Karakterláncok kezelése

Kormányos Andor

Komplex Rendszerek Fizikája Tanszék

2022 október 11.

Szöveges fájlok kezelése

A szöveges fájlok valamilyen karakterkódolással vannak tárolva

- a C nyelv alapértelmezésben a rendszer egybájtos kódolását használja
- adatfájloknál a legegyszerűbb valami külső programmal egybájtosra konvertálni
- linuxon pl. az `iconv` programmal

A szöveges fájlok sorokból állnak

- a sorok végén új sort jelölő karakter (`\r`, `\n` stb.)
- de egy sor *akármilyen hosszú lehet*!

Alapvető függvények szöveges fájlok olvasására

- `fscanf`: formátumozott olvasás (korábban már láttuk)
- `fgets`: egyetlen sor beolvasása
  első új sor jelig, a fájl végéig, vagy korlátos karakterszámig
- `fgetc`: egyetlen karaktert olvas be

Bájt szintű, bináris olvasás

- `fread`: megadott számú bájtot olvas (vagy a fájl végéig)

Egyszerű beolvasás fscanf-fel

Ha a feladat csupán azonos típusú adatok egymás utáni beolvasása

- legegyszerűbben a fscanf függvény hívogatásával
- fscanf átugorja a whitespace-eket!

```
1   int main () {
2       int n=20;
3       double act_num;
4
5       FILE *fp;
6       fp = fopen("inpfile.txt", "r");
7
8       for (int i=0; i<n;i++)
9       {
10      fscanf(fp, "%lf", &act_num);
11      printf("current number %lf",act_num)
12      }
13
14      close(f);
15    return(0);
16  }
```

Egyszerű beolvasás `fscanf`-fel

Pl: dátumok beolvasása, amelyek egymás alatti sorokban helyezkednek el “év hónap nap” formátumban

2017 augusztus 7
2019 február 8
:
:
stb.

```
1   #inlcude<stdio.h>
2   #inlcude<stdlib.h>
3
4   int main () {
5      char str1[20];
6      int year, day;
7      FILE *fp;
8
9      fp = fopen("inpfile.txt", "r");
10
11     while (fscanf(fp, "%d  %s  %d", &year, str1, &day)==3)  // az fscanf visszateresi erteke integer
12     {
13       printf("%d  %s  %d\n",year, str1, day);
14       // beolvasott adatok feldolgozasa
15     }
16
17    return(0);
18  }
```

Mátrix beolvasása szöveges fájlból soronként

Tekintsük a következő példa fájlt:

- a fájl elején fejléc, ahol a sorok '#'-sel kezdődnek
- az adatok egy oszlopban helyezkednek el
- a mátrix egy sorába tartozó számok egymás alatt vannak
- a különböző sorokhoz tartozó számokat tartalmazó adattömbök között egy üres sor van

Mátrix beolvasása szöveges fájlból soronként

Tekintsük a következő példa fájlt:

- a fájl elején fejléc, ahol a sorok '#'-sel kezdődnek
- az adatok egy oszlopban helyezkednek el
- a mátrix egy sorába tartozó számok egymás alatt vannak
- a különböző sorokhoz tartozó számokat tartalmazó adattömbök között egy üres sor van

```
1
2     FILE *finp = fopen(argv[1],"r");
3
4      if(finp == NULL) {
5         printf("Error opening input file");
6         return(-1);}
7
8
9      while (fgets(str,maxstrl,finp)!=NULL )
10     {
11       if (str[0]=='#')
12       {printf("%s",str);}
13       else if (strlen(str)>1)
14       {
15        if (vektindx>maxdim)
16        {
17         printf("Max vector size reached\n");
18         exit(-1);
19        }
20        else {
21        matr[vektindx]=atof(str);
22        vektindx++;
23        }
24       }
25     }
```

- `fgets`-sel olvasunk be sorokat, amig a fájl végére nem érünk.
- legfeljebb `maxstrl` karaktert olvasunk be soronként
- `char str[maxstrl]` stringtömbe kerül az beolvasott sor

Mátrix beolvasása szöveges fájlból soronként

Tekintsük a következő példa fájlt:

- a fájl elején fejléc, ahol a sorok '#'-sel kezdődnek
- az adatok egy oszlopban helyezkednek el
- a mátrix egy sorába tartozó számok egymás alatt vannak
- a különböző sorokhoz tartozó számokat tartalmazó adattömbök között egy üres sor van

```
1
2     FILE *finp = fopen(argv[1],"r");
3
4      if(finp == NULL) {
5         printf("Error opening input file");
6         return(-1);}
7
8
9      while (fgets(str,maxstrl,finp)!=NULL )
10     {
11       if (str[0]=='#')
12       {printf("%s",str);}
13       else if (strlen(str)>1)
14       {
15        if (vektindx>maxdim)
16        {
17         printf("Max vector size reached\n");
18         exit(-1);
19        }
20        else {
21        matr[vektindx]=atof(str);
22        vektindx++;
23        }
24       }
25     }
```

- `fgets`-sel olvasunk be sorokat, amig a fájl végére nem érünk.
- legfeljebb `maxstrl` karaktert olvasunk be soronként
- `char str[maxstrl]` stringtömbe kerül az beolvasott sor
- ha az első karakter '#', akkor azt a sort írja ki

Mátrix beolvasása szöveges fájlból soronként

Tekintsük a következő példa fájlt:

- a fájl elején fejléc, ahol a sorok '#'-sel kezdődnek
- az adatok egy oszlopban helyezkednek el
- a mátrix egy sorába tartozó számok egymás alatt vannak
- a különböző sorokhoz tartozó számokat tartalmazó adattömbök között egy üres sor van

```
1
2     FILE *finp = fopen(argv[1],"r");
3
4      if(finp == NULL) {
5         printf("Error opening input file");
6         return(-1);}
7
8
9      while (fgets(str,maxstrl,finp)!=NULL )
10     {
11       if (str[0]=='#')
12       {printf("%s",str);}
13       else if (strlen(str)>1)
14       {
15        if (vektindx>maxdim)
16        {
17         printf("Max vector size reached\n");
18         exit(-1);
19        }
20        else {
21        matr[vektindx]=atof(str);
22        vektindx++;
23        }
24       }
25     }
```

- `fgets`-sel olvasunk be sorokat, amig a fájl végére nem érünk.
- legfeljebb `maxstrl` karaktert olvasunk be soronként
- `char str[maxstrl]` stringtömbe kerül az beolvasott sor
- ha az első karakter '#', akkor azt a sort írja ki
- ha a sor nem üres, akkor alakítsa a stringet `float` számmá és helyezze el