Programozás 2

Fájl, fájlkezelés. Szöveges és bináris állományok.

Miért van szükség fájlokra?

- **Program és felhasználó/külső környezet közötti adatcsere.**
- **A program adatainak hosszú távú tárolása (ellentétben a memóriával).**

Néhány példa:

- Naplózás, naplófájlok (log.txt)
- Konfigurációs fájlok (settings.cfg)
- Eredményadatok mentése (output.csv)

Mi az a fájl?

A fájl bájtok sorozata.

Két fő fájltípus különböztetünk meg:

- **Szöveges fájl**: olvasható karakterek (.txt, .csv)
- **Bináris fájl**: gépi formátumú adatok (.dat, .bin)

Hogyan tudunk fájlba írni vagy onnan olvasni?

A fájlkezeléshez szükséges konstans-, típus- és függvénydeklarációk az **<stdio.h>** fejállományban találhatók.

Az állományok tartalmának eléréséhez az alábbi lépéseket szükséges megtennünk:

1. a fájl változó definiálása,
2. az állomány megnyitása,
3. az állomány tartalmának feldolgozása (írás, olvasás, pozicionálás),
4. az állomány bezárása.

1. A fájl változó definiálása

Minden fájlművelethez szükség van egy **FILE*** típusú mutatóra.

```
#include <stdlib.h>
#include <stdio.h>

FILE *fp;  // fájlmutató
```

Megjegyzés: az **<stdlib.h>** fejállományban található az **exit()** függvény. Ez nem közvetlenül a fájlkezeléshez szükséges, mi ennek segítségével állítjuk le a program futását, ha az állomány megnyitásánál hiba keletkezik.

2. Az állomány megnyitása

Az állomány az **fopen()** függvény segítségével nyitható meg:

FILE *fopen(const char *filename, const char *mode)

- Ennek első paramétere (**filename**) az állomány neve.
- A második paraméter (**mode**) az állomány megnyitásának módja (pl. megnyitás írásra “**w**", vagy olvasásra “**r**").

```
fp = fopen("adatok.txt", "r");

if (fp == NULL) {
    printf("Hiba a fajl megnyitasakor!");
    exit(0);
}
```

Lehetséges hozzáférési módok (**mode**):

Szöveges fájl	Bináris fájl	
"r" vagy **"rt"**	**"rb"**	Létező fájl megnyitása olvasásra.
"w" vagy **"wt"**	**"wb"**	Új fájl megnyitása írásra. Ha a fájl már létezik, tartalma elvész!
"a" vagy **"at"**	**"ab"**	Fájl megnyitása a végéhez való hozzáírásra. Ha a fájl nem létezik, akkor létrejön.
"r+" vagy **"rt+"**	**"rb+"**	Létező fájl megnyitása írásra és olvasásra.
"w+" vagy **"wt+"**	**"wb+"**	Új fájl megnyitása írásra és olvasásra. Ha a fájl már létezik, tartalma elvész!
"a+" vagy **"at+"**	**"ab+"**	Fájl megnyitása a fájl végén végzett írásra és olvasásra. Ha a fájl nem létezik, akkor létrejön.

3. Az állomány tartalmának feldolgozása

Adatok **írására** és **olvasására** használható függvények áttekintése:

	Szöveges fájl **Olvasás**	Szöveges fájl **Írás**	Bináris fájl **Olvasás**	Bináris fájl **Írás**
Karakter (szöveges fájl) Bájt (bináris fájl)	**fgetc()**	**fputc()**	**fgetc()**	**fputc()**
Karakterlánc (szöveges fájl) Blokk (bináris fájl)	**fgets()**	**fputs()**	**fread()**	**fwrite()**
Formázott (szöveges fájl)	**fscanf()**	**fprintf()**	**-**	**-**

