
Programozás alapjai

Bevezetés

Ficsor Lajos

Miskolci Egyetem

Általános Informatikai Tanszék

Tartalom

- A gépi kódú programozás és hátrányai
- A magas szintű programozási nyelv fogalma
- A programkészítés fázisai

A gépi kódú program

- A számítógép a működése során gépi utasításokat (instrukciókat) hajt végre egymás után
- Az instrukciók csak primitív műveleteket tudnak végrehajtani. Például:
 - regiszterekben tárolt egész számok összeadása
 - egy regiszter tartalmának adott memóriacímre másolása
 - stb.
- Az instrukciók gép-specifikusak, a processzorral együtt tervezik meg azokat

A gépi kódú programozás hátrányai

- Egy feladat megfogalmazása a gép nyelvén rendkívül aprólékos munka
- A gépi kódú programozáshoz a hardware elemek pontos ismerete szükséges
- Ha egy gép hardware összetétele megváltozik, a programot ennek megfelelően módosítani kell.
- Ha a programot egy másik processzorral épített gépen akarjuk futtatni, a programot teljesen újra kell írni, a **gépi kódú programok nem hordozhatók!**

A megoldás

- Az **operációs rendszer** (egyik szolgáltatásaként) elrejtí a hardware elemek kezelésének részleteit (pl. egy lemez tartalmát file-rendszerként kezelhetjük) - de ez egy másik tárgy témája.
- A programok írására **magasszintű programozási nyelv**et használunk, amely processzor és - többnyire - hardware független utasításokat tartalmaz.
- A programok hordozhatóak, egy másik processzorra változtatás nélkül áttehető a program.

A megoldás (folytatás)

- A programozási nyelv az emberi nyelvhez közelebb áll, mint a gépi nyelvhez - könnyebben megtanulható.
- A programozási nyelv utasításai bonyolultabb folyamatokat írnak le, mint az instrukciók - ugyanannak a problémának a leírása sokkal kevesebb utasítást igényel.
- A gyakran előforduló részfeladatokra előre elkészített programrészek (könyvtári rutinok) használhatók. ("**Library**")

A programozási nyelv

- A természetes nyelvhez hasonlít a felépítése
- Számos nyelv létezik, de mindegyiknek precíz definíciója van.
- Egy nyelv definíciója tartalmaz:
 - alapelemek leírását (szintaktikai egységek) - mint a természetes nyelvekben a szavak
 - formális szabályokat az alapelemek összeépítésére (szintaktika)- mint a nyelvekben a nyelvtan
 - a nyelvi szerkezetek jelentésének definícióit (szemantika) - mint a nyelvekben a használat szabályai

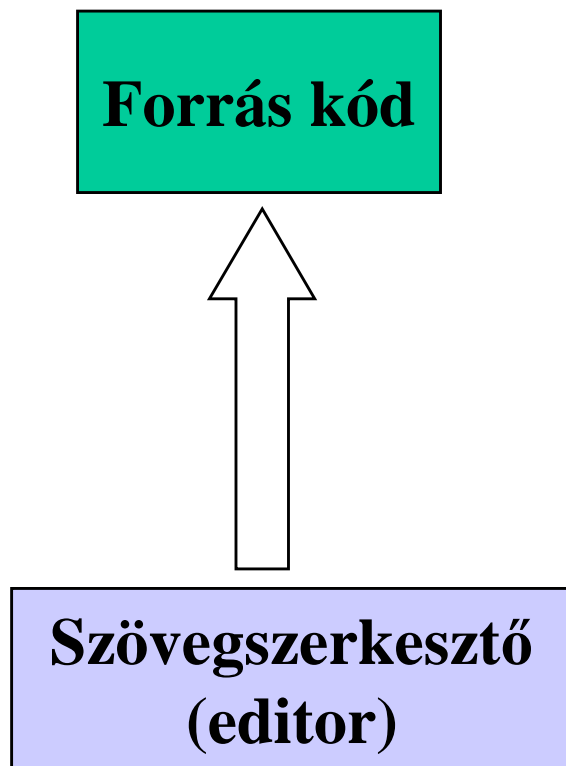
Megoldandó problémák

- Egyetlen processzor sem érti egyetlen magas szintű programozási nyelv utasításait sem!
- Megoldás: a programot le kell fordítani instrukciók sorozatára. A munka automatizálható:
fordítóprogram (compiler)
- Minden operációs rendszer - processzor párosra külön fordítóprogram kell. Ez azonban megoldja a gépfüggetlen programok írását.

Megoldandó problémák (folyt.)

- Egy programot általában több részben (modulban) írunk meg
- Felhasználunk előre elkészített modulokat is (könyvtári rutinokat)
 - Megoldás: ezeket össze kell építeni a **linker** (tárgykódú szerkesztő) program segítségével

A programkészítés menete 1.



Elkészítjük a program forráskódját

Ez egy (esetleg több) szövegfile

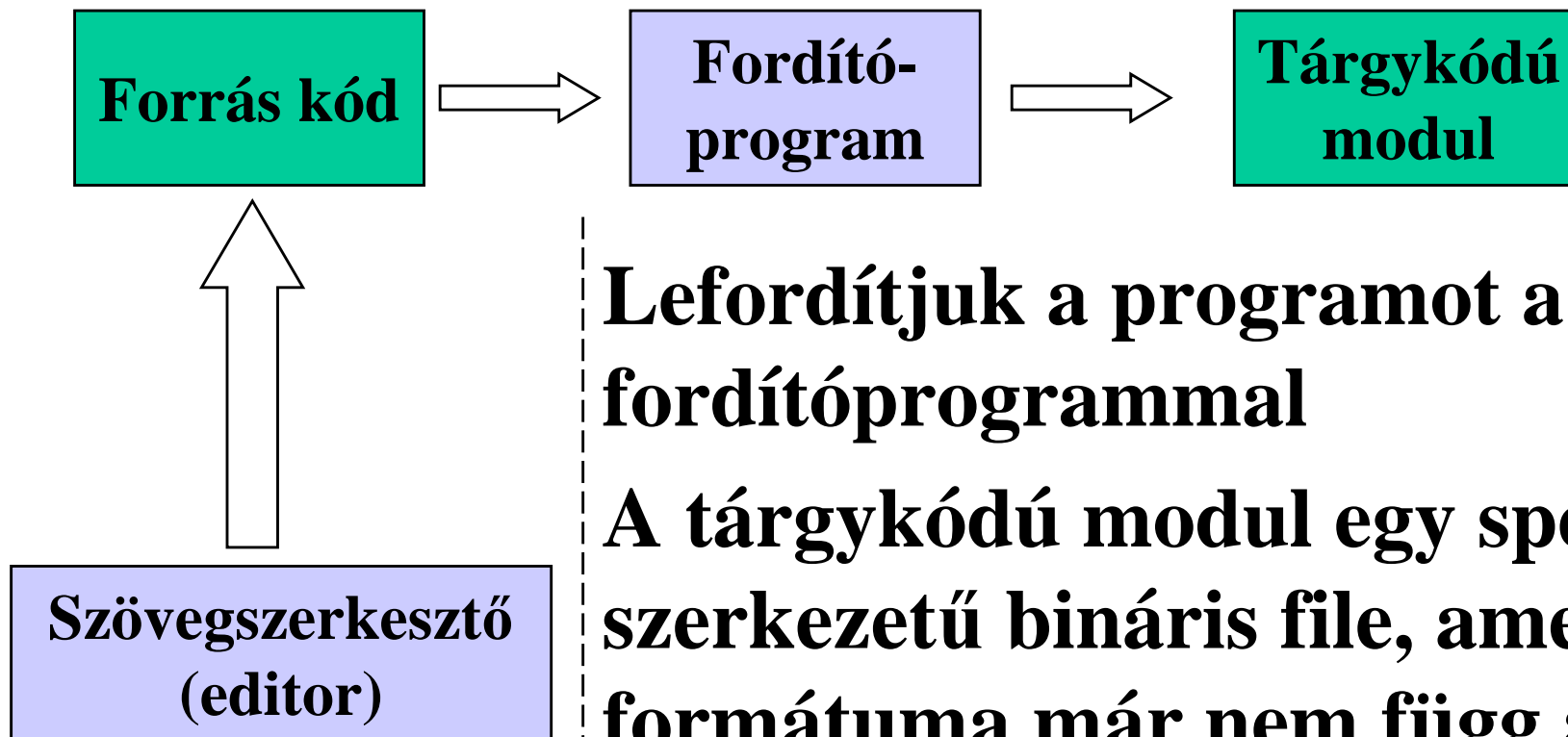
Ehhez egy szövegszerkesztő programot használunk

A file kiterjesztése általában utal a használt programozási nyelvre.

Pl.:

`proba.c`

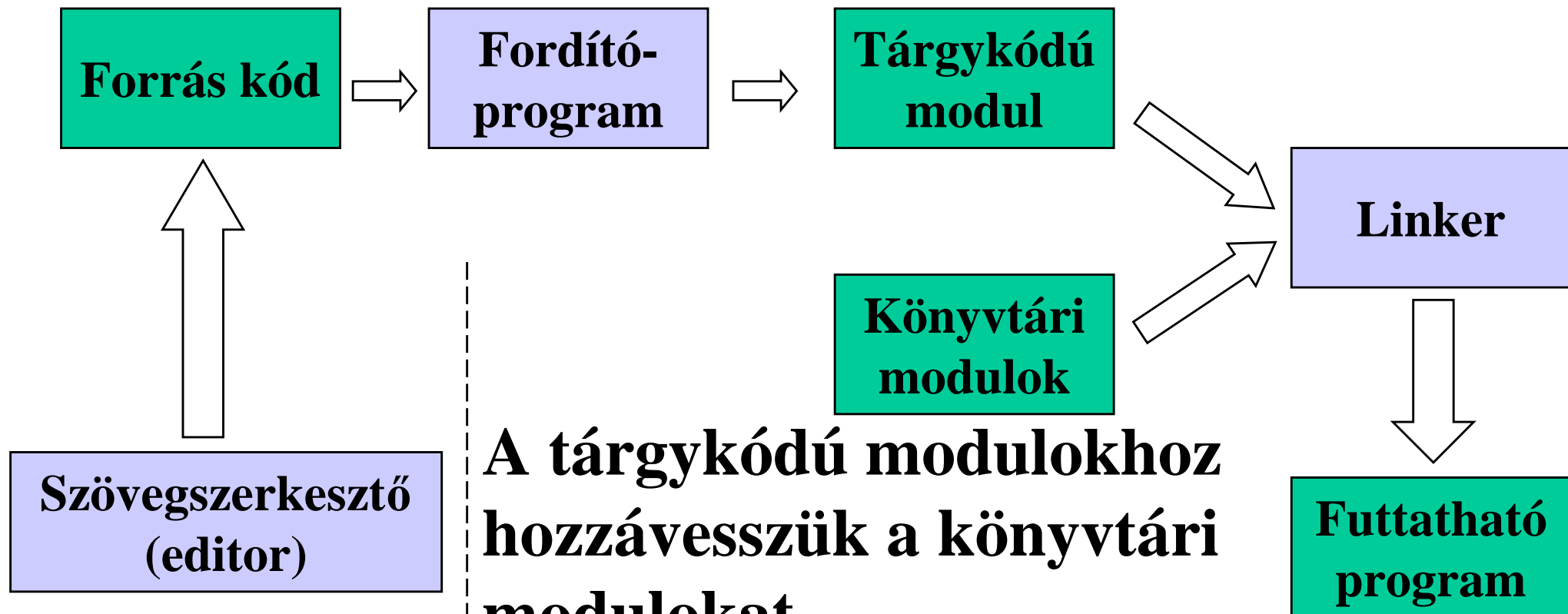
A programkészítés menete 2.



Lefordítjuk a programot a fordítóprogrammal

A tárgykódú modul egy speciális szerkezetű bináris file, amelynek formátuma már nem függ a programozási nyelvtől. Kiterjesztése általában .obj vagy .o

A programkészítés menete 3.



A tárgykódú modulokhoz hozzávesszük a könyvtári modulokat

A linker összeszerkeszti

A végeredmény a futtatható program

Megjegyzések

- A forrásprogram szövegfile, elkészítéséhez szövegszerkesztő (text editor) és nem dokumentum szerkesztő alkalmas. Windows alatt például a "Jegyzetömb" (Notepad) alkalmazás, és nem a Word.
- Az előző ábrák azt az esetet mutatják, amikor parancssoros felületen, az egyes programok egyedi indításával készítjük el a programot.

Megjegyzések (folyt.)

- Ma már a legegyszerűbb fejlesztőeszközök is integrálják az editor, compiler, a linker és a hibakereső (debugger) programokat, ezzel egyszerűsítve a munkát.
- Mivel egy valóságos program több tárgykódú modulból és esetenként több könyvtár felhasználásával szerkeszthető össze, a linkernek fel kell sorolni a szükséges modulokat. Erre is tartalmaz eszközöket egy integrált fejlesztő eszköz.

Megjegyzések (folyt.)

- Az előbb vázolt eszközök csak egyszerű **programok** és nem **alkalmazások** fejlesztését teszik lehetővé.
- Egy programozási nyelv például általában nem tartalmaz ablakos felhasználói felület készítésére alkalmas eszközöket. (Ez alól részben kivétel a Java nyelv.)
- A **program** és az **alkalmazás** közötti különbségről a későbbiek során még lesz szó.
- Valódi alkalmazások fejlesztésére összetettebb (és drágább) fejlesztő környezet szükséges.

A program és az alkalmazás különbsége 1.

- A program egy technikai fogalom: algoritmusok számítógépes megvalósítása.
- Az alkalmazás felhasználó vagy felhasználók egy csoportjának a munkáját vagy egyéb tevékenységét segítő számítógépes rendszer.
- Egy alkalmazás legfontosabb részei:
 - program vagy programok rendszere
 - működtető környezet (hardware/software)
 - adatok
 - dokumentációk

A program és az alkalmazás különbsége 2.

- A program tehát egy szűkebb fogalom, mint az alkalmazás (A tárgy keretében a gyakorlás érdekében programokat írunk.)
- Az alkalmazás mindig **felhasználó(k)** érdekében készül.
- Az alkalmazás kifejlesztése vagy megvásárlása pénzbe kerül, tehát termék, áru.
- A termékek megfelelő minőségű gyártásához szükséges ismeretek együttesét **technológiának** hívjuk. (Software technológia)

A program és az alkalmazás különbsége 3.

- A mai szóhasználatban a **software** szó a legtöbbször **alkalmazást** jelöl.
- Mivel az alkalmazás termék, a gyártásának a technológiáját ki kellett dolgozni - **software technológia**.