Programozási Nyelvek

Pataki Norbert

Programozási Nyelvek és
Fordítóprogramok Tanszék

Programozási Nyelvek (C++)

Órakezdés

Számonkérés

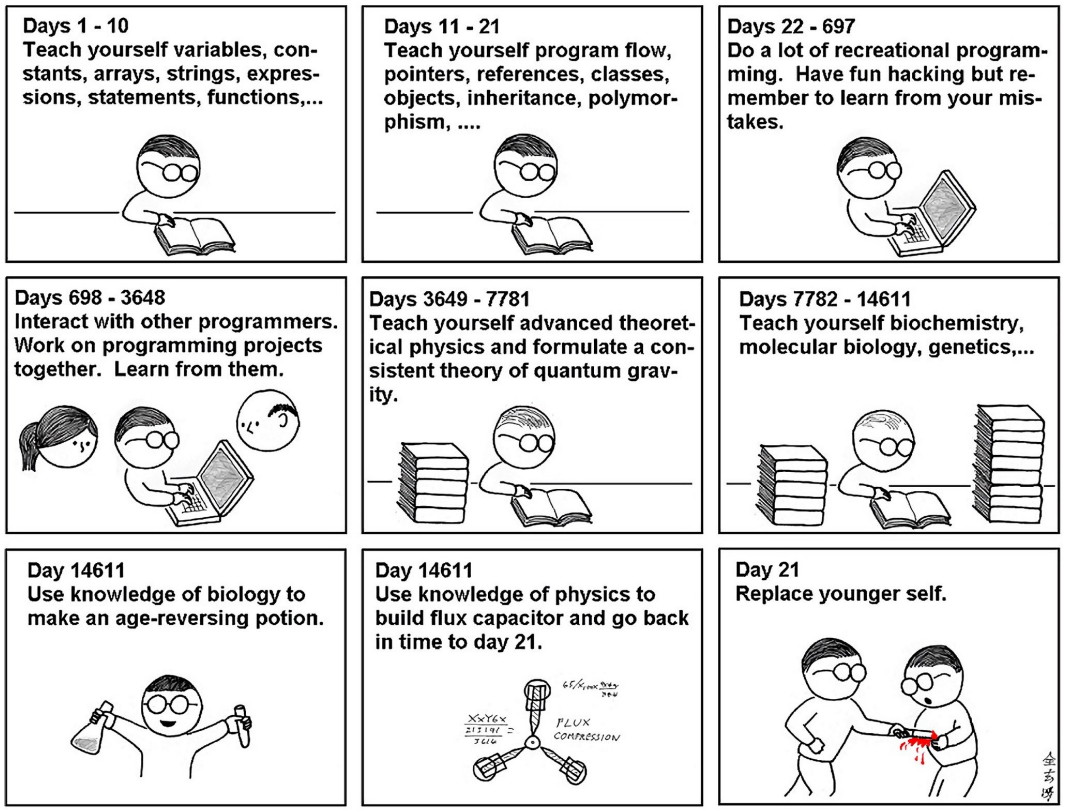
- 3 darab +/- (gyakorlat)
- Opcionális beadandó, code review
- Vizsga:
  - Elmélet: 15 tesztkérdés, kb. 30-45 perc
  - +/- szerepe
  - Elmélet értékelése:
    - 8 pont alatt: 1
    - 8-9 pont: 2
    - 10-11 pont: 3
    - 12-13 pont: 4
    - 14- pont: 5
  - Vizsga: Gyakorlati feladat, 3 óra
  - Vizsgajegy: elmélet + gyakorlat / 2.0, ha az elmélet és a gyakorlat is legalább 2-es
  - Vizsgajegy: gyakorlat felé kerekítve, ha az átlag nem egész szám

Ajánlott irodalom

- Bjarne Stroustrup: A C++ programozási nyelv
- Scott Meyers: Hatékony C++

Ellenjavallt irodalom

21 days


Days 1 - 10
Teach yourself variables, constants, arrays, strings, expressions, statements, functions,...
Days 11 - 21
Teach yourself program flow, pointers, references, classes, objects, inheritance, polymorphism, ...
Days 22 - 697
Do a lot of recreational programming. Have fun hacking but remember to learn from your mistakes.
Days 698 - 3648
Interact with other programmers. Work on programming projects together. Learn from them.
Days 3649 - 7781
Teach yourself advanced theoretical physics and formulate a consistent theory of quantum gravity.
Days 7782 - 14611
Teach yourself biochemistry, molecular biology, genetics,...
Day 14611
Use knowledge of biology to make an age-reversing potion.
Day 14611
Use knowledge of physics to build flux capacitor and go back in time to day 21.
FLUX COMPRESSION
Day 21
Replace younger self.
As far as I know, this is the easiest way to
"Teach Yourself C++ in 21 Days".