Karakterek írása szöveges fájlba

int fputc(int char, FILE *stream)

https://www.tutorialspoint.com/c_standard_library/c_function_fputc.htm

```c
#include <stdlib.h>
#include <stdio.h>

FILE *fp;  // fájlmutato

int main() {
    fp = fopen("abece.txt", "w");   // szoveges fajl megnyitasa irasra
    if (fp == NULL) {
        printf("Hiba a fajl megnyitasakor!");
        exit(0);
    }
    for (int i=0; i<26; i++) {
        char ch = 'a' + i;
        fputc(ch, fp);   // egy 'a', 'b', ... , 'z' karakterek fajlba irasa
    }
    fputc('\n', fp);   // '\n' sorvege karakter fajlba irasa
    for (int i=0; i<26; i++) {
        char ch = 'A' + i;
        fputc(ch, fp);   // egy 'A', 'B', ... , 'Z' karakterek fajlba irasa
    }
    fclose(fp);   // fajl bezarasa
}
```

abece.txt

```
abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

Karakterek olvasása szöveges fájlból

int fgetc(FILE *stream)

https://www.tutorialspoint.com/c_standard_library/c_function_fgetc.htm

```c
#include <stdlib.h>
#include <stdio.h>

FILE *fp;  // fájlmutato

int main() {
    fp = fopen("abece.txt", "r");   // szoveges fajl megnyitasa olvasasra
    if (fp == NULL) {
        printf("Hiba a fajl megnyitasakor!");
        exit(0);
    }
    char ch;
    while ((ch = fgetc(fp)) != EOF) {   // olvasas karakterenkent,
                                        // amig nincs vege a fajlnak
                                        // EOF = End Of File
        printf("%c", ch);   // karakter kiirasa a kepernyore
    }
    fclose(fp);   // fajl bezarasa
}
```

```
abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

Karaktereklánc írása szöveges fájlba

int fputs(const char *str, FILE *stream)

https://www.tutorialspoint.com/c_standard_library/c_function_fputs.htm

```c
#include <stdlib.h>
#include <stdio.h>

FILE *fp;  // fájlmutato

int main() {
    fp = fopen("szoveg.txt", "w");   // szoveges fajl megnyitasa irasra
    if (fp == NULL) {
        printf("Hiba a fajl megnyitasakor!");
        exit(0);
    }
    fputs("Ez egy sor...\n", fp);      // karakterlanc fajlba irasa
    fputs("Egy masik sor...\n", fp);   // karakterlanc fajlba irasa
    fclose(fp);   // fajl bezarasa
}
```

szoveg.txt

```
Ez egy sor...
Egy masik sor...
```

Karaktereklánc olvasása szöveges fájlból

char *fgets(char *str, int n, FILE *stream)

https://www.tutorialspoint.com/c_standard_library/c_function_fgets.htm

```c
#include <stdlib.h>
#include <stdio.h>

FILE *fp;  // fájlmutato

int main() {
    fp = fopen("szoveg.txt", "r");   // szoveges fajl megnyitasa olvasasra
    if (fp == NULL) {
        printf("Hiba a fajl megnyitasakor!");
        exit(0);
    }
    char s[200];
    while (fgets(s, sizeof(s), fp) != NULL) {   // karakterlanc olvasasa fajlbol,
                                                // amig nem ad vissza NULL erteket
                                                // (amig nincs mar mit kiolvasni)
        printf("%s", s);   // karakterlanc kiirasa a kepernyore
    }
    fclose(fp);   // fajl bezarasa
}
```

```
Ez egy sor...
Egy masik sor...
```

Formázott írás szöveges fájlba

int fprintf(FILE *stream, const char *format, ...)

