Programozás
(GKxB_INTM114)

Dr. Hatwagner F. Miklós

Széchenyi István Egyetem, Győr

https://github.com/wajzy/GKxB_INTM114.git

2024. február 18.

- Rendszertervezés (System Engineering)
  - Üzleti folyamat tervezés (Business Process Engineering)
  - Terméktervezés (Product Engineering)
- Szoftvertervezés (Software Engineering)
  - Követelményspecifikáció, -elemzés
  - Tervezés
  - Implementáció
  - Validáció, tesztelés
  - Telepítés
  - (Karbantartás, követés, továbbfejlesztés)

Irodalom: Dr. Ulbert Zsolt: Szoftverfejlesztési folyamatok és szoftver minőségbiztosítás

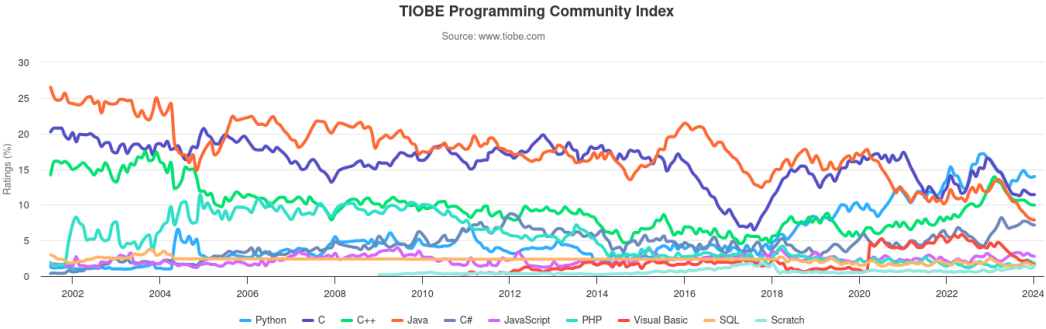
- Gépi kód
- Assembly

pelda02.asm (Forrás: Agárdi Gábor: Gyakorlati Assembly)

```
Pelda02 Segment                         ;Szegmensdefinicio.
        assume cs:Pelda02,ds:Pelda02    ;Cs es ds regiszterek beallitasa a szegmens elejere.
Start:  mov    ax,Pelda02               ;A ds regiszter beallitasa.
        mov    ds,ax
        mov    ax,0b800h                ;A kepernyomemoria szegmens-
        mov    es,ax                    ;cimet es regiszterbe tolti.
        mov    di,1146                  ;A di indexregiszterbe beallitja az offsetcimet.
        mov    al,"A"                   ;Al regiszterbe az "A" betu ascii kodjat tolti.
        mov    ah,7                     ;A betu szinet fekete alapon feher szinure allitja.
        mov    es:[di],ax               ;Az es:di altal mutatott cimre irja ax tartalmat azaz
                                        ;a fekete alapon feher "A" betut.
        mov    ax,4c00h                 ;Kilepes a DOS-ba.
        int    21h
Pelda02 Ends                            ;A szegmens vege.
        End    Start                    ;A program vege
```

- C
  - Dennis Ritchie, Bell Laboratories (1969-1973): C programnyelv → UNIX operációs rendszer
  - „Szabványok”: K&R (1978), ANSI (vagy C89, 1989), C99, C11.
  - Tulajdonságok: általános célú, imperatív (parancsoló, a programnak *hogyan* kell működnie a megfelelő állapotváltozások eléréséhez), strukturált (forrásfájlok, blokkok, ciklusok, stb. → áttekinthetőség)
- C++
  - Bjarne Stroustroup (1979): „C with Classes”
  - „Szabványok”: C++ (1983), „The C++ Programming Language” (1985), . . . , ISO/IEC 14882:2020
  - Tulajdonságok, általános célú, procedurális, funkcionális, objektum-orientált, nagyrészt C kompatibilis

- Irodalom
  - Brian W. Kernighan, Dennis M. Rithcie: A C programozási nyelv - Az ANSI szerint szabványosított változat
  - Benkő László, Benkő Tiborné, Tóth Bertalan: Programozzunk C nyelven! - Kezdőknek - középhaladóknak
  - Bauer Péter: C programozás
  - Bauer Péter, Hatwágner F. Miklós: Programozás I-II
  - Bjarne Stroustrup: A C++ programozási nyelv I-II. kötet
- Szoftverek
  - Microsoft Visual Studio
  - QT Creator IDE
  - GNU Compiler Collection
  - Code::Blocks
  - Geany



