◆ Ce este `fseek` în C?

`fseek` este o funcție din biblioteca standard C (`stdio.h`) folosită pentru a **modifica poziția cursorului** (pointerul de fișier) într-un fișier deschis. Cu alte cuvinte, îți permite să „te muți” într-o anumită poziție din fișier pentru a citi sau scrie date de acolo.

◆ Sintaxa funcției

```
int fseek(FILE *stream, long offset, int origin);
```

◈ Parametri:

- `stream`: pointerul către fișierul deschis (ex: `f` în `FILE *f = fopen("fisier.txt", "r");`).
- `offset`: deplasarea în octeți față de poziția specificată de `origin`.
- `origin`: punctul de referință de unde începe deplasarea. Poate fi:
  - `SEEK_SET` – începutul fișierului.
  - `SEEK_CUR` – poziția curentă a cursorului.
  - `SEEK_END` – sfârșitul fișierului.

◆ Valoarea returnată

- Returnează `0` dacă deplasarea s-a efectuat cu succes.
- Returnează `-1` în caz de eroare (ex: fișierul nu permite deplasarea, offset invalid etc.).

◆ Exemple de utilizare

◈ Exemplul 1: Deplasare la un punct specific în fișier

```
#include <stdio.h>
int main() {
    FILE *f = fopen("exemplu.txt", "r");
    if (f == NULL) {
        perror("Eroare la deschiderea fișierului");
        return 1;
    }

    fseek(f, 10, SEEK_SET);  // Mută cursorul la al 11-lea octet (se începe de la 0)

    int c = fgetc(f);        // Citește caracterul de la acea poziție
    printf("Caracterul este: %c\n", c);

    fclose(f);
    return 0;
}
```

◈ Exemplul 2: Citirea de la finalul fișierului

```
fseek(f, -5, SEEK_END);  // Se mută la 5 octeți înainte de sfârșitul fișierului
```

◈ Exemplul 3: Salt înainte în fișier (relativ la poziția curentă)

```
fseek(f, 20, SEEK_CUR);  // Se mută cu 20 de octeți mai departe față de poziția actuală
```

◆ Alte funcții utile împreună cu `fseek`

✓ `ftell(FILE *stream)`

Returnează poziția curentă a cursorului în fișier (în octeți).

```
long poz = ftell(f);
printf("Cursorul este la poziția: %ld\n", poz);
```

✓ `rewind(FILE *stream)`

Echivalent cu `fseek(f, 0, SEEK_SET)`. Mută cursorul la începutul fișierului.

◆ Când se folosește `fseek`?

- Când vrei să sari peste anumite date într-un fișier.
- Când citești date binare și vrei acces direct la o anumită înregistrare.
- În fișiere mari, pentru a evita parcurgerea secvențială lentă.
- În combinație cu `fwrite`, `fread`, `fgetc`, `fputc` pentru procesare fișiere.

◆ Atenție la:

- Nu folosi `fseek` în mod normal pe fișiere text pe unele platforme (ex: Windows), deoarece caracterele `\r\n` pot complica poziționarea exactă în octeți.
- Folosește `fseek` mai ales pentru fișiere binare (`rb`, `wb`, `rb+`, etc.).
- Dacă folosești `fseek` pe fișiere deschise în mod text, folosește doar `SEEK_SET` cu `0`, adică pentru revenirea la început.

◆ Exemplu 1:

1. Să salvăm mai mulți studenți într-un fișier binar.
2. Să citim un student aflat la o poziție anume (de exemplu, al treilea student din fișier) folosind `fseek`.

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

typedef struct {
    int id;
    char nume[30];
    float medie;
} Student;
```

```
void scrieStudenti(const char *numeFisier) {
    FILE *f = fopen(numeFisier, "wb");
    if (!f) {
        perror("Nu se poate crea fișierul");
        exit(1);
    }

    Student studenti[3] = {
        {1, "Ana Popescu", 9.50},
        {2, "Ion Ionescu", 8.75},
        {3, "Maria Georgescu", 10.00}
    };

    fwrite(studenti, sizeof(Student), 3, f);
    fclose(f);
}
```

```
void citesteStudentPozitia(const char *numeFisier, int pozitie) {
    FILE *f = fopen(numeFisier, "rb");
    if (!f) {
        perror("Nu se poate deschide fișierul");
        exit(1);
    }

