Siedme 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. Šifrujeme správy.

Substitučný šifrovací algoritmus spočíva v nahradení každého písmena správy iným písmenom podľa kľúča. Kľúčom môže byť slovo, kde každé písmeno určuje, ako sa nahradia jednotlivé písmená správy. Ak je napríklad kľúčovým slovom CAESAR, tak keďže C je 3. písmenom abecedy, prvé písmeno správy sa nahradí písmenom, ktoré je v abecede o 3 písmená ďalej. Druhé písmeno sa nahradí písmenom, ktoré za ním v abecede nasleduje, lebo A je 1. písmenom abecedy a tak ďalej. V prípade, že presiahneme dĺžku abecedy, pokračujeme opäť od začiatku abecedy. Keď je správa dlhšia ako kľúč, kľúč sa postupne zopakuje.

Napíšte funkciu sifruj (dodržujte prototyp)

char *sifruj(char *sprava, char *heslo)

ktorá dostane v parametroch správu a heslo (kľúč). Správu zašifruje, správa je zároveň aj výstupom funkcie (pomôcka: return sprava, nič nie je nutné alokovať, správa sa mení priamo v reťazci a funkcia len smerník na začiatok rovnakého reťazca aký dostala – len je zmenený jeho obsah). Funkciu demonštrujte v hlavnom programe tak, že necháte používateľa vložiť správu a kľúč (heslo) a vypíšete výsledok šifrovania.

```
Zadajte text na sifrovanie: ETTUBRUTE↲
Zadajte sifru / heslo: CAESAR↲
vystup sifrovania: HUYNCJXUJ↲

Zadajte text na sifrovanie: AAAAAABBBBBB↲
Zadajte sifru / heslo: ABCDEF↲
vystup sifrovania: BCDEFGCDEFGH↲
```

Úloha druhá. Dešifrujeme správy.

Napíšte funkciu

char *odsifruj(char *sprava, char *heslo)

ktorá odšifruje správu zašifrovanú v príklade 1. Funkcia zase vracia smerník na správu aj na výstupe (aby sa dala pekne používať v reťazi volaní). V hlavnom programe funkcie (ak budete robiť obe) použite za sebou. Odporúčaná schéma programu a volania funkcií:

```
vložene vstupov: správa a heslo
printf("vystup sifrovania: %s", sifruj(sprava, heslo));
printf("vystup desifrovania: %s", odsifruj(sprava, heslo));
```

Edukačná poznámka: Uvedomte si, že funkcie menia obsah reťazca **sprava**. Ten nie je potrebné nejak špeciálne priraďovať. Smerník máte neustále k dispocií v premennej **sprava**. To, že funkcie vracajú smerník na správu sme využili na pekné použitie v spojení s **printf**.

Úloha tretia. Analyzujeme súbor.

Napíšte funkciu analyzuj, povinne použite prototyp:

```
int analyzuj(char *subor, int *riadky, int *medzery, int *cisla)
```

kde **subor** je názov textového súboru na analýzu. Funkcia vypočíta a cez parametre vráti:

- počet riadkov súboru
- počet medzier súboru
- počet číslic v súbore (0-9)
- funkcia ako výstup vráti veľkosť súboru (počet znakov)
- ak sa funkcii nepodarí otvoriť súbor, vráti **-1**

povinné použitie funkcie v main()

```
int riadky, medzery, cisla, velkost;
….
velkost = analyzuj("vstup.txt", &riadky, &medzery, &cisla);
if (velkost == -1) printf("subor sa nepodarilo otvorit!")
else printf("subor ma velkost %d, pocet riadkov %d, medzery: %d a cisla %d",  velkost, riadky, medzery, cisla);
```

Do funkcie vstupuje NÁZOV súboru, nie štruktura FILE. Ak súbor vo funkcii otvárate, aj ho nezabúdajte zatvoriť. Neotvárajte súbor 2x, na skúške strhávame za to body.

Iné zadania na precvičenie:

1. Napíšte program, ktorý načíta reťazec (povinne použite scanf("%s", ...) a vytvorí nový v ktorom každú **hviezdičku** zdvojí a každú **bodku** vymaže. Na obrazovku vypíše výsledný reťazec (pomocou jediného printf). Celé riešenie musí používať jediné pole (reťazec).
   Ukážkový vstup:
   `ab*d.f*h↵`

Ukážka výstupu:
```
ab**df**h↵
```

2. Napíšte program, ktorý načíta súbor a vytvorí nový súbor, v ktorom budú všetky riadky pôvodného súboru uložené v obrátenom poradí (otočené budú znaky v rámci riadku, nie poradie riadkov v súbore).
   Ukážkový vstup:
   ```
   1234567890↵
   abcdef↵
   ```
   Ukážka výstupu:
   ```
   0987654321↵
   fedcba↵
   ```

3. Napíšte program, ktorý na obrazovku vypíše dĺžku najkratšieho riadka zo súboru `list.txt`. Ak sa súbor nepodarí otvoriť, program vypíše správy `Chyba: otvaranie suboru`. Ak sa súbor nepodarí zatvoriť, program vypíše správu `Chyba: zatvaranie suboru`. Každá správa je ukončená znakom konca riadku.
   Ukážkový súbor `list.txt`:
   ```
   Mily Jezisko,↵
   Na Vianoce si prajem bager.↵
   Janko
   ```
   Ukážkový výstup:
   ```
   5↵
   ```

4. Napíšte program, ktorý načíta číslo `n`. Ak je `n < 3`, `n > 15` alebo je `n` párne číslo, program vypíše na obrazovku správu `Zly vstup` a skončí. Ak bude program pokračovať, zo znakov `'*'`, `'o'` a `'-'` nakreslí do súboru `obrazok.txt` obrázok (pripomínajúci českú vlajku) s rozmermi `(2n-1)x(2n-1)` podľa ukážkového výstupu. Ošetrite otvorenie a zatvorenie súboru. Ak sa súbor nepodarí otvoriť, program vypíše správy `Chyba: otvaranie suboru`. Ak sa súbor nepodarí zatvoriť, program vypíše správu `Chyba: zatvaranie suboru`. Každá správa je ukončená znakom konca riadku.
   Ukážkový vstup:
   ```
   5↵
   ```
   Ukážkový výstup v súbore `obrazok.txt`:
   ```
   *--------↵
   **-------↵
   ***------↵
   ****-----↵
   *********↵
   ****ooooo↵
   ***oooooo↵
   **ooooooo↵
   *ooooooooo↵
   ```

5. Napíšte program, ktorý zo štandardného vstupu načíta celé číslo `n`, ($0 \leq$ `n` ≤ 50) a následne načíta `n` reálnych čísel, ktoré zapíše do poľa. Pomocou ukazovateľovej aritmetiky vypíšte všetky záporné čísla v poli na 2 desatinné miesta, každé v zvlášť riadku (v poradí v akom sú v poli zapísané). **Nepoužívajte indexy.**

   Ukážkový vstup:
   ```
   3↵
   1.5↵
   -3.8↵
   0.2↵
   ```
   Ukážkový výstup:
   ```
   -3.80↵
   ```

6. Napíšte program, ktorý bude pracovať s 3 poliami celých čísel: `cisla`, `kladne`, `zaporne`. Program zo štandardného vstupu načíta celé číslo `n`, ($0 \leq$ `n` ≤ 50) a následne načíta `n` celých čísel, ktoré zapíše do poľa `cisla`. Ďalej prekopíruje kladné čísla do poľa `kladne` a záporné čísla do poľa `zaporne`. Na konci program vypíše obsah polí `kladne` a `zaporne` formátované podľa ukážkového výstupu. **Nepoužívajte indexy, použite ukazovateľovú aritmetiku.**

   Ukážkový vstup:
   ```
   5↵
   -2↵
   4↵
   1↵
   -5↵
   3↵
   ```
   Ukážkový výstup:
   ```
   Kladne: 4 1 3↵
   Zaporne: -2 -5↵
   ```

7. Daný je neprázdny reťazec `str` párnej dĺžky a znak `c`. Napíšte funkciu `vloz_do_stredu(char *str, char c)`, ktorá do stredu reťazca `str` vloží znak `c`. Napr. pre `str="ABBA"` a znak `c='+'` bude upravený reťazec `"AB+BA"`. Predpokladajte, že pole znakov `str` má dostatok miesta pre ďalší znak. Hlavná funkcia `main()` načíta hodnoty `str` a `c` zo vstupu a vypíše upravený reťazec `str`.

   Ukážka vstupu:
   ```
   ABBA +
   ```
   Výstup
   ```
   AB+BA
   ```
   Ukážka vstupu:
   ```
   Nira t
   ```
   Výstup pre ukážkový vstup:
   ```
   Nitra
   ```

8. Napíšte funkciu `odstran_male_pismena(char *str)`, ktorá vo vstupnom reťazci `str` odstráni písmená malej anglickej abecedy. Napr. reťazec `"SlovenskaRepublika"` upraví na `"SR"`. Hlavná funkcia `main()` načíta hodnoty `str` zo vstupu a vypíše upravený reťazec `str`.

   Ukážka vstupu:
   ```
   SlovenskaRepublika
   ```
   Ukážkový vstup:
   ```
   SR
   ```