# 2.1.3 Vezérlési szerkezetek

* Egy nyelv vezérlésátadó utasításai az egyes műveletek végrehajtási sorrendjét határozzák meg.

### Utasítások és blokkok

* Egy olyan kifejezés, mint x = 0, i++ vagy printf(...) utasítássá válik, ha egy pontosvesszőt írunk utána. Pl:

x = 0;

i++;

printf(...);

* A {} kapcsos zárójelekkel deklarációk és utasítások csoportját fogjuk össze egyetlen *összetett utasításba* vagy *blokkba*, ami szintaktikailag egyenértékű egyetlen utasítással.
* A kapcsos zárójelek használatára jó példa a függvények utasításait összefogó zárójelpár, vagy az if, else, while, for, ill. hasonló utasítások utáni utasításokat összefogó zárójelpár.

### Az if-else utasítás

* Az if-else utasítást döntés kifejezésére használjuk. Formálisan az utasítás szintaxisa a következő:

if (kifejezés)

1. utasítás

else

2. utasítás

ahol az else rész opcionális. Az utasítás először kiértékeli a kifejezést, és ha ennek értéke igaz (azaz a kifejezés értéke nem nulla), akkor az 1. utasítást hajtja végre. Ha a kifejezés értéke hamis (azaz nulla) és van else rész, akkor a 2. utasítás hajtódik végre.

* Mivel az if-else szerkezet else ága opcionális egymásba ágyazott if-else szerkezeteknél, hiányzó else ágak esetén nem világos, hogy a meglévő else ág melyik if utasításhoz tartozik. Például az

if (n > 0)

if (a > b)

z = a;

else

z = b;

programrészletben az else a belső if utasításhoz tartozik, amit a programrész tagolása is mutat. Általános szabályként megfogalmazhatjuk, hogy az else mindig a hozzá legközelebb eső, else ág nélküli if utasításhoz tartozik. Ha nem így szeretnénk, akkor a kívánt összerendelés kapcsos zárójelekkel érhető el, mint pl. az

if (n > 0) {

if (a > b)

z = a;

}

else

z = b;

szerkezetben.

### Az else-if utasítás

* Az

if (kifejezés)

utasítás

else if (kifejezés)

utasítás

else if (kifejezés)

utasítás

else if (kifejezés)

utasítás

.

.

.

else

utasítás

szerkezet olyan gyakran fordul elő, hogy mindenképpen megérdemli a részletesebb elemzést. Ez a szerkezet adja a többszörös döntések (elágazások) programozásának egyik legáltalánosabb lehetőségét. A szerkezet úgy működik, hogy a gép sorra kiértékeli a *kifejezéseket* és ha bármelyik ezek közül igaz, akkor végrehajtja a megfelelő *utasítást*, majd befejezi az egész vizsgáló láncot. Itt is, mint bárhol hasonló esetben, az utasítás helyén kapcsos zárójelek között elhelyezett blokk is állhat.

* A szerkezet utolsó (if nélküli) else ága alapértelmezés szerint a „fentiek közül egyik sem” esetet kezeli. Néha ilyenkor semmit sem kell csinálni, ezért a szerkezetet záró

else

utasítás

ág hiányozhat, vagy valamilyen lehetetlen eset érzékelésével hibaellenőrzésre használható.

### A switch utasítás

* A switch utasítás is a többirányú programelágaztatás egyik eszköze. Az utasítás úgy működik, hogy összehasonlítja egy kifejezés értékét több egész értékű állandó kifejezés értékével, és az ennek megfelelő utasítást hajtja végre. A switch utasítás általános felépítése:

switch (kifejezés) {

case állandó kifejezés: utasítások

case állandó kifejezés: utasítások

.

.

.

default: utasítások

}

* Mindegyik case ágban egy egész állandó vagy állandó értékű kifejezés található, és ha ennek értéke megegyezik a switch utáni kifejezés értékével, akkor végrehajtódik a case ágban elhelyezett egy vagy több utasítás.
* Az utolsó, default ág akkor hajtódik végre, ha egyetlen case ághoz tartozó feltétel sem teljesült.
* A default ág opcionális, ha elhagyjuk és a case ágak egyike sem teljesül, akkor semmi sem történik.
* A case ágak és a default ág tetszőleges sorrendben követhetik egymást.