Imperatív programozás

Áttekintés

Kozsik Tamás és mások

Eötvös Loránd Tudományegyetem

2022. augusztus 20.

ELTE
EÖTVÖS LORÁND
TUDOMÁNYEGYETEM

Tartalomjegyzék

A tárgy célja

- Fogalomrendszer
- Terminológia magyarul és angolul
- Tudatos nyelvhasználat (C programozási nyelven)
  - Imperatív programozás
  - Procedurális programozás
  - Moduláris programozás
- Részben: programozási készségek
- Linux és parancssori eszközök használata
- TIOBE index (`https://www.tiobe.com/tiobe-index/`)

Programozási paradigmák

Meghatározás

Gondolkodási sémák, szükséges nyelvi eszközök

> „. . . egy nyelvről akkor mondjuk, hogy *támogat* egy programozási stílust, ha olyan szolgáltatásai vannak, melyek által az adott stílus használata kényelmes (könnyű, biztonságos és hatékony) lesz.”
>
> *Bjarne Stroustrup - A C++ programozási nyelv*

Programozási paradigmák

Példák

- Imperatív programozás
- Funkcionális programozás
- Logikai programozás

- Procedurális programozás
- Moduláris programozás
- Objektumelvű programozás

- Szekvenciális programozás
- Konkurens programozás
- Párhuzamos programozás
- Elosztott programozás

- Aspektuselvű programozás
- Komponenselvű programozás
- Szolgáltatáselvű programozás
- Szerződésalapú programozás

Programozási paradigmák

Példák az algoritmus stílusa alapján

Deklaratív

Funkcionális (Haskell)

```
sum [] = 0
sum (x:xs) = x + sum xs
```

Logikai (Prolog)

```
sum_list([], 0).
sum_list([X|Xs], Sum) :-
  sum_list(Xs, Rest),
  Sum is X + Rest.
```

Imperatív (C)

```
int array[SIZE] = ...;
int sum = 0;

for (int i = 0; i < SIZE; ++i)
  sum += array[i];
```

Végrehajtás módja

Parancsértelmező (interpreter)

- Forráskód feldolgozása utasításonként
  - Ha hibás az utasítás, hibajelzés
  - Ha rendben van, végrehajtás
- Az utasítás végrehajtása: beépített gépi kód alapján

Hátrányok

- Futási hiba, ha rossz a program (ritkán végrehajtott utasítás???)
- Lassabb programvégrehajtás

Előnyök

- Programírás és -végrehajtás integrációja
  - REPL = Read-Evaluate-Print-Loop
  - Prototípus készítése gyorsan
- Kezdők könnyebben elsajátítják

Végrehajtás módja

Fordítás és futtatás szétválasztása

Hello world Pythonban (`main.py`)

```
print("Hello World!")
```

Interpretálás, futtatás

```
$ python main.py
Hello World!
```

Végrehajtás módja

Fordítás és futtatás szétválasztása

- Sok programozási hiba kideríthető a program futtatása nélkül is
- Előre megvizsgáljuk a programot
- Ezt csak egyszer kell (a fordítás során)
- Futás közben kevesebb hiba jön elő
- Cél: hatékonyabb és megbízhatóbb gépi kód
- „Fordítási idő” és „Futási idő”

Végrehajtás módja

Fordítás és futtatás szétválasztása

Hello world C-ben (main.c)

```c
#include <stdio.h>

int main()
{
  printf("Hello World!\n");
}
```

Fordítás, futtatás

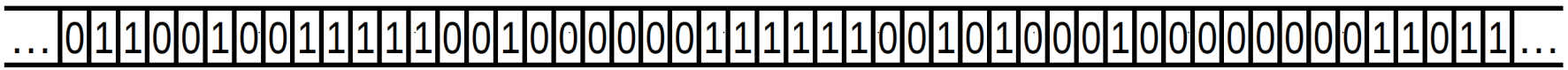
```
$ gcc main.c
$ ./a.out
Hello World!
```