Tiobe programozási nyelv népszerűségi index, 2024. február

szamok.cpp

```cpp
#include <iostream>
using namespace std;

int main() {
  for(int i=1; i<=10; i++)
    cout << i << ' ';
  cout << endl;
  return 0;
}
```

Szamok.java

```java
class Szamok {
  public static void main(String[] args) {
    for(int i=1; i<=10; i++)
      System.out.print(i + " ");
    System.out.println();
  }
}
```

szamok.php

```php
<?php
  for($i=1; $i<=10; $i++)
    echo $i.' ';
?>
```

szamok.js

```javascript
var uzenet = "";
for(var i=1; i<=10; i++)
  uzenet += i + " ";
alert(uzenet);
```

1. Forrásszöveg megszerkesztése (többnyire .cpp kiterjesztés, ASCII szövegfájl)

elso.cpp

```
// Ez a sor egy megjegyzes
#include<iostream>
using namespace std;

int main() {
  cout << "Ez az elso C++ programunk!" << endl;
  return 0;
}
```

2. Összeállítás (build)

```
g++ -Wall -o elso elso.cpp
```

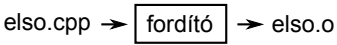
3. Futtatás

Linux terminál

```
wajzy@wajzy-notebook:~/Dokumentumok/gknb_intm114/ea01$ ./elso
Ez az elso C++ programunk!
wajzy@wajzy-notebook:~/Dokumentumok/gknb_intm114/ea01$
```

Az összeállítási folyamat résztevékenységei

1. Fordítás (compiler)



Fordítás (compile) GCC-vel

g++ -Wall -c elso.cpp

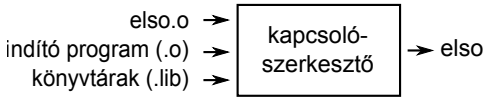
Üzenetek típusai:

- hibaüzenetek (error) → szintaktikai hiba, nem jön létre tárgymodul
- figyelmeztető üzenetek (warning) → figyelmeztetés gyanús megoldásra, javaslattétel, létrejön a tárgymodul (object file)

Az összeállítási folyamat résztevékenységei

2. Kapcsoló-szerkesztés (link)
   - fv.-ek tárgykódja: statikus könyvtárakban (.lib, run-time library vagy standard library)

g++ -o elso elso.o



A kapcsoló-szerkesztő hibaüzenetei


Összeállítás
g++ -Wall -o elso elso.cpp
elso.cpp
elso.o
elso
Fordítás
g++ -Wall -c elso.cpp
indító program
könyvtárak
Kapcsoló-szerkesztés
g++ -o elso elso.o

elso.cpp

```
// Ez a sor egy megjegyzes
#include<iostream>
using namespace std;

int main() {
  cout << "Ez az elso C++ programunk!" << endl;
  return 0;
}
```

Megjegyzések:

- // után a sor végéig
- /* és */ között akár több soron át
- Az előfeldolgozó törli őket

Direktívák:

- # kezdetű sorok
- #include<...> beszerkeszti a fejfájl (header) tartalmát → pl. konstansok, könyvtári függvények használatához (pl. `/usr/include/c++/4.8.4/iostream`)

Direktíva, megjegyzés: előfeldolgozó (preprocessor) dolgozza fel

A `main` függvény

- Függvény: adatok és végrehajtható utasítások csoportja. Működésük paraméterekkel hangolható, értéket adhatnak vissza.
- Függvény definíció: teljes információt szolgáltat a függvényről
- *típus függvénynév(formális-paraméterlista) { függvény-test }*
- A `main` speciális: a program belépési pontja (entry point)
- Állapotkódot ad vissza az OS-nek (0: minden OK)
- Visszatérési érték: `return` után

; utasítás (statement) végének jelzése

Szabványos folyamok

- Kimenet (stdout, ≈ képernyő), használata `cout`-tal
- Bemenet (stdin, ≈ billentyűzet), használata `cin`-nel
- Hiba (stderr, ≈ képernyő), használata `cerr`-rel (nem pufferelt)