https://www.tutorialspoint.com/c_standard_library/c_function_fprintf.htm

```c
#include <stdlib.h>
#include <stdio.h>

FILE *fp;  // fájlmutato

int a[3][5] = { {10, 15, 18,  4, 19},
                { 7,  9, 30, 24, 91},
                {67, 50,  3, 74,  8} };

int main() {
    fp = fopen("matrix.txt", "w");   // szoveges fajl megnyitasa irasra
    if (fp == NULL) {
        printf("Hiba a fajl megnyitasakor!");
        exit(0);
    }
    for (int i=0; i<3; i++) {
        for (int j=0; j<5; j++) {
            fprintf(fp, "%3d", a[i][j]);   // matrix elemeinek fajlba irasa
        }
        fprintf(fp, "\n");   // sortores fajlba irasa
    }
    fclose(fp);   // fajl bezarasa
}
```

matrix.txt

```
 10  15  18   4  19
  7   9  30  24  91
 67  50   3  74   8
```

Formázott olvasás szöveges fájlból

int fscanf(FILE *stream, const char *format, ...)

https://www.tutorialspoint.com/c_standard_library/c_function_fscanf.htm

```c
#include <stdlib.h>
#include <stdio.h>

FILE *fp;  // fájlmutato

int a[3][5];

int main() {
    fp = fopen("matrix.txt", "r");   // szoveges fajl megnyitasa olvasasra
    if (fp == NULL) {
        printf("Hiba a fajl megnyitasakor!");
        exit(0);
    }
    for (int i=0; i<3; i++) {
        for (int j=0; j<5; j++) {
            fscanf(fp, "%d", &a[i][j]);   // matrix elemeinek olvasasa fajlbol
            printf("%3d", a[i][j]);   // matrix elemeinek kiirasa a kepernyore
        }
        printf("\n");   // sortores kiirasa a kepernyore
    }
    fclose(fp);   // fajl bezarasa
}
```

```
 10  15  18   4  19
  7   9  30  24  91
 67  50   3  74   8
```

Bájtok írása bináris fájlba

int fputc(int char, FILE *stream)

https://www.tutorialspoint.com/c_standard_library/c_function_fputc.htm

```c
#include <stdlib.h>
#include <stdio.h>

FILE *fp;  // fájlmutato

int a[3][5] = { {10, 15, 18,  4, 19},
                { 7,  9, 30, 24, 91},
                {67, 50,  3, 74,  8} };  // 0-255 kozotti ertekek (byte) !!!

int main() {
    fp = fopen("matrix.dat", "wb");   // binaris fajl megnyitasa irasra
    if (fp == NULL) {
        printf("Hiba a fajl megnyitasakor!");
        exit(0);
    }
    for (int i=0; i<3; i++) {
        for (int j=0; j<5; j++) {
            fputc((char)a[i][j], fp);   // matrix elemeinek fajlba irasa
        }
    }
    fclose(fp);   // fajl bezarasa
}
```

matrix.dat

.....[C2.J.

Bájtok olvasása bináris fájlból

int fgetc(FILE *stream)

https://www.tutorialspoint.com/c_standard_library/c_function_fgetc.htm

```c
#include <stdlib.h>
#include <stdio.h>

FILE *fp;  // fájlmutato

int a[3][5];

int main() {
    fp = fopen("matrix.dat", "rb");   // binaris fajl megnyitasa olvasasra
    if (fp == NULL) {
        printf("Hiba a fajl megnyitasakor!");
        exit(0);
    }
    for (int i=0; i<3; i++) {
        for (int j=0; j<5; j++) {
            a[i][j] = fgetc(fp);   // matrix elemeinek olvasasa fajlbol
            printf("%3d", a[i][j]);
        }
        printf("\n");
    }
    fclose(fp);   // fajl bezarasa
}
```

```
 10  15  18   4  19
  7   9  30  24  91
 67  50   3  74   8
```

Blokkok írása bináris fájlba

size_t fwrite(const void *ptr, size_t size, size_t nmemb, FILE *stream)

https://www.tutorialspoint.com/c_standard_library/c_function_fwrite.htm

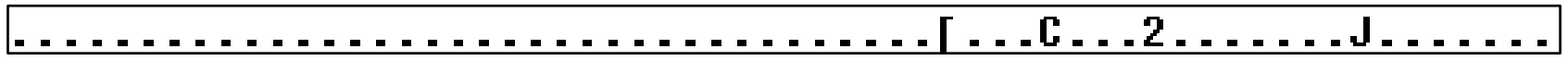
```c
#include <stdlib.h>
#include <stdio.h>

FILE *fp;  // fájlmutato

int a[3][5] = { {10, 15, 18,  4, 19},
                { 7,  9, 30, 24, 91},
                {67, 50,  3, 74,  8} };

int main() {
    fp = fopen("matrix.bin", "wb");   // binaris fajl megnyitasa irasra
    if (fp == NULL) {
        printf("Hiba a fajl megnyitasakor!");
        exit(0);
    }
    fwrite(a, sizeof(int), 3*5, fp);   // matrix fajlba irasa
    fclose(fp);   // fajl bezarasa
}
```

matrix.bin



Blokkok olvasása bináris fájlból

size_t fread(void *ptr, size_t size, size_t nmemb, FILE *stream);

https://www.tutorialspoint.com/c_standard_library/c_function_fread.htm

```c
#include <stdlib.h>
#include <stdio.h>

FILE *fp;  // fájlmutato

int a[3][5];

int main() {
    fp = fopen("matrix.bin", "rb");   // binaris fajl megnyitasa olvasasra
    if (fp == NULL) {
        printf("Hiba a fajl megnyitasakor!");
        exit(0);
    }
    fread(a, sizeof(int), 3*5, fp);   // matrix beolvasasa fajlbol
    fclose(fp);   // fajl bezarasa
    for (int i=0; i<3; i++) {
        for (int j=0; j<5; j++) {
            printf("%3d", a[i][j]);   // matrix eleminek kiirasa a kepernyore
        }
        printf("\n");   // sortores kiirasa a kepernyore
    }
}
```

```
 10  15  18   4  19
  7   9  30  24  91
 67  50   3  74   8
```

A **fájlpozíció átállítására** szolgáló függvény:

int fseek(FILE *stream, long int offset, int whence);

- Az első paraméter (**stream**) a fájlmutató.
- A második paraméterrel (**offset**) adható meg az elmozgatás mértéke bájtokban.
- A harmadik paraméter (**whence**) adja meg, hogy az elmozgatást (**offset**-et) mihez adjuk hozzá. A **whence** paraméter értéke az alábbi konstansok valamelyike lehet:
  - **SEEK_SET** - a fájl eleje,
  - **SEEK_CUR** - a fájlmutató aktuális pozíciója,
  - **SEEK_END** - a fájl vége.

Pozícionálással kapcsolatos további függvények:

- **long ftell(FILE *stream)** - fájlpozíció lekérdezése (bináris fájlok esetén addigi bájtszám),
- **void rewind(FILE *stream)** – a fájlpozíciót a fájl elejére állítja.

Egyéb fájlkezelő függvények:

- **int feof(FILE *stream)** - a fájl végének lekérdezése (fájl végén vagyunk-e),
- **int ferror(FILE *stream)** - hiba lekérdezése (történt-e fájl írási vagy olvasási hiba),
- **int fflush(FILE *stream)** - a puffer ürítése írás/olvasás esetén (tehát pl. írásnál azonnal kiírja az adatokat a fájlba).

Ha a fájlt írásra és olvasásra is megnyitottuk (pl. "r+", "w+", "a+" módban), akkor tudunk írni és olvasni is az állományból. Fontos azonban, hogy nem szabad közvetlenül írni olvasás után (vagy olvasni írás után) az **fseek()** vagy **fflush()** használata nélkül!

4. Az állomány bezárása

Az állományt az **fclose()** függvény segítségével tudjuk bezárni:

int fclose(FILE *stream);

Köszönöm a figyelmet!