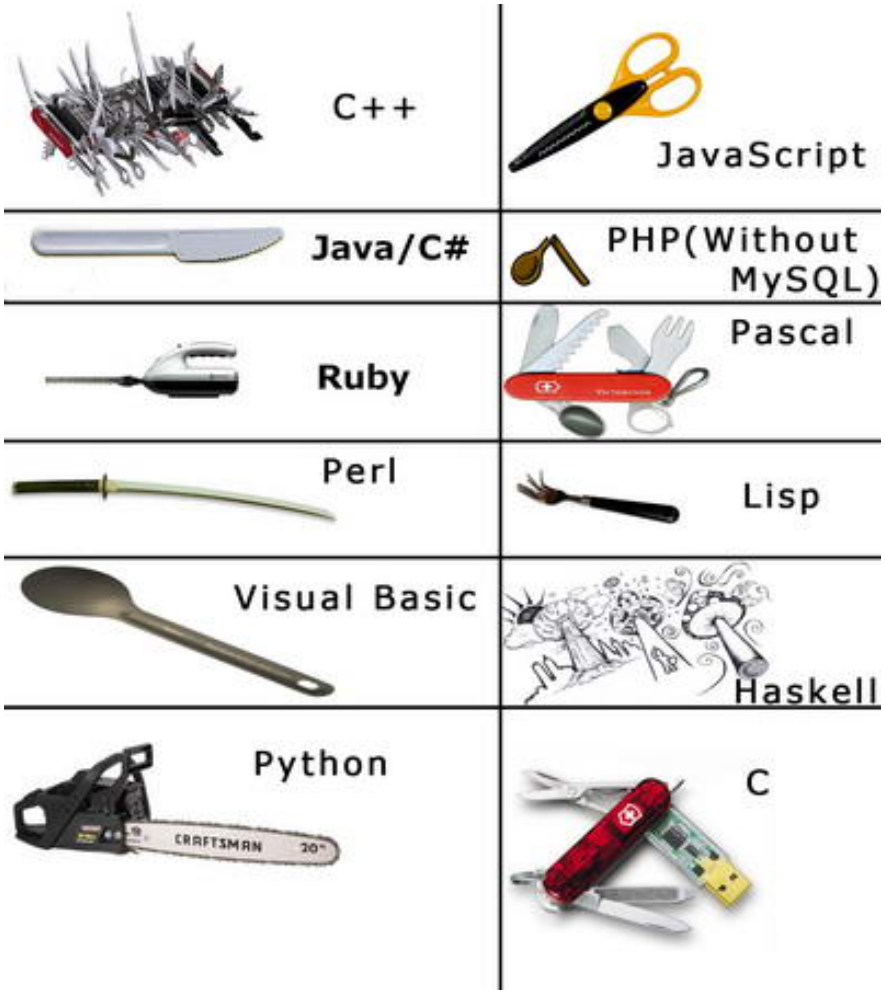
21 days


LEARN C++
IN ONE VIDEO
2:52 / 35040:04
Speed
0.25
0.5
0.75
Normal
1.25
1.5
1.75
30000

Tematika

- Fő témák:
  - Nyelvek (változása)
  - C++ fókuszban – előnyök és hátrányok
  - Gyakorlat
  - Előadás
  - Nyelvi fogalmak
- Kapcsolódó témák:
  - Compilerek, statikus elemzés
  - Forráskód szerepe, esztétikus kódok
  - Jó programozói gyakorlatok nyelv-független módon
  - CLI eszközök (tudatos használata)

Programozási nyelvek



C++

Történelem

- 1930-as évek: matematikai alapok
- 1945: ENIAC
- Kezdetben fizikai huzalozás
- 1950-es évek: Fortran
- ...

