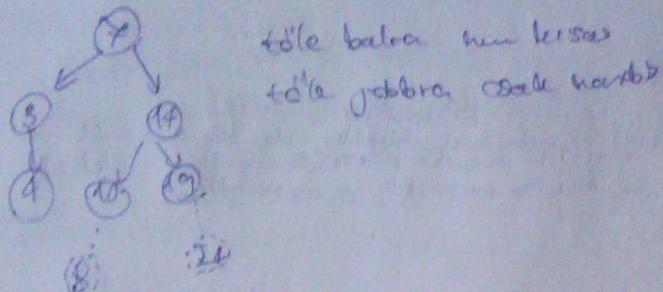


Feladat: Adott inputot!
Szerződött összefüggésekkel be,
(előiről): rendesített címkékkel.

Megoldás: Kereszt a segítségével



Összefoglalás (referenciák)

A referencia az ellenőrzi, az eredeti "fárkereje" (a pointerhez kapcsolva). Pointortól eltérően nincs saját mutatója, nem lehet dereferálni őt műves arithmetikával.

int $x = 1;$ $\rightarrow \boxed{1}$
int $\&r = x;$ \rightarrow

$\Gamma \models ;$ $\begin{array}{c} r, x: \text{int} \\ \boxed{x=2} \\ \{ \end{array}$
 $\Gamma = 10;$ $\begin{array}{c} r, x: \text{int} \\ \boxed{10} \end{array}$ } Ugyanaz a hely 2-től kezdve 10
 ~~= 10~~ 10 lenne.

Sziszimulációhoz használható.

void f(int) \rightarrow "Felfogás", változatos, eredeti érték megfelelő

void g(&int)

void g(int&) \rightarrow feldolgozás, változtatás az eredeti értékben is.)

void h(const int&) \rightarrow feldolgozás az eredeti értékben, változtatás nélkül!

Pataki Norbert | patakino@elte.hu | C++ | C++
9gyar

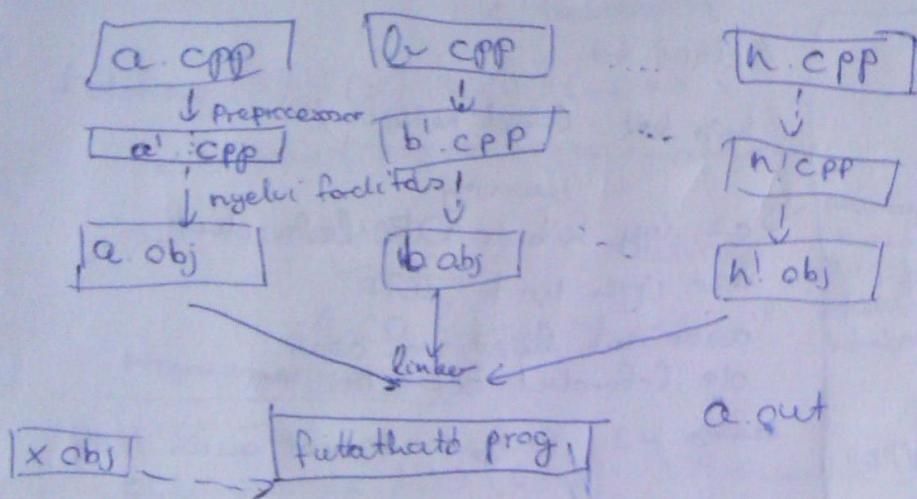
18:00 - 18:30

Szövegkeverés:

- 4 db +/- nem elérkezett bejelentett 1/1
- opcionális bejelentés
- <http://patakino.web.elte.hu/png2/>
- fejlesztői ZH:
 - 1. rész bejegyzés (besz)
 - 2. rész gyakorlat

1. gyakorlat

C/C++ fordítási modellje



1. preprocessor
file bemenetlása
szöveges kiírás
beállítások
search & replace

2. Nyelv Fordítás

3. Linker

Stabilizálás a színtelenben

1. nem célszó fajl include-olásra
#include "BLABLA"
/#endif

- miért nyelv az ami nem esetleges szintű objektum ad?
- + interpreter... interpretálva PHP
- többek között alkalmi nyelv JAVA
jól param, adb, assztyf → .obj

2. const int N = 42

 ++N;
 int f(int);
{...}
}

a.obj

 int f(int);
{
}...

b.obj

obban az a fordítási sorrendben
ez helyes, de nem adja +az ezeret
helytudja előfordulni az f hívásoknál
hogy meglé f(x)
Röviden meg:

Ambiguous reference

int fac(int);
int x = fac(5);

// fac ("xy");

(megadottuk a típusát!, mi + van, és kiválasztottuk a legelsőt, hogy helyes -e.)

Amikor nem találja meg f(x),
annak el kell várni.
Ezatlanos referencia

C++
GCC

gcc/g++

1/2

- g++ hello.cpp

→ a.out
 $(\text{./a.out} \rightarrow \text{Linux})$
 $\text{a.exe} \rightarrow \text{Windows}$

- g++ hello.cpp a.cpp b.cpp n.obj

-w -Wall függelvezetésű kiadvány
nincs 'C' oldal

int x = -2;

unsigned int y = 3;

if ($x \leq y$)

{ std::cout << "ok" ;

}

else

{ std::cout << "WTF" ;

}

// eljel bőt

// hely bőt (eljel helyes!)

→ az így hibás DE lefordult

azt (jön mi a WTF)

azonosítást van csinálva

de lefordult azt elazítja

mesg -2 | ha az unsigned okan lesz

unused ...

- pedantic -ansi

long long típus

- zöve-famps

cout << k;

std::cin >> k;

cout << [k];

futás közben elvill ki
C++ szabvány szerint felel

- C (nincs többé weak obj; generalisált)

- g

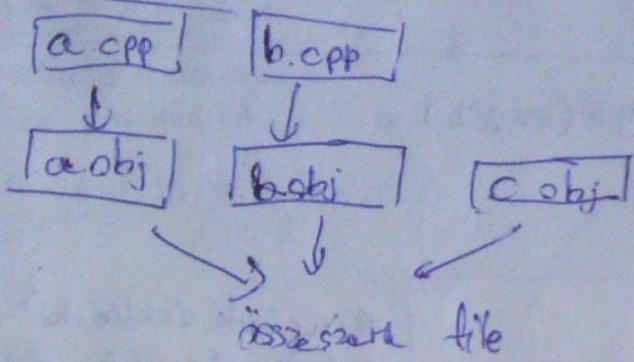
- O filename

- O n

- ove

Optimalizációk kölcsönök
 $n=0, 1, 2, 3 \rightarrow$ minden

Összeszerkezetes



- statikus linkelés: linkeléskor minden fületet feloldunk minden
- dinamikus linkelés fülettekkel több hagyomány körbe kerülhetnek feloldásra

2. C++ Gyakorlat

#

C - program helloWorld.c

```
#include < stdio.h >
int main ()
{
    printf ("Hello world!");
    return 0;
}
```

iostream

#include < iostream >

int main ()

{

std::cout << "Hello World" << endl;

}

return 0;

std::endl;

using namespace std;

new tell std:::

Using std::cout;

→ std::cout new tell endl - hasz (zen)

PVM: könyvtár

pvm - send (...)

PVM - receive (...)

PVM - ...

newterez (namespace) letrehozasa

namespace A

```
void f() → A:f()
{
}
}
```

namespace B

```
void f() /B:f()
{
}
}
```

22 Jeladat intervallen $[-80, +150]$ berechne Celsius Werte

-80°F ... ${}^{\circ}\text{C}$

-70°F ... ${}^{\circ}\text{C}$

$$x \mapsto 5/9 * (x - 32)$$

-180°F ... ${}^{\circ}\text{C}$

```
#include <stdio.h>
```

```
int main()
```

```
{ int lower = -80;  
  int upper = 150;  
  int step = 10;  
  int i = lower;
```

```
for ( ; i <= upper; i += step)
```

```
{ printf("Fahr: %d \t Cels: %f \n", i,
```

$5/9 * (i - 32)$);
 int
 int

#include <stdio.h>

#define LOWER -80

#define UPPER 150

#define STEP 10

x :

out i = LOWER

float

for (; i <= ...)

{ printf("F: %d \t

})
 ?

valtnakat preprocessor
segts segret adjec te
haz ne taglaljan
beljet a meudan

keplet:

maztakat

```
#define FAHR2CELS(x)  
(5/9 * (x + 32))
```

```
int main()
```

```
{ ... }
```

obigen tipasian: 0-out or hi

$5/9 \rightarrow 0$

ha az azin szimat (lebegipontos) fesszit
 \Rightarrow az egész huf is lebegipontos (ez)

%od forward kintor :oda objek oztani
utu

```
char profit(char *s, ...);
```

C++ Földaf

#include <cmath>

inline double fabr2_cels (double x)

{ return 5.9 * (x - 32);
}

int main ()

{

const int lower = -80;

const int upper = 150;

const int step = 10;

@ for (int i=lower; i<=upper; i+=step)

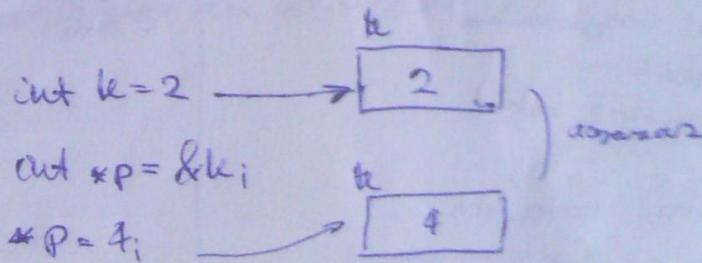
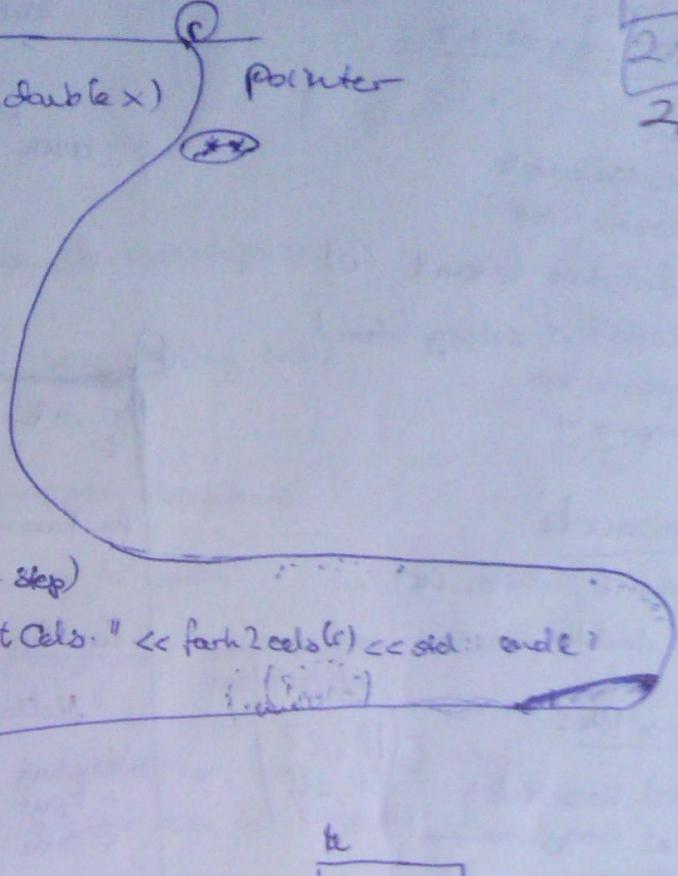
{ std::cout << "Föld" << i << "It Cels." << fabