Vezérlési szerkezetek

Változókkal műveleteket szeretnénk végezni. A C nyelvben erre a többi nyelvben megszokott módszerek adottak: ciklusok és elágazások.

Ciklusok

A C nyelv futószalagjából három lehetőségünk van. Mindhárom esetben a kerek zárójelekkel határolt rész ciklusmagnak, a kapcsos zárójelek közötti részt ciklustörzsnek nevezzük. Utasításokat mindkét részen elhelyezhetünk.

Definíció (szintaxis): ciklus (ciklusmag) { ciklustörzs }

Részletesen:

- For

  Ez a leggyakrabban használt. Leginkább ranged loop-okhoz használjuk őket, ami feladat a legáltalánosabb is: adott egy konténerünk (tömb, vektor, lista, stb) és az összes elemén vagy meghatározott részhalmazán szeretnénk végigiterálni.

  Szintaktikája nagyon flexibilis, a ciklusmag három alrészét szabadon használhatjuk.

  - Szintaktikája általában:

    ```
    for
    (
        ciklusváltozó;
        futási feltétel a ciklusváltozóra;
        ciklusváltozó módosítása
    )
    {
        utasítások
    }
    ```

    Megjegyzések:
    - A ciklusváltozó lehet külső változó is. Tipikusan, ha szeretnénk használni a cikluson kivül is, pl. keresésnél.
    - A ciklusváltozó módosítását a ciklustörzsben is megtehetjük.
    - A ciklus futását a törzsben is megszakíthatjuk *break* utasítással, tipikusan valamilyen elágazás törzsében.
- while
- do...while

  E két ciklus használata azonos, egy különbség van: a *while* a ciklus elején, a *do..while* a ciklus végén ellenőrzi, hogy a ciklusmag állítása igaz-e.

  ```
  while ( ciklusfeltétel )
  { utasítások }

  do
  { utasítások }
  while ( ciklusfeltétel )
  ```

  Megjegyzések:
  - Általában külső változóra elkövetett feltétel vizsgálatot végzünk.

- do..while gyakori felhasználásai: felhasználói input bekérése és ellenőrzése, amíg az input nem megfelelő.

Megjegyzés:
Tetszőleges, a ciklusmagban szereplő utasítás kiszervezhető. A ciklusváltozó definiálható a ciklus előtt is, illetve cikluson kivüli változó használható mint ciklusváltozó. A

Break:
A ciklustörzsben is megszakítható a ciklus futása a ***break*** utasítással. Többnyire elágazásokban alkalmazzuk, megfelelő feltételek teljesülésekor.

Végtelen ciklusok:
- for(;;) { *utasítások* }
- while (1) { *utasítások* }
- do { *utasítások* } while(1)
- ***Megjegyzés:*** A legtöbb multifunkciós program tartalmaz végtelen ciklust. Például az operációs rendszer is, böngészők, játékok.

Feladatok:

1. Írj programot, mely *for* ciklussal összeadja a számokat 0 és 10 között. Old meg ugyanezt *while* és *do..while* ciklussal is.

2. Írj programot, mely egy szám faktoriálisát számolja ki ciklus segítségével. A program olvasson be egy számot scanf-fel és ennek faktoriálisát írja ki. (A felhasználói hibákat most ne vegyük figyelembe.)

3. Írj programot, mely bekér két számot, az első kisebb mint a második (feltételezve, hogy a felhasználó valóban számokat ad meg és valóban növekvő sorrendben). Írjon ki a program egy táblázatot a két szám között 5-tel lépdelve, ahol az első oszlop a Celsius fokot jelenti, a második pedig ennek Fahrenheit fokát.

4. Írj programot, amely bekér egy számot és megkeresi annak legnagyobb osztóját.

Elágazások

Két elágazási módszer áll a rendelkezésünkre: az if-else if-else és a switch-case. Az első általános célú, a második *integrális* típusok esetén használható, elsősorban mint felsorolások megkülönböztetésére (pl. enumeration esetén).

Részletesen:
- if (feltétel 1)
  { *utasítások* }
  else if (feltétel 2) // opcionális
  { *utasítások* }
  else if (feltétel 3) // opcionális
  { *utasítások* }
  …
  else // opcionális

{ *utasítások* }

Megjegyzés:

- A kapocszárójelek elhagyhatóak, ha csak egy utasítást tartalmaz. Egymásba ágyazott elágazások esetén ez erősen nem ajánlott, más működést eredményezhet a zárójelek hiánya és megléte.

- switch (változó)
  {
  case 1:
  { *utasítások* }
  break;
  case 2:
  { *utasítások* }
  break;
  ….
  default:
  }

Megjegyzés:

- A switch-case természetesen átírható if-else if-else elágazásokra is.
- A break elhagyható, de akkor a következő case is vizsgálatra kerül, sőt, a default le fog futni. Bizonyos esetekben természetesen ez így tervezett működés.
- A kapocszárójelek elhagyhatóak a case-kben, kivéve, ha helyi változókat deklarálunk az adott case-ben.

Feladatok:

5. Írj programot, ami beolvas egy karaktert, majd megvizsgálja, hogy az szám-e. Amennyiben igen, írja ki a számot sikeres üzetenettel, ellenkező esetben hibaként írja ki.

6. Írj programot, amely beolvas egy egész számot, majd egy switch-case-ben kiértékeli azt: ha 0, akkor *“Hello World!”*-t ír, ha 1, akkor *“Good Bye!”*-t, minden más esetben *“What’s this input?”*-t. Próbáld ki *break* és *break* nélkül is.

A ciklusok és az elágazási szerkezetek tetszőleges mélységben, tetszőleges mennyiségben egymásba ágyazhatóak.

Feladatok:

10. Írj programot, ami bekér karaktereket újsor karakterig, de maximum 20-t és fordított sorrendben kiírja.

11. Írj programot, ami bekér karaktereket újsor karakterig, de maximum 20-t és megvizsgálja, hogy a beadott karakterlánc palindrom-e.