

Cum citim de la intrare

– un *cuvânt* (orice până la spațiu alb)

```
char s[80]; scanf("%79s", s);
```

spațiu alb = spațiu sau `\f \n \r \t \v`

ignoră spații albe inițiale; adaugă `'\0'` la sfârșit

Atenție! Nu se poate citi o linie de text în acest fel !

din Ana are mere va citi doar primul cuvânt: Ana

– o *linie de text*, până la `'\n'`

```
char s[80]; fgets(s, 80, stdin);
```

citește max. 80-1 caractere, inclusiv `'\n'`, adaugă `'\0'` la sfârșit

`stdin`: identificator definit în `stdio.h` pt. fișierul standard de intrare

Cum testăm sfârșitul de fișier

1. În orice punct de program: `int feof(FILE *fp)`
returnează nenul (adevărat) dacă s-a atins sfârșitul lui `fp`; 0 dacă nu
pentru fișierul de intrare: `feof(stdin)`

2. După valoarea returnată de funcțiile de intrare:

```
int c; c = getchar(); if (c == EOF) /* ... */
```

Atenție! `c` trebuie declarat `int` pentru a testa de EOF

valoarea EOF (-1) e diferită de cea a oricărui caracter (0 .. 255)

`scanf` returnează EOF (-1) dacă întâlnește imediat sfârșitul de fișier
(nu și dacă a reușit să citească măcar ceva)

⇒ Folosiți doar `if (scanf(...)) == nr_variabile_dorite` pentru a testa
citire corectă, nu doar `if ((scanf(...)))` (pentru că și EOF e nenul)

`fgets` returnează NULL dacă fișierul se termină înainte de a citi ceva

Exemplu: prelucrarea unui fișier linie cu linie

```
char lin[128]; while (fgets(lin, 128, stdin)) /* prelucrează lin */
```

Citirea anumitor categorii de caractere

Atenție: Funcțiile din `ctype.h` returnează *nenu* pentru un caracter de felul dorit și 0 în caz contrar (*NU* neapărat 1 și 0)
nu scrieți niciodată `if (isalpha(c) == 1)` ci doar `if (isalpha(c))`

Atenție la cicluri infinite pentru sfârșit de fișier:

```
int c; while (isdigit(c = getchar())) /* ceva */
```

va ieși din ciclu când `c` nu e cifră, inclusiv la EOF (nu e cifră)

```
int c; while (!isdigit(c = getchar())) /* ceva */
```

se va bloca la EOF, pentru că nu e cifră (nici `isalpha`, `isspace`, etc.)

```
while (!isdigit(c = getchar()))
```

```
    if (c == EOF) break; /* sau ce vrem să facem la EOF */
```

```
    else /* restul prelucrării */
```

Ignoră oricâte spații albe: `scanf(" ");`

Ignoră până la sfârșit de linie: `scanf("%*[^\\n]"); scanf("%*1[\\n]");`

(citește și ignoră `(*)` oricâte caractere diferite `(^)` de `\\n` și un `\\n`)

Citirea și prelucrarea până la sfârșit de fișier

Pentru a prelucra corect până la EOF nu e suficientă secvența din stânga deoarece `!feof()` la începutul ciclului nu garantează o citire corectă. Trebuie testată citirea corectă (`getchar() != EOF`, `fgets(...) != NULL`, valoarea lui `(f)scanf`), și tratat cazul de eroare (ex. ieșirea din buclă)

```
while !(feof(fisier)) {           for (;;) {
    citeste();                     if (citeste() != CORECT) break;
    prelucreaza();                 prelucrează();
}                                  }
```

Indicatorul de sfârșit de fișier e poziționat doar când se încearcă citirea *dincolo* de sfârșitul fișierului, nu când s-a citit ultimul caracter.

– după citirea ultimului element, `feof()` poate fi adevărat sau nu
ex. pt. un fișier de întregi separați prin spații, `feof()` e poziționat după citirea ultimului doar dacă nu e urmat de altceva (ex. spațiu, `\n`)
ex. la citirea linie cu linie, `feof()` e poziționat după citirea ultimei linii doar dacă ea nu se termină cu `\n`.

– dacă `feof()` e fals, fișierul poate să mai conțină un element sau nu

Problema de la laborator

Să se tipărească, pe câte o linie, toate secvențele de cifre din intrare.
O abordare: textul e o repetitie de: grup de cifre, grup de alte caractere
⇒ structura: două cicluri consecutive, într-un ciclu pana la EOF
'\n' se tipareste la trecerea intre cele doua (preferabil dupa cifre)
se incepe cu grupul de alte caractere (posibil vid)

```
void main(void)
{
    int c;
    do { /* si EOF e !isdigit, atentie la ciclu infinit! */
        while (!isdigit(c = getchar())) if (c == EOF) return;
        /* aici, c e sigur o cifra; repeta cat timp e cifra */
        do putchar(c); while (isdigit(c = getchar()));
        putchar('\n'); /* gata cifrele, c e altceva, poate EOF */
    } while (c != EOF);
}
```

Extragerea secvențelor de cifre (cont.)

Mai simplu: când vedem o cifră, citim și tipărim cât timp e cifră

```
void main(void) {  
    int c;  
    while ((c = getchar()) != EOF)  
        if (isdigit(c)) { /* prima cifra; continua cu restul */  
            do putchar(c); while (isdigit(c = getchar()));  
            putchar('\n'); /* gata grupul de cifre */  
        } /* daca nu e cifra, nu trebuie facut nimic */  
}
```

De considerat, pentru prelucrări pe secvențe de caractere:

- cum e definită secvența căutată, și ce poate fi între secvențe ?
- ce trebuie făcut la separarea între secvențe (cicluri în program) ?
- care e starea (caracterul curent) înainte și după fiecare ciclu ?
- EOF poate interveni oricând. Se oprește corect programul ?

Încercați să priviți problema (și soluția) ca un automat:

În orice punct din program, ce poate interveni ? ce trebuie făcut ?