    // Mută cursorul la înregistrarea dorită
    fseek(f, (pozitie - 1) * sizeof(Student), SEEK_SET);

    Student s;
    fread(&s, sizeof(Student), 1, f);

    printf("Studentul de la poziția %d:\n", pozitie);
    printf("ID: %d\n", s.id);
    printf("Nume: %s\n", s.nume);
    printf("Medie: %.2f\n", s.medie);
```

```
    fclose(f);
}
```

```
int main() {
    const char *numeFisier = "studenti.dat";

    scrieStudenti(numeFisier);
    citesteStudentPozitia(numeFisier, 3); // citim al 3-lea student

    return 0;
}
```

🔍 Explicație:

- `fwrite(studenti, sizeof(Student), 3, f);` → scrie 3 structuri `Student` în fișier.
- `fseek(f, (pozitie - 1) * sizeof(Student), SEEK_SET);` → mută cursorul direct la studentul dorit.
- `fread(&s, sizeof(Student), 1, f);` → citește exact acea structură.

◆ Exemplu 2:

- Să modificăm informațiile studentului de la o anumită poziție (ex: poziția 2).
- Să înlocuim vechile date cu unele noi.

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

typedef struct {
    int id;
    char nume[30];
    float medie;
} Student;
```

```
void scrieStudenti(const char *numeFisier) {
    FILE *f = fopen(numeFisier, "wb");
    if (!f) {
        perror("Nu se poate crea fișierul");
        exit(1);
    }

    Student studenti[3] = {
        {1, "Ana Popescu", 9.50},
        {2, "Ion Ionescu", 8.75},
        {3, "Maria Georgescu", 10.00}
    };

    fwrite(studenti, sizeof(Student), 3, f);
    fclose(f);
}
```

```
void citesteStudentPozitia(const char *numeFisier, int pozitie) {
    FILE *f = fopen(numeFisier, "rb");
    if (!f) {
        perror("Nu se poate deschide fișierul");
        exit(1);
    }

    fseek(f, (pozitie - 1) * sizeof(Student), SEEK_SET);

    Student s;
    fread(&s, sizeof(Student), 1, f);

    printf("Studentul de la poziția %d:\n", pozitie);
```

```
    printf("ID: %d\n", s.id);
    printf("Nume: %s\n", s.nume);
    printf("Medie: %.2f\n", s.medie);

    fclose(f);
}
```

```
void modificaStudentLaPozitie(const char *numeFisier, int pozitie, Student nou) {
    FILE *f = fopen(numeFisier, "rb+");  // rb+ permite citire și scriere
    if (!f) {
        perror("Nu se poate deschide fișierul pentru modificare");
        exit(1);
    }

    fseek(f, (pozitie - 1) * sizeof(Student), SEEK_SET);
    fwrite(&nou, sizeof(Student), 1, f);

    fclose(f);
}
```

```
int main() {
    const char *numeFisier = "studenti.dat";

    scrieStudenti(numeFisier);
    citesteStudentPozitia(numeFisier, 2);  // Înainte de modificare

    Student studentNou = {2, "Ionescu Vlad", 9.90};
    modificaStudentLaPozitie(numeFisier, 2, studentNou);

    printf("\nDupă modificare:\n");
    citesteStudentPozitia(numeFisier, 2);  // După modificare

    return 0;
}
```

🔍 Ce e nou?

- `fopen(numeFisier, "rb+")`: deschide fișierul pentru **citire și scriere** fără a-l rescrie.
- `fseek` duce cursorul exact unde este structura studentului.
- `fwrite` o suprascrie.

◆ Citire date student din consolă:

O versiune în care utilizatorul introduce din consolă datele studentului nou (ID, nume și medie), iar programul îl va **înlocui** pe studentul aflat la o poziție aleasă tot de utilizator.