Osztályozás

- Általános célú vs. Domain-specifikus
- Alacsony-szintű vs. Magas-szintű
- Fordított vs. interpretált vs. bytecode
- Típusrendszeri sajátosságok
- Programozási paradigmák

Alacsony-szintű kód

```
    .text
    .globl  _Z3gcdii
    .type   _Z3gcdii, @function
_Z3gcdii:
.LFB0:
    .cfi_startproc
    pushq   %rbp
    .cfi_def_cfa_offset 16
    .cfi_offset 6, -16
    movq    %rsp, %rbp
    .cfi_def_cfa_register 6
    movl    %edi, -4(%rbp)
    movl    %esi, -8(%rbp)
.L5:
    movl    -4(%rbp), %eax
    cmpl    -8(%rbp), %eax
    je  .L2
    movl    -4(%rbp), %eax
    cmpl    -8(%rbp), %eax
    jle .L3
    movl    -8(%rbp), %eax
    subl    %eax, -4(%rbp)
    jmp .L5
.L3:
    movl    -4(%rbp), %eax
    subl    %eax, -8(%rbp)
    jmp .L5
.L2:
    movl    -4(%rbp), %eax
    popq    %rbp
    .cfi_def_cfa 7, 8
    ret
    .cfi_endproc
.LFE0:
    .size   _Z3gcdii, .-_Z3gcdii
    .ident  "GCC: (Ubuntu 5.4.0-6ubuntu1~16.04.12) 5.4.0 20160609"
    .section    .note.GNU-stack,"",@progbits
```

Programozási paradigmák fogalma

- Összetett problémák
- Dekompozíció
- Eszközök
- Know-how

Programozási paradigmák

- Imperatív
- Procedurális
- Strukturált
- Objektum-orientált
- Deklaratív
- Funkcionális
- Logikai
- Generikus

Programozási paradigmák

- Generatív technológiák
  - Aspektus-orientált
  - Metaprogramozás
- Intencionális
- Feature-orientált

