Štvrté cvičenie

Všetky programy vhodne štrukturujte. V úvode programu uveďte komentár vyjadrujúci, čo program robí, kto a kedy ho vytvoril. Dôležité riadky programu okomentujte. Pri funkcii `main()` používajte návratový typ `int`. Vstupno/výstupnú špecifikáciu dodržiavajte presne. Pre lepšiu zrozumiteľnosť používame pri ukážkach vstupu a výstupu znak konca riadku ↵, ktorý je na vstupe a výstupe neviditeľný.

1. Napíšte program, ktorý bude čítať z obrazovky znaky pokiaľ nie je stlačená hviezdička. Prečítané znaky upraví a vypíše na obrazovku. Výstup ukončí znakom konca riadku. Úpravy znakov sú nasledovné:
   a. malé písmená zmení na veľké,
   b. veľké písmená nechá nezmenené,
   c. tabulátor (`'\t'`) a nový riadok (`'\n'`) zmení na medzeru a
   d. všetky ostatné znaky (okrem hviezdičky) nahradí znkom `'-'`.

   Ukážkový vstup:
   ```
   abc4DEF↵
   -//      \\+*
   ```
   Ukážkový výstup:
   ```
   ABC-DEF --- ---*↵
   ```

2. 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↵
   ```

3. 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↵
   ```

4. 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 --↵
   ```

5. 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);
}
```

6. 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`↵

7. Napíšte program, ktorý načíta číslo `n`. Ak je `n < 1`, `n > 15` alebo je `n` párne číslo, program vypíše chybu `Zly vstup` a skončí. Ak bude program pokračovať, zo ‘-‘ a číslic nakreslí rovnoramenný trojuholník s výškou `n`, tak, že v každom riadku bude jedna číslica určujúca počet číslic v riadku (ak je v riadku dvojciferný počet číslic, číslica sa vypočíta ako počet modulo 10).
   Ukážkový vstup:
   `5`↵
   Ukážkový výstup:
   `1----`↵
   `22---`↵
   `333--`↵
   `4444-`↵
   `55555`↵
   `4444-`↵
   `333--`↵
   `22---`↵
   `1----`↵

8. 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--↵
   ```

9. Napíšte program, ktorý načíta reálne číslo `x` nasledované koncom riadku. Do súboru `nasobky.txt` zapíše 1, 2, ..., 10- násobky čísla `x`. Súbor má obsahovať 10 riadkov s nasledujúcim formátovaním: v `i`-tom riadku vypíšte `i * x = ix`, kde `i` je číslo riadku na 2 miesta, `x` je načítané číslo vypísané na 2 desatinné miesta a `ix` je `i`-ty násobok čísla `x` tiež vypísaný na 2 desatinné miesta. Každý riadok je ukončený znakom konca riadku.
   Ukážkový vstup:
   ```
   2.5↵
   ```
   Ukážka súboru `nasobky.txt`:
   ```
    1 * 2.50 = 2.50↵
    2 * 2.50 = 5.00↵
    3 * 2.50 = 7.50↵
    4 * 2.50 = 10.00↵
    5 * 2.50 = 12.50↵
    6 * 2.50 = 15.00↵
    7 * 2.50 = 17.50↵
    8 * 2.50 = 20.00↵
    9 * 2.50 = 22.50↵
   10 * 2.50 = 25.00↵
   ```

10. Napíšte program, ktorý zo štandardného vstupu (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`.
    Ukážkový vstup:
    ```
    s↵
    ```

Ukážka súboru `znak.txt`:
```
abrakadabra↵
bubu
```
Ukážka súboru `novy.txt`:
```
abrakadabra↵
bubu
```

11. 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↵
    ```

12. 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↵
    ```

13. Napíšte program, ktorý určí, či majú dva súbory `prvy.txt` a `druhy.txt` rovnaký obsah. Program nečíta žiadne dáta zo štandardného vstupu. 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`:
    ```
    ahujx↵
    *
    ```
    Ukážkový výstup:
    ```
    Pocet roznych znakov: 1↵
    Jeden zo suborov je dlhsi o 3 znakov↵
    ```