorý z klávesnice načíta znak nasledovaný koncom riadku. Ďalej číta znaky zo súboru `znak.txt`. Ak program prečítal z klávesnice `'s'`, vypisuje načítané znaky do súboru `novy.txt`. Ak načítal ľubovoľný iný znak, vypisuje načítané znaky na štandardný výstup (obrazovku). Súbor `novy.txt` alebo

štandardný výstup bude teda obsahovať presnú kópiu obsahu súboru `znak.txt`. Povinne použite **stdout** (štandardný výstup) tak, že kopírovanie vstupného súboru na obrazovku alebo do výstupného súboru je realizované v rovnakom cykle (podobný program je v prednáškach).

5. Napíšte program, ktorý určí, či majú dva súbory `prvy.txt` a `druhy.txt` rovnaký obsah. Ak majú súbory rovnaký obsah, program vypíše `Subory su identicke`. Ak súbory rovnaký obsah nemajú, vypíše program `Pocet roznych znakov:` nasledovaný medzerou, počtom rôznych znakov v súboroch a ukončený koncom riadku. i-ty znak v jednom súbore považujte za rôzny od i-teho znaku v druhom súbore, ak oba znaky existujú (t.j. ani jeden súbor nemá menej ako i znakov) a príslušné znaky sa nerovnajú. Ak majú súbory nerovnakú dĺžku, na výstup program vypíše ešte jeden riadok obsahujúci správu `Jeden zo suborov je dlhsi o x znakov`. Pričom `x` je počet znakov o ktoré je jeden zo súborov dlhší. Správa je nasledovaná koncom riadku.
   Ukážka súboru `prvy.txt`:
   ```
   ahoj
   ```
   Ukážka súboru `druhy.txt`:
   ```
   ahuj svet
   ```
   Ukážkový výstup:
   ```
   Pocet roznych znakov: 1↵
   Jeden zo suborov je dlhsi o 5 znakov↵
   ```

6. Napíšte program, ktorý číta znaky zo súboru `vstup.txt` po riadkoch. Každý riadok prepíše do súboru `CISLA.TXT`. Po každom prepísanom riadku na ďalšom riadku uvedie počet malých písmen z prečítaného riadku. Ak súbor už predtým existoval a obsahoval nejaké dáta, program tieto dáta nezmaže a svoj výstup napíše na koniec súboru `cisla.txt`. Program nečíta žiaden vstup zo štandardného vstupu a nevypisuje žiaden výstup na štandardný výstup. Predpokladajte, že posledný riadok je vždy ukončený koncom riadku.
   Ukážka súboru `vstup.txt`:
   ```
   ahoj123↵
   x*Y*z↵
   ```
   Ukážka súboru `cisla.txt` pred spustením programu:
   ```
   qwerty↵
   6↵
   ```
   Ukážka súboru `cisla.txt` po spustení programu:
   ```
   qwerty↵
   6↵
   ahoj123↵
   4↵
   x*Y*z↵
   2↵
   ```

7. Napíšte program, ktorý bude čítať znaky zo súboru `text.txt` pokiaľ nenačíta znak `'*'`. Ak načíta znak `'x'` alebo `'X'` vypíše `Precital som X`, ak znak `'y'` alebo `'Y'` vypíše `Precital som Y`, ak načíta znaky `'#'`, `'$'` alebo `'&'` vypíše

`Precital som riadiaci znak` a ak načíta znak `'*'` vypíše `Koniec` a skončí čítanie súboru. Po prečítaní súboru vypíše správu `Pocet precitanych medzier:` nasledovanú medzerou a počtom prečítaných medzier. Každá správa je nasledovaná koncom riadku.
Ukážka súboru `text.txt`:

```
$ abc 5 xyz #↵
& Q *# abf
```

Ukážkový výstup:

```
Precital som riadiaci znak↵
Precital som X↵
Precital som Y↵
Precital som riadiaci znak↵
Precital som riadiaci znak↵
Koniec↵
Pocet precitanych medzier: 6↵
```

8. Napíšte program, ktorý načíta dve celé čísla, oddelené medzerou, nasledované znakom konca riadku. Program vypočíta súčet celých čísel, ktoré sa nachádzajú medzi zadanými číslami. Výstupom je jeden riadok obsahujúci celé číslo a znak konca riadku. Ak bude prvé načítané číslo väčšie ako druhé načítané číslo, tak čísla vymeňte. Ak sa medzi zadanými číslami nenachádza žiadne celé číslo, vypíšte správu: `Neda sa vypocitat↵`. Použite cyklus. (Poznámka: viete použiť vzorec a vypočítať výsledok bez použitia cyklu?)
Ukážkový vstup:

```
3 7↵
```

Ukážkový výstup:

```
15↵
```

9. Napíšte program, ktorý načíta celé číslo nasledované koncom riadku. Výstupom programu je faktoriál načítaného čísla nasledovaný znakom konca riadku.
Ukážkový vstup:

```
5↵
```

Ukážkový výstup:

```
120↵
```

10. Napíšte program, ktorý načíta 2 celé čísla `p` a `k (0 < p, k < 100)` a vypíše čísla od 1 do p nasledovne: Ak je číslo deliteľné číslom `k`, vypíše sa na 2 miesta, inak sa vypíšu dve pomlčky za sebou. Medzi číslami (vypísanými na 2 miesta) a pomlčkami je vždy 1 medzera. Výpis je ukončený znakom konca riadku.
Ukážkový vstup: `10 2↵`
Ukážkový výstup: `1 -- 3 -- 5 -- 7 -- 9 --↵`

11. V nasledujúcom programe zmeňte cyklus `for` na `while`:

```
#include <stdio.h>

void main()
{
   int i;

   for (i = 0; i < 10; i++)
      printf("%d. \n", i+1);
}
```

12. Napíšte program, ktorý načíta celé číslo `n` nasledované znakom konca riadku. Potom načíta postupnosť `n` celých čísel, každé nalsedované znakom konca riadku. Program určí, či načítaná postupnosť čísel je správna. Postupnosť je správna, ak:

    a. Prvé číslo je z rosahu `<0, 10>`
    b. Pre každé `i`-te číslo (`i∈<2,n>`) platí, že nie je väčšie ako dvojnásobok predchádzajúceho (`i-1`)-ho čísla, ani menšie ako polovica predchádzajúceho (`i-1`)-ho čísla.

    Ak je postupnosť správna, vypíše program správu `Postupnost je spravna` a odriadkuje, inak vypíše `Postupnost nie je spravna` a odriadkuje.
    Ukážkový vstup:
    `3↵`
    `5↵`
    `7↵`
    `9↵`
    Ukážkový výstup:
    `Postupnost je spravna↵`

13. Napíšte program, ktorý načíta 3 celé čísla `n`, `s`, `v` oddelených medzerami. Ak je `n < 1`, `n > 15`, `n` je párne číslo, alebo `s` a `v` nie sú z intervalu `<1, 5>`, program vypíše chybu `Zly vstup` a skončí. Ak bude program pokračovať, zo znakov `'-'` a číslic nakreslí `s×v` obrázkov (s vedľa seba a v pod seba) rovnoramenných trojuholníkov s výškou `n` tak, ako je uvedené v príklade 7.
    Ukážkový vstup:
    `3 3 2↵`
    Ukážkový výstup:
    `1--1--1--↵`
    `22-22-22-↵`
    `333333333↵`
    `22-22-22-↵`
    `1--1--1--↵`
    `1--1--1--↵`
    `22-22-22-↵`
    `333333333↵`
    `22-22-22-↵`
    `1--1--1--↵`