C++: multiparadigmás nyelv

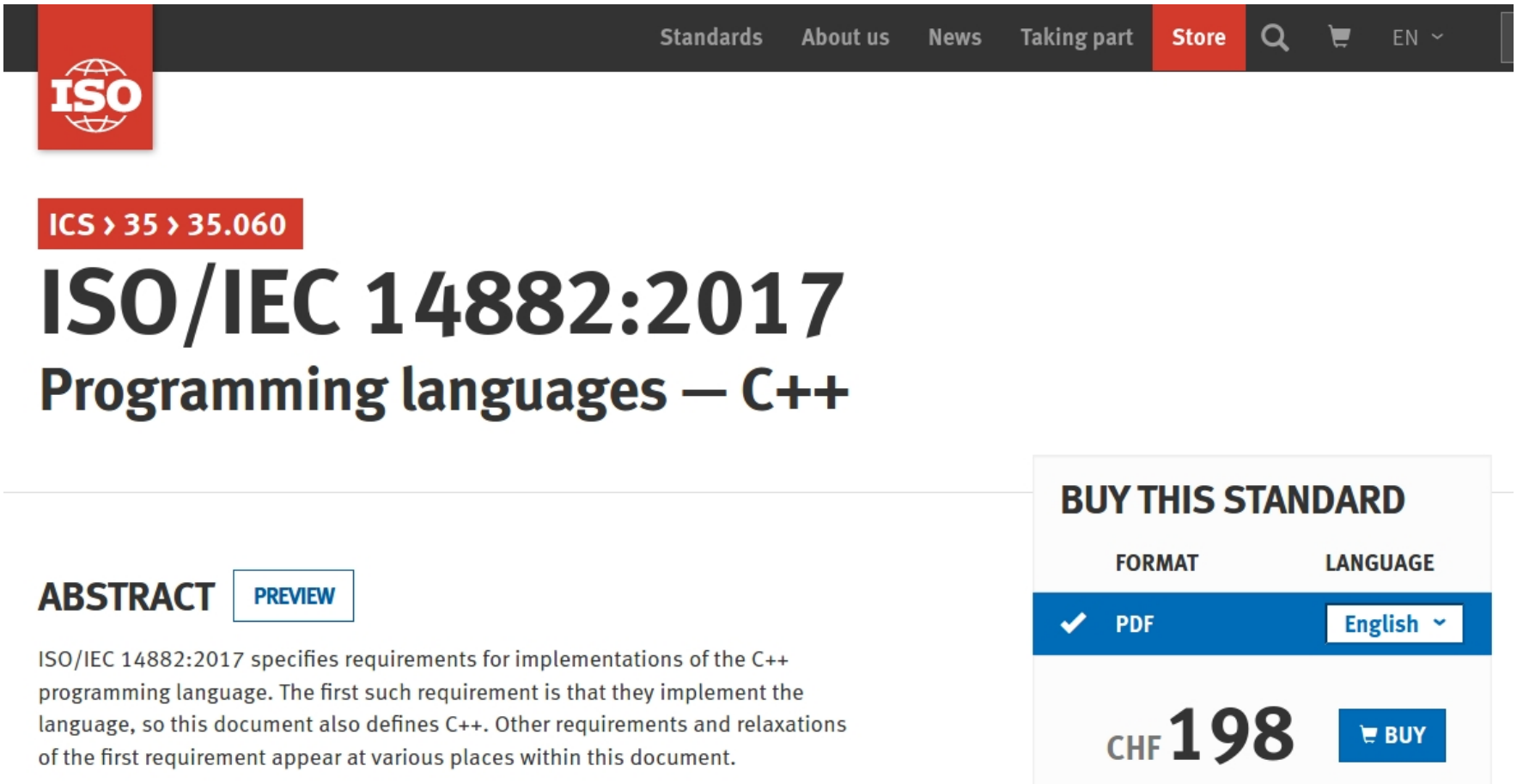
- Procedurális alapok (C)
- Objektum-orientáltság
- Generikus programozás (STL)
- Metaprogramozás

```
<source>: In member function 'void Prime_print<i>::f() [with int i = 13]':
<source>:37:   instantiated from 'void Prime_print<i>::f() [with int i = 14]'
<source>:37:   instantiated from 'void Prime_print<i>::f() [with int i = 15]'
<source>:37:   instantiated from 'void Prime_print<i>::f() [with int i = 16]'
<source>:56:   instantiated from here
<source>:36: error: invalid conversion from 'int' to 'void*'
<source>:36: error:   initializing argument 1 of 'D<i>::D(void*) [with int i =
<source>: In member function 'void Prime_print<i>::f() [with int i = 11]':
<source>:37:   instantiated from 'void Prime_print<i>::f() [with int i = 12]'
<source>:37:   instantiated from 'void Prime_print<i>::f() [with int i = 13]'
<source>:37:   instantiated from 'void Prime_print<i>::f() [with int i = 14]'
<source>:37:   instantiated from 'void Prime_print<i>::f() [with int i = 15]'
<source>:37:   instantiated from 'void Prime_print<i>::f() [with int i = 16]'
<source>:56:   instantiated from here
<source>:36: error: invalid conversion from 'int' to 'void*'
<source>:36: error:   initializing argument 1 of 'D<i>::D(void*) [with int i =
<source>: In member function 'void Prime_print<i>::f() [with int i = 7]':
<source>:37:   instantiated from 'void Prime_print<i>::f() [with int i = 8]'
<source>:37:   instantiated from 'void Prime_print<i>::f() [with int i = 9]'
<source>:37:   instantiated from 'void Prime_print<i>::f() [with int i = 10]'
<source>:37:   instantiated from 'void Prime_print<i>::f() [with int i = 11]'
<source>:37:   instantiated from 'void Prime_print<i>::f() [with int i = 12]'
<source>:37:   instantiated from 'void Prime_print<i>::f() [with int i = 13]'
<source>:37:   instantiated from 'void Prime_print<i>::f() [with int i = 14]'
```