Alacsony és magas szintű nyelvek

Közelség a hardver architektúrához

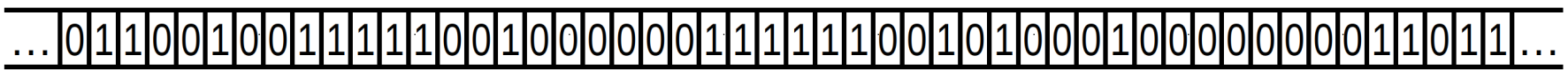
Alacsony és magas szintű nyelvek

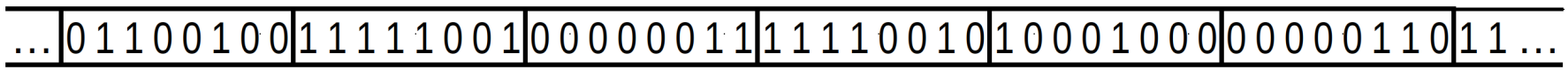
Közelség a hardver architektúrához



Alacsony és magas szintű nyelvek

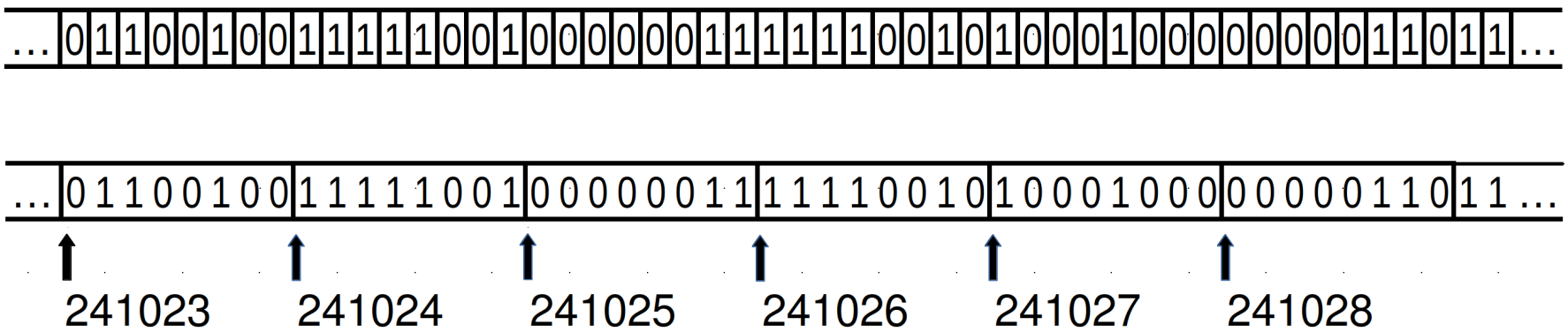
Közelség a hardver architektúrához





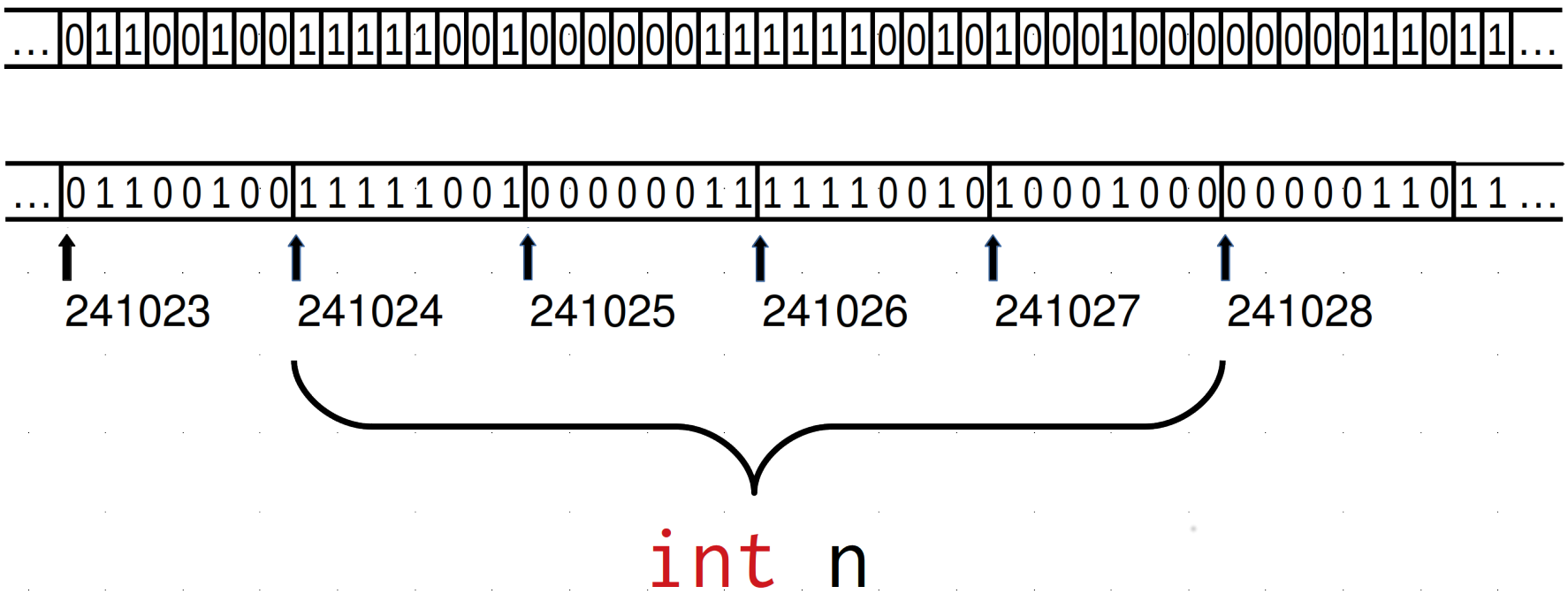
Alacsony és magas szintű nyelvek

Közelség a hardver architektúrához



Alacsony és magas szintű nyelvek

Közelség a hardver architektúrához



Alacsony és magas szintű nyelvek

Assembly

```
...
main:
.LFB0:
    .cfi_startproc
    pushq   %rbp
    .cfi_def_cfa_offset 16
    .cfi_offset 6, -16
    movq    %rsp, %rbp
    .cfi_def_cfa_register 6
    subq    $16, %rsp
    movl    $0, -8(%rbp)
    movl    $1, -4(%rbp)
    jmp .L2
.L3:
    movl    -4(%rbp), %eax
    addl    %eax, -8(%rbp)
    addl    $1, -4(%rbp)
.L2:
    cmpl    $10, -4(%rbp)
    jle .L3
    movl    -8(%rbp), %eax