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

typedef struct {
    int id;
    char nume[30];
    float medie;
} Student;

void scrieStudenti(const char *numeFisier) {
    FILE *f = fopen(numeFisier, "wb");
    if (!f) {
        perror("Nu se poate crea fișierul");
        exit(1);
    }
```

```
    Student studenti[3] = {
        {1, "Ana Popescu", 9.50},
        {2, "Ion Ionescu", 8.75},
        {3, "Maria Georgescu", 10.00}
    };

    fwrite(studenti, sizeof(Student), 3, f);
    fclose(f);
}

void citesteStudentPozitia(const char *numeFisier, int pozitie) {
    FILE *f = fopen(numeFisier, "rb");
    if (!f) {
        perror("Nu se poate deschide fișierul");
        exit(1);
    }

    fseek(f, (pozitie - 1) * sizeof(Student), SEEK_SET);

    Student s;
    fread(&s, sizeof(Student), 1, f);

    printf("Studentul de la poziția %d:\n", pozitie);
    printf("ID: %d\n", s.id);
    printf("Nume: %s\n", s.nume);
    printf("Medie: %.2f\n", s.medie);

    fclose(f);
}

void modificaStudentLaPozitie(const char *numeFisier, int pozitie) {
    FILE *f = fopen(numeFisier, "rb+");
    if (!f) {
        perror("Nu se poate deschide fișierul pentru modificare");
        exit(1);
    }

    Student nou;
    printf("\nIntrodu noile date pentru studentul de la poziția %d:\n", pozitie);
    printf("ID: ");
    scanf("%d", &nou.id);
    printf("Nume: ");
    getchar(); // curăță newline din buffer
    fgets(nou.nume, sizeof(nou.nume), stdin);
    nou.nume[strcspn(nou.nume, "\n")] = '\0'; // elimină '\n' de la final
    printf("Medie: ");
    scanf("%f", &nou.medie);

    fseek(f, (pozitie - 1) * sizeof(Student), SEEK_SET);
    fwrite(&nou, sizeof(Student), 1, f);

    fclose(f);
}

int main() {
    const char *numeFisier = "studenti.dat";
    scrieStudenti(numeFisier);

    int poz;
    printf("Ce poziție dorești să modifici (1-3)? ");
    scanf("%d", &poz);

    printf("\n--- Înainte de modificare ---\n");
    citesteStudentPozitia(numeFisier, poz);

    modificaStudentLaPozitie(numeFisier, poz);
```

```
    printf("\n--- După modificare ---\n");
    citesteStudentPozitia(numeFisier, poz);

    return 0;
}
```

◆ Ce face acest program:

1. Creează 3 studenți în fișierul binar.
2. Întreabă utilizatorul ce poziție vrea să modifice.
3. Afișează datele studentului de la acea poziție.
4. Permite introducerea unui nou student din consolă.
5. Suprascrie structura respectivă.
6. Afișează din nou studentul de la acea poziție.

◆ Explicație:

```
nou.nume[strcspn(nou.nume, "\n")] = '\0';
```

Această linie este folosită **după ce citim un șir de caractere cu `fgets`**, pentru a **elimina caracterul newline `\n`** pe care `fgets` îl introduce automat la finalul șirului, dacă încape în buffer.

◆ Funcția `strcspn`

```
size_t strcspn(const char *str, const char *reject);
```

Această funcție **caută prima apariție** a oricărui caracter din șirul `reject` în șirul `str` și returnează **poziția indexului** acelui caracter. **În cazul nostru:** `strcspn(nou.nume, "\n")` → caută poziția primului `\n` (newline) în șirul `nou.nume`.
Dacă, de exemplu:

```
nou.nume = "Maria Gutu\n"
```

atunci:

```
strcspn(nou.nume, "\n") // returnează 11 (poziția lui '\n')
```

◆ Ce face linia completă:

```
nou.nume[strcspn(nou.nume, "\n")] = '\0';
```

Înlocuiește caracterul `\n` (dacă există) cu `\0`, adică cu terminatorul de șir (nul). Astfel, șirul devine curat, fără newline la final.

◆ Exemplu 3:

- Să adăugăm o nouă funcție: `adaugaStudentLaFinal`
- Să adăugăm opțiune în `main` pentru alegerea acțiunii (modificare sau adăugare)

◆ Cod complet cu meniu:

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

typedef struct {
    int id;
    char nume[30];
    float medie;
} Student;

void scrieStudenti(const char *numeFisier) {
    FILE *f = fopen(numeFisier, "wb");
    if (!f) {
        perror("Nu se poate crea fișierul");
        exit(1);
    }
```

```
    Student studenti[3] = {
        {1, "Ana Popescu", 9.50},
        {2, "Ion Ionescu", 8.75},
        {3, "Maria Georgescu", 10.00}
    };

    fwrite(studenti, sizeof(Student), 3, f);
    fclose(f);
}

void citesteStudentPozitia(const char *numeFisier, int pozitie) {
    FILE *f = fopen(numeFisier, "rb");
    if (!f) {
        perror("Nu se poate deschide fișierul");
        exit(1);
    }

    fseek(f, (pozitie - 1) * sizeof(Student), SEEK_SET);

    Student s;
    fread(&s, sizeof(Student), 1, f);

    printf("Studentul de la poziția %d:\n", pozitie);
    printf("ID: %d\n", s.id);
    printf("Nume: %s\n", s.nume);
    printf("Medie: %.2f\n", s.medie);

    fclose(f);
}

void modificaStudentLaPozitie(const char *numeFisier, int pozitie) {
    FILE *f = fopen(numeFisier, "rb+");
    if (!f) {
        perror("Nu se poate deschide fișierul pentru modificare");
        exit(1);
    }

    Student nou;
    printf("\nIntrodu noile date pentru studentul de la poziția %d:\n", pozitie);
    printf("ID: ");
    scanf("%d", &nou.id);
    printf("Nume: ");
    getchar();
    fgets(nou.nume, sizeof(nou.nume), stdin);
    nou.nume[strcspn(nou.nume, "\n")] = '\0';
    printf("Medie: ");
    scanf("%f", &nou.medie);

    fseek(f, (pozitie - 1) * sizeof(Student), SEEK_SET);
    fwrite(&nou, sizeof(Student), 1, f);

    fclose(f);
}
```

```
void adaugaStudentLaFinal(const char *numeFisier) {
    FILE *f = fopen(numeFisier, "ab"); // "ab" = append binar
    if (!f) {
        perror("Nu se poate deschide fișierul pentru adăugare");
        exit(1);
    }

    Student nou;
    printf("\nIntrodu datele pentru studentul nou:\n");
    printf("ID: ");
    scanf("%d", &nou.id);
    printf("Nume: ");
    getchar();
```

```
    fgets(nou.nume, sizeof(nou.nume), stdin);
    nou.nume[strcspn(nou.nume, "\n")] = '\0';
    printf("Medie: ");
    scanf("%f", &nou.medie);

    fwrite(&nou, sizeof(Student), 1, f);
    fclose(f);

    printf("Student adăugat cu succes la finalul fișierului!\n");
}
```

```
int main() {
    const char *numeFisier = "studenti.dat";
    scrieStudenti(numeFisier);  // doar la prima rulare, poți comenta apoi!

    int optiune;
    printf("Alege opțiunea:\n");
    printf("1. Modifică un student existent\n");
    printf("2. Adaugă un student nou la final\n");
    printf("Opțiune: ");
    scanf("%d", &optiune);

    if (optiune == 1) {
        int poz;
        printf("Ce poziție dorești să modifici (1, 2, 3...)? ");
        scanf("%d", &poz);
        printf("\n--- Înainte de modificare ---\n");
        citesteStudentPozitia(numeFisier, poz);
        modificaStudentLaPozitie(numeFisier, poz);
        printf("\n--- După modificare ---\n");
        citesteStudentPozitia(numeFisier, poz);
    } else if (optiune == 2) {
        adaugaStudentLaFinal(numeFisier);
    } else {
        printf("Opțiune invalidă.\n");
    }

    return 0;
}
```

◆ Ce face noua funcție `adaugaStudentLaFinal`:

- Deschide fișierul cu mod `"ab"` (append binary).
- Cere de la utilizator datele studentului nou.
- Scrie acel student **la finalul fișierului**, fără a modifica nimic deja existent.

◆ Funcția `afiseazaTotiStudentii`

```
void afiseazaTotiStudentii(const char *numeFisier) {
    FILE *f = fopen(numeFisier, "rb");
    if (!f) {
        perror("Nu se poate deschide fișierul pentru citire");
        exit(1);
    }

    Student s;
    int index = 1;

    printf("\n--- Lista tuturor studenților ---\n");
    while (fread(&s, sizeof(Student), 1, f) == 1) {
        printf("Poziția %d:\n", index++);
        printf("  ID: %d\n", s.id);
        printf("  Nume: %s\n", s.nume);
        printf("  Medie: %.2f\n\n", s.medie);
    }
```

```
    fclose(f);
}
```