Ciklusok

Kormányos Andor

Komplex Rendszerek Fizikája Tanszék

2023 szeptember 19.

Ciklusok

Ha a programnak többször meg kell ismételnie utasításokat, akkor ciklusokat írunk.

A C++ nyelvben háromféle ciklus van

- `for (...) { }` – iteratív ciklus
- `while() { }` – **elöl tesztelő ciklus**, zárójelben az ismétlés feltétele
  - a feltétel egy kifejezés, amíg ez teljesül, a `{ }` blokban levő utasítássorozat újra lefut
  - a feltétel a ciklusmag lefutása **előtt** értékelődik ki
- `do { } while ();` – **hátul tesztelő ciklus**
  - a feltétel a ciklusmag lefutása **után** értékelődik ki
  - a ciklusmag egyszer mindenképpen lefut
  - figyelem! a végére kell `;`

A `for` ciklus

- adott számú ismétlés esetén használjuk
- pl. sorban végigmegyünk egy vektor elemein és kiírjuk őket

Szintakszis:

```
for ( [init] ; [condition] ; [increment] ) {
  // do something
}
```

Zárójelen belüli tagok:

1. **init**: utasítás, ami az első iteráció előtt hajtódik végre
   - jellemzően a ciklusváltozó inicializálása
2. **condition**: kifejezés, minden iteráció előtt kiértékelődik
   - ha értéke $\neq 0$, akkor a ciklusmag lefut
   - elöltesztelős ciklus, mint a `while`
3. **increment**: utasítás, ami a ciklusmag legvégén hajtódik végre
   - tipikusan a ciklusváltozó növelésére, csökkentésére
   - csak akkor fut le, ha a feltétel teljesült, és a ciklusmag is lefutott

Néhány példa `for` ciklusra

- felfelé vagy lefelé is léptethetjük a ciklusváltozót:

```
1 for (int i = 0; i <=10 ; ++i) {
2   // do something
3 }
```

```
1 for (int i = 99; i >= 0; --i) {
2   // do something
3 }
```

Fun fact: C-ben általában post-increment-et használunk a ciklusváltozóra (i++), a C++ -ban pre-increment-et (++i)

- Ciklusok egymásba ágyazhatóak

```
1 for (int i = 0; i < 100; ++i) {
2   for (int j = 0; j < 20; ++j) {
3     // do something
4   }
5 }
```

`while` ciklus

- általában akkor használjuk, ha nem ismert, pontosan hányszor kell egy utasítást végrehajtani, csak egy feltétel adott
- pl olvasson be számokat addig, amíg a fájl végére nem ér (később részletesen)

Egyszerű példa:

```
1 double d;
2 while (cin >> d) {
3   std::cout << d*d << std::endl;
4 }
```

- sorban beolvassa a számokat, amíg meg nem szakítjuk a programot `End-of-file`-karakterrel
- End-of-file: `Ctrl-d` (Linux), `Ctrl-z+Return` (Windows)

`while` és `for` ciklus

- a `while` ciklust is tudjuk a `for` ciklushoz hasonlóan használni, de a `for` ciklus tömörebb

```
1  int i = 0;
2  while (i < 100) {
3    // do something
4    ++i;
5  }
```

Az ennek megfelelő `for` ciklus

```
1  for (int i = 0; i < 100; ++i) {
2    // do something
3  }
```

A `break` és a `continue` utasítás

A `break` utasítás kilép a (legbelső) ciklusból

- általában valamilyen feltétel teljesülése esetén
- nem iteráció végén, hanem speciális esetekben használjuk
- `for` ciklusnál is működik, de többnyire `while`-nál használják
- illetve a `switch` utasítás különböző eseteit is ezzel kell zárni

A `continue` utasítás átugorja a ciklusmag hátralevő részét

- ezt is feltétellel együtt használjuk
- a ciklus maga folytatódik, csak a ciklusmag nem fut le