    movl    %eax, %esi
    leaq    .LC0(%rip), %rax
    movq    %rax, %rdi
    movl    $0, %eax
    call    printf@PLT
    movl    $0, %eax
    leave
...
```

C

```c
#include <stdio.h>

int main()
{
  int sum = 0;

  for (int i = 1; i < 11; ++i)
    sum += i;

  printf("%d\n", sum);
}
```

Python

```python
print(sum(range(1, 11)))
```

ELTE
EÖTVÖS LORÁND
TUDOMÁNYEGYETEM

Programok felépítése

- Kulcsszavak, literálok, operátorok, egyéb jelek, azonosítók
- Kifejezések
- Utasítások
- Alprogramok (függvények/eljárások, rutinok, metódusok)
- Modulok (könyvtárak, osztályok, csomagok)

Viselkedés definiáltsága

- Lefordul-e?
  (Nyelvi szabályok betartása, pl. pontosvessző az utasítások végén)
- Futás közben történik-e hiba?
  (Nullával való osztás)
- Definiált-e a program viselkedése?
  (10 elemű tömb 11-ik elemének lekérdezése)
- Platformfüggő-e a viselkedés?
  (Melyik a legnagyobb egész szám?)
- Implementációfüggő-e a konstrukció?
  (Függvénydefiníció függvény törzsében)