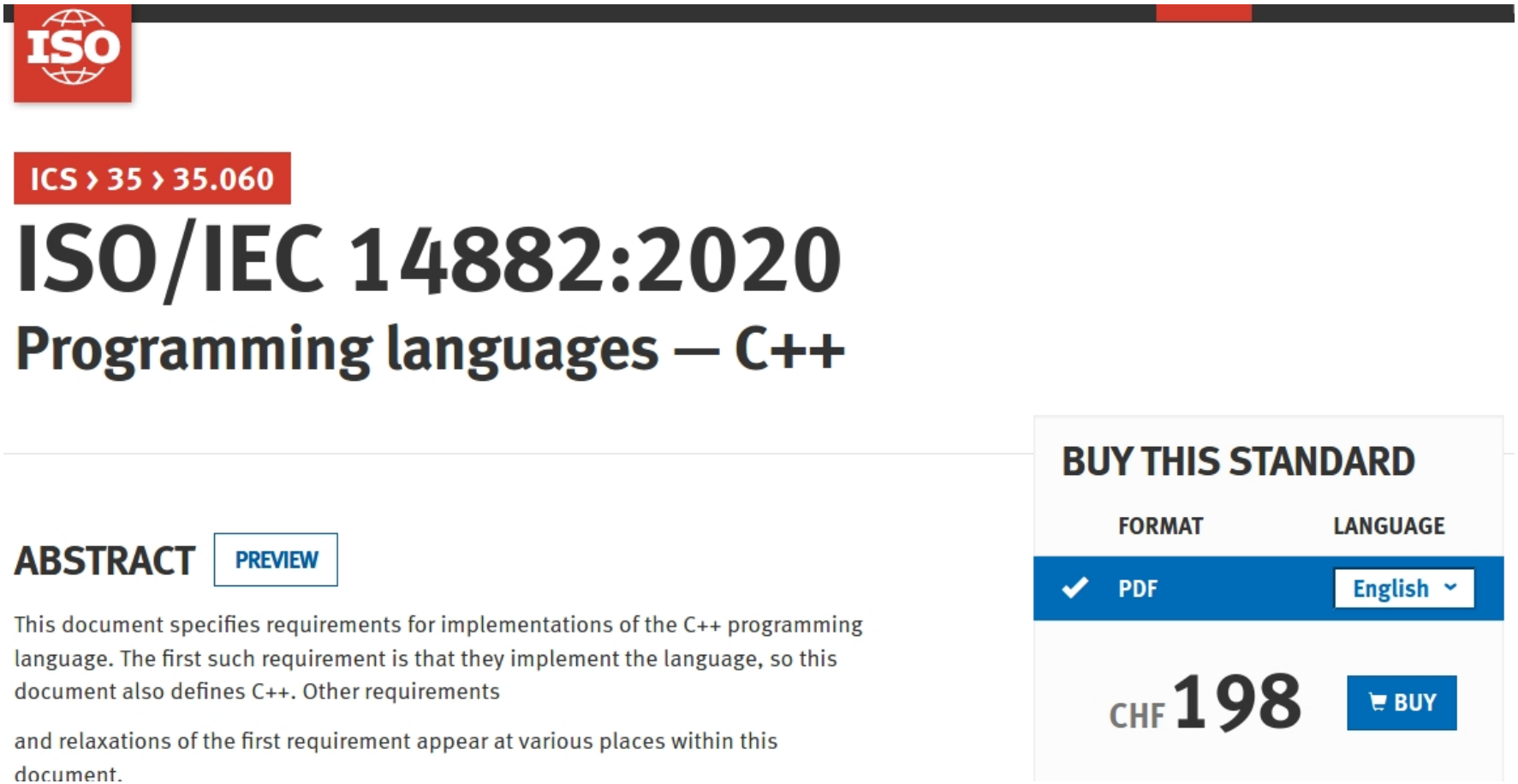
C++ történelem

- Simula-67
- 1970: Unix, C
- 1980: Stroustrup elkezd dolgozni a C++ nyelven
- 1994: HP STL
- 1994: Erwin Unruh metaprogram
- 1998: Első ISO C++ szabvány
- 2003: Javítások
- C++11/14/17/20

C++17 Standard



C++20 Standard



ICS › 35 › 35.060

ISO/IEC 14882:2020

Programming languages — C++

BUY THIS STANDARD

FORMAT	LANGUAGE
✔ PDF	English

CHF 198 BUY

ABSTRACT PREVIEW

This document specifies requirements for implementations of the C++ programming language. The first such requirement is that they implement the language, so this document also defines C++. Other requirements

and relaxations of the first requirement appear at various places within this document.

A C++ alapkoncepciói

- „C backward kompatibilitás”
- Jobb eszközök a C-s problémákra
- Hatékonyság **és** magas-szintű nyelvi konstrukciók
- A fordítóprogram a „barátunk”
- Ne fizessünk azért, amit nem kérünk
- Általános, hatékony és átfogó szabványkönyvtár