

Prelucrări iterative

Marius Minea

28 martie 2006

Scrierea ciclurilor

În conceperea programelor care conțin cicluri:

- identificăm ce variabilă se modifică în fiecare iterație
- identificăm care e condiția de continuare / oprire
- nu uităm instrucțiunea care modifică acea variabilă (altfel ciclul se va continua la infinit)

La ieșirea dintr-un ciclu, condiția e falsă \Rightarrow aceasta ne spune ceva despre valorile posibile ale variabilelor din condiție.

Folosim această informație pentru a gândi mai departe programul.

Verificăm programul executând mental întâi un exemplu simplu și apoi la rulare, *teste* pentru diferitele situații posibile

Programarea calculatoarelor. Curs 5

Marius Minea

Ciclul cu test final

do *instrucțiune* while (*expresie*);

- uneori știm sigur că un ciclu trebuie executat cel puțin o dată
- ca și ciclul cu test inițial, execută *instrucțiunea* atât timp cât valoarea expresiei e nenulă (adevărată).

– expresia se evaluează însă *după* fiecare iterație

– echivalent cu:

```
    instrucțiune
    while ( expresie ) instrucțiune
```

Programarea calculatoarelor. Curs 5

Marius Minea

Instrucțiunea break

- produce ieșirea din corpul instrucțiunii while, do, for sau switch *imediat înconjurătoare*; execuția continuă cu instrucțiunea următoare
- mai convenabilă decât testarea unei variabile boolene la ciclul următor
- mai lizibil, dacă codul peste care se sare e complex

```
int c, wcnt = 0;
do { // elimina spatiile initiale, cel mult pana la '\n'
    do c = getchar(); while (!isspace(c) && c != '\n');
    if (c == '\n' || c == EOF) break; // gata linia
    wcnt = wcnt + 1; // incepe un nou cuvânt
    // citeste caractere noi cat timp nu sunt spatii albe sau EOF
    do c = getchar(); while (!isspace(c) && c != EOF);
    // daca am ajuns la sfarsit de linie sau EOF iesim
} while (c != '\n' && c != EOF);
```

Programarea calculatoarelor. Curs 5

Marius Minea

Exemplu: găsirea de factori primi

```
#include <stdio.h>
unsigned primfact(unsigned n, unsigned d) { // cel mai mic factor prim
    while (n % d != 0) { // incepand de la d
        d = d + 1;
        if (d > n/2) return n; // nu mai pot fi alti factori pana la n
    }
    return d; // cand iesim din while, sigur n % d == 0
}

void printfact(unsigned n) {
    unsigned d = 2; // primul numar prim
    while (n > 1) {
        d = primfact(n, d); // salvam valoarea pentru primul factor gasit
        printf("%u ", d); // si data viitoare continuam de acolo
        n = n / d;
    }
    putchar('\n');
}

int main(void) {
    printfact(108);
    return 0;
}
```

Programarea calculatoarelor. Curs 5

Marius Minea