Tartalomjegyzék

Ismétlés

- Filekezelés
  - fopen, fprintf, fscanf, fclose
  - FILE *
  - write, add, read (w, a, r)
  - a kiíró/beolvasó függvények használata akár a parancssoriaké

- Program argumentumai
  - int argc, char* argv[]
  - kezdetben minden szöveg, atoi-val int-é, atof-el float-á konvertálás
  - 0. indexû a file neve, 1. indexû az elsõ argumentum

- Typedef és struktúrák
  - több változót egységbe zárni, pl vektor, komplex szám
  - . operátorral elérhetõ adattagok
  - létrehozásuk és függvény paraméter/visszatérési érték használatuk akár mint egy beépített változónak

Stringek

A string *típus* szöveges tartalmat tárol, valójában egy karakter tömb (char[]), egy lezáró nullával ('\0') a végén.

Különbség az 'a' és az "http://wiki.math.bme.hua"http://wiki.math.bme.hu között:

- 'a' char típusú, "http://wiki.math.bme.hua"http://wiki.math.bme.hu char[] (vagyis char *) típusú
- az "http://wiki.math.bme.hua"http://wiki.math.bme.hu 2 karaktert tárol egy 'a' és egy '\0'

Könnyen létrehozhatunk egy konstans karakter tömböt:

```
char s[] = "kiskutya";
```

Ez ekvivalens a már korábban tanult tömb megadással:

```
char s[] = {'k', 'i', 's', 'k', 'u', 't', 'y', 'a', '\0'};
```

Ezekben az esetekben nem adtuk meg a tömbök hosszát, így az minimális lesz, tehát 9 hosszú. Ha továbbra is másra is szeretnénk használni a string-ünket, akkor lehet hosszabbra szeretnék deklarálni, ekkor megadható a mérete:

```
char s[30] = "kiskutya";
```

Feladatok

1. Hossz

Írjatok függvényt mely egy string-et kap bemenetként, int-et ad vissza és kiszámolja hogy milyen hosszú a string (hány karakterbõl áll). Segítség: minden string végén ott a lezáró nulla.

2. Egyezések

Írjatok függvényt mely egy string-et és egy karaktert kap bemenetként, int-et ad vissza és kiszámolja hogy hányszor szerepel az adott karakter a string-ben, és visszaadja ezt a számot.

3. atoi

Írjatok függvényt mely megvalósítja a már ismert atoi függvényt (az atoi használata nélkül). Azaz string-et kap bemenetként, és a string-ben tárolt szám számértékét adja vissza int-ként. (Azaz ha "http://wiki.math.bme.hu95"http://wiki.math.bme.hu van a string-ben akkor 95-öt ad vissza.)

String.h

A string.h tartalmaz pár hasznos függvényt mellyel string-eket lehet manipulálni, a legfontosabbak:

- strlen(const char *): az 1. feladatot valósítja meg, azaz visszaadja a kapott string hosszát. (a const csak arra utal, hogy a string tartalmát nem fogja megváltoztatni)
- strcpy(char *cel, const char *forras): átmásolja a forras string tartalmát a cel string-be
- strcat(char *cel, const char *forras): hozzáfûzi a forras string tartalmát a cel string-hez
- strcmp(const char *str1, const char *str2): összehasonlítja az str1-et és az str2-t, ha 0-t ad akkor megegyeznek, pozitívat ad, ha az elsõ nem egyezésnél az elsõ string-ben volt nagyobb karakter, negatívat különben

Egy példa az strcat-re:

```
char fname[30] = "Bob";
char lname[30] = "by";

printf("%s", strcat(fname, lname));
```

Feladatok

4. Próba

Próbáljátok ki a fenti függvényeket, ehhez segítség:

```
char str1[] = "Volt egyszer ";
char str2[] = "egy kiskutya, elment";
char str3[] = "a vasarba.";

// Hozzatok letre olyan string-et amibe el fog ferni ha mind3 string-et osszefuzzuk.
// Majd tegyetek is meg ezt az osszefuzest
// Szamoljatok meg ha meg nem tettetek, hogy hany karakteru ez az uj string
// Masoljatok egy pont akkora string-be mint amekkora kell neki (mivel a hosszt az strlen-el hatar
// Vegul hasonlitsatok ossze a ket string-et hogy valoban megegyeznek-e
```

5. strcmp

Írjatok meg egy az strcmp-nek megfelelõ függvényt, azaz ami megkap két stringet és 1-et ad vissza ha az els? a nagyobb (lexikografikus rendezés, vagyis ABC-rend szerint), -1-et ad ha a második a nagyobb, és 0-t ad ha egyforma a két string.

Ne felejtsétek el tesztelni, akár a 4. feladat string-jeivel akár újakkal.