A `cout`

- üzenetek megjelenítése
- `<<` operátor (műveleti jel): adat folyamba írása
- `endl` újsor karakter + puffer ürítése
- Képernyőre íráskor valójában *függvényhívás* történik

Névtér: halmaz, melyben minden azonosító egyedi

elso.cpp

```
// Ez a sor egy megjegyzes
#include<iostream>
using namespace std;

int main() {
  cout << "Ez az elso C++ programunk!" << endl;
  return 0;
}
```

elso_nevter.cpp

```
// Ez a sor egy megjegyzes
#include<iostream>

int main() {
  std::cout << "Ez az elso C++ programunk!" << std::endl;
  return 0;
}
```

Átmeneti állományok megőrzése

```
g++ -save-temps -Wall -o "elso" "elso.cpp"
```

Fájlok: előfeldolgozás eredménye, assembly.

Feladat: írjuk ki az első 10 természetes szám négyzetét!

negyzetszamok.cpp

```
#include<iostream>
using namespace std;
int main() {
  cout << "Termeszetes szamok negyzetei\n\n";
  cout << 1 << '\t' << 1*1 << '\n';
  cout << 2 << '\t' << 2*2 << '\n';
  cout << 3 << '\t' << 3*3 << '\n';
  cout << 4 << '\t' << 4*4 << '\n';
  cout << 5 << '\t' << 5*5 << '\n';
  cout << 6 << '\t' << 6*6 << '\n';
  cout << 7 << '\t' << 7*7 << '\n';
  cout << 8 << '\t' << 8*8 << '\n';
  cout << 9 << '\t' << 9*9 << '\n';
  cout << 10 << '\t' << 10*10 << '\n';
  return 0;
}
```

Kimenet

```
Termeszetes szamok negyzetei

1       1
2       4
3       9
4       16
5       25
6       36
7       49
8       64
9       81
10      100
```

Literálok: forrásszövegbe gépelt konstansok

- Egész konstansok
- Karakter konstansok: '-ok között
- Karakterlánc (string) konstansok: "-ek között

Vezérlőkarakterek, nem nyomtatható jelek, szintaktikai jelentéssel bíró jelek megadása → escape jelsorozat (escape sequence), \ jel vezeti be, leggyakrabban használtak:

Esc. szekv.	Jelentés
\b	visszalépés (backspace)
\n	új sor (new line)
\r	kocsi vissza (carriage return)
\t	vízszintes tabulátor (horizontal tab, HTAB)
\\	fordított törtvonal (backslash)
\'	aposztróf
\"	idézőjel
\ooo	oktális szám
\xhh	hexadecimális szám
\0	zérus ASCII kódú karakter

Néhány aritmetikai operátor

Operátor	Leírás	Példa
+	Összeadás	$5 + 3 == 8$
-	Kivonás	$5 - 3 == 2$
*	Szorzás	$5 * 3 == 15$
/	Egészosztás	$5/3 == 1$
%	Maradék képzés	$5\%3 == 2$

Megjegyzések:

- Kis egész kitevőjű hatványok $\rightarrow$ szorzás(ok)
- Meddig tart majd a gépelés (+kódméret nő, +hibalehetőségek), ha az első 1000 szám négyzetére lesz szükség?!

negyzetszamok2.cpp

```
#include<iostream>
using namespace std;
int main() {
  cout << "Termeszetes szamok negyzetei\n\n";
  int szam;
  szam = 1;
  while(szam <= 10) {
    cout << szam << '\t' << szam*szam << '\n';
    szam = szam + 1;
  }
  return 0;
}
```

Változók

- Pl.: `int szam;`
- típus
  - az adat jellege (numerikus, szöveges)
  - hogyan tárolják a memóriában
  - milyen művelet végezhető vele
- memóriaterület
  - értéket tárolja típusnak megfelelően
  - lokális változók ({ és } közötti blokkokban) kezdőértéke definiálatlan, „memóriaszemét”
- név, vagy azonosító (funkcióra utaló, „beszédes elnevezés”)

Azonosítóképzési szabályok

- Első karakter: kis- vagy nagybetű, ill. _
- További karakterek: u. a., és számjegyek
- Nem lehet kulcsszó vagy védett azonosító
- Kis- és nagybetűre érzékenyek
- Ajánlás: ne kezdődjön egy vagy két _ karakterrel
- Szignifikáns karakterek száma

Mi lehet a gond?

Gipsz Jakab
66_os_ut
Menő_Manó
auto

OK

meno_mano
Meno_Mano
sokReszbolOsszeteve

Fontosabb egész típusok (fixpontos ábrázolás) → alaptípus és típusmódosítók

Rögzített méretű egész típusok (C++11)

Típus	Leírás
`char`	Alt. előjeles, 8 bites egész
`signed char`	Előjeles 8 bites egész
`unsigned char`	Előjel nélküli, 8 bites egész

`short`
`signed short`
`signed short int`
	Előjeles rövid egész

`unsigned short` `unsigned short int`	Előjel nélküli rövid egész

`signed`
`int`
`signed int`
	Előjeles egész

`unsigned` `unsigned int`	Előjel nélküli egész

`long`
`signed long`
`signed long int`
	Előjeles hosszú egész

`unsigned long` `unsigned long int`	Előjel nélküli hosszú egész

Megjegyzések:

- Típusmódosítók: `signed/unsigned`, `short/long`
- Egész literál ábrázolása: `int`
- `char` típus mérete: mindig 1 bájt, de a karakter literál `int`-ben!
- `char` típus előjel-kezelése: platformtól és fordítótól függ, de ált. előjeles és beállítható
- `1 == sizeof(char) <= sizeof(short) <= sizeof(int) <= sizeof(long) <= sizeof(long long)`, ahol `sizeof` a típus/változó méretét bájtban megadó operátor

Változó definíció

- Általános alak: *típus azonosítólista;*
- Azonosító, típus megadása, memóriaterület foglalása
- Pl.: `int x; int i, j, k; unsigned int y;`

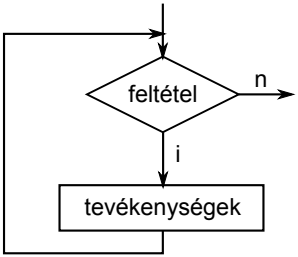
Értékadás

- Operátor: =
- *balérték* = *jobbérték*;
- kifejezés (expression): értéket állít elő konstansok, változók, műveletek (operátorok) segítségével

Relációs operátorok

Operátor	Leírás
==	egyenlő
!=	nem egyenlő
<	kisebb
<=	kisebb, vagy egyenlő
>	nagyobb
>=	nagyobb, vagy egyenlő

Elöltesztelő ciklus



<megelőző tevékenységek>
while(feltétel kifejezése) {
*   tevékenységek*
}
<további tevékenységek>

A *ciklusmag* (ismételt rész) lehet

- egyetlen egyszerű utasítás
- összetett utasítás: több utasításból képzett blokk

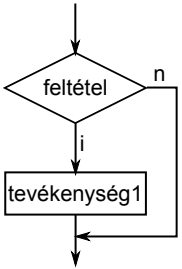
Olvassunk be egy egész számot, majd döntsük el, hogy páros-e!

paros.cpp

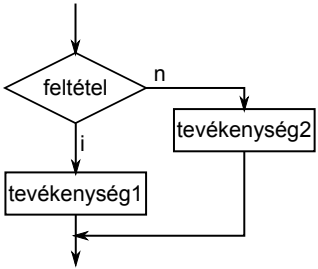
```
#include<iostream>
using namespace std;
int main() {
  int szam;
  cout << "Adjon meg egy egeszet, amirol eldontjuk, "
       << "hogy paros-e vagy paratlan!" << endl;
  cin >> szam;
  if(szam%2 == 0) {
    cout << "A szam paros." << endl;
  } else {
    cout << "A szam paratlan." << endl;
  }
  return 0;
}
```

Beolvasás szabvány bemenetről: std::cin

Szelekció



if(kifejezés) utasítás1



if(kifejezés) utasítás1
else utasítás2

Utasítások lehetnek összetettek → többirányú elágazás

Műveleti sorrend (kifejezések)

- zárójelezés
- műveletek precedenciája (időbeli sorrend)

Operátor	Asszociativitás
sizeof	jobbról balra
* / %	balról jobbra
+ -	balról jobbra
< <= > >=	balról jobbra
== !=	balról jobbra
=	jobbról balra

Vezérlési szerkezetek

- szekvencia
- iteráció
- szelekció