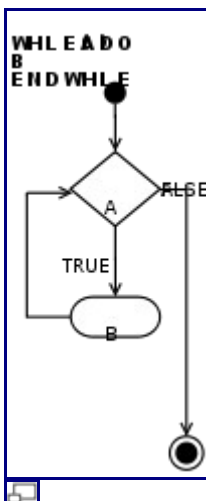


Változók fogalma és típusai

- Változók fogalma:
 - egy érték eltárolására, megjegyzésére szolgál (hely a memóriában),
 - szabadon módosítható.
- Változók típusa: meghatározza az eltárolható értéket
 - egész szám (1, 2, 4 bájtt): Byte, Integer, Long
 - valós szám (4, 8, 8 bájtos fixpontos): Single, Double, Currency
 - szöveg (max. 232 bájtt): String,
 - logikai változó (2 bájtt): - Boolean
 - dátum (8 bájtos): Date

Elöltesztelő ciklus [[szerkesztés](#)]



Az előltesztelő ciklus sémája:

Ismételd, amíg a feltétel igaz
utasítások
ciklus vége

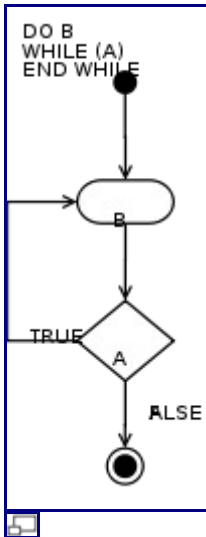
Az előltesztelő ciklus tehát először megvizsgálja, hogy a feltétel fennáll-e. Ha igen, akkor lefuttatja

a ciklusmagot, és újból kezdődik; ha nem, akkor a program a ciklus utáni ponton folytatódik, azaz a ciklusmag kimarad. Lehetséges tehát, hogy az előtesztelő ciklus egyszer sem fog lefutni.

Az előtesztelő ciklus tipikus példája az [adatállományok](#) beolvasása; előfordulhat ugyanis, hogy az állomány üres, és ilyenkor nincs mit beolvasni. Hasonló a helyzet a [könyvtárak](#) listázásakor is, hiszen a könyvtárban nem biztos, hogy van állomány.

Az előtesztelő ciklus tipikus kulcsszava a *while*.

Hátutesztelő ciklus [[szerkesztés](#)]



A hátutesztelő ciklus sémája:

```
Ismételd  
    utasítások  
amíg a feltétel igaz
```

Mivel a feltételvizsgálat a ciklusmag után áll, ezért a hátutesztelő ciklus legalább egyszer mindenképpen lefut.

A konkrét programnyelvi megvalósítástól függ, hogy a hátutesztelő ciklusban a folytatás vagy a kilépés feltételét kell-e megadni. A ciklus az első esetben addig fut, amíg a ciklusvégben megadott feltétel igaz (ennek tipikus kulcsszava a *while*), a másik esetben pedig addig, amíg igazzá nem válik (tipikus kulcsszava az *until*).

Jellemző példa a hátutesztelő ciklusra az adatbevitel ellenőrzése. Tegyük fel, hogy regisztrálni szeretnénk egy ingyenes [postafiókot](#) az [interneten](#). A [szerveren](#) futó programnak először ellenőriznie kell, hogy szabad-e még a választott azonosító, majd azt is, hogy kitöltöttük-e az összes kötelező adatmezőt, és addig kell ismételnie ezt a lépést, amíg az összes adat helyessé nem válik. Sok szolgáltató ezt egy lépésben végzi el, vagyis először ki kell töltenünk az űrlapot, aztán tudjuk meg, hogy szabad-e még az azonosító. Ebben az esetben a ciklus a következőképpen nézhet ki:

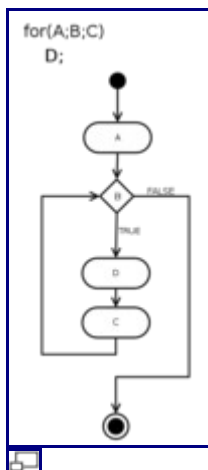
```
Ismételd  
    Olvasd be az adatokat  
    Ha az azonosító foglalt, írd ki, és add vissza az űrlapot  
    Ha egy kötelező mező hiányzik, írd ki, és add vissza az űrlapot  
amíg hibás az űrlap
```

Mivel a feltételt csak az adatbeolvasás után ellenőrzi a program, ezért az adatbeolvasás legalább egyszer mindenképpen megtörténik. Ha sikerült elsőre jól kitöltenünk az adatlapot, akkor tovább lehet lépni.

Zavart okozhat a szöveges leírásokban a magyar **amíg** szó két, egymással lényegében ellentétes

jelentése. Az algoritmusok leírásában az *amíg* szót mindig a 'mialatt', 'miközben' értelmében használjuk, ami az angol *while*-nak felel meg. A fenti eljárást úgy is fogalmazhatnánk, hogy „*olvasd be az adatokat, ahányszor kell, amíg jók lesznek*” – ebben az esetben az *amíg* szót az angol *until* értelmében használtuk, ami szintén lehet a hátultesztelő ciklus kulcsszava. Ezért előfordulhat, hogy a programkódban épp a szöveges algoritmusban látható feltétel tagadását kell megfogalmaznunk.

Számlálós (FOR) ciklus [[szerkesztés](#)]



A számláló ciklus általánosságban olyan ciklust jelent, amely egy felsorolható típus adott intervallumán léptet végig, speciálisan egész számokon. Üres intervallumra nem fut le.

C++-ban a forciklus teljesen ekvivalens az elől vagy hátultesztelős ciklussal, de más nyelvekben nem feltétlenül van ez így: a forciklus átírható más ciklusba, de visszafele már nem feltétlenül, a for ciklusnak kisebb lehet a kifejezőereje.

A ciklus fajtája	A futások száma	Legkevesebb hányszor fut le?
Feltételes Elöltesztelő	előre nem ismert	lehet, hogy egyszer sem
Hátultesztelő	előre nem ismert	legalább egyszer
Számlálós	előre ismert lehet,	hogyan egyszer sem

Az **értelmező**, **értelmező program** vagy angol kifejezéssel **interpreter** egy olyan program (ritkábban beépített hardver), ami képes arra, hogy az általa felismert nyelven megfogalmazott utasításokat bemenő adatként kezelje, és a futtató gép saját utasításkészletének megfelelő utasítások sorozatává alakítsa át, majd ezeket a utasítás sorozatokat **azonnal** futtassa is.

Míg egy fordítóprogram a forrásprogramokat utasításonként a futtató gép által végrehajtható (gépi kódú) utasítások sorozatává alakítja át – fordítja – azaz a forrásprogramból a futtatásra kész forma teljes egészében előáll, addig az értelmező a [forrásprogramot](#) anélkül is végre hajthatja – azonnal – hogy a teljes forrásprogramot beolvassa.

A **fordítóprogram** (angolul **compiler**) olyan [számítógépes program](#), amely valamely [programozási nyelven](#) írt programot képes egy másik programozási nyelvre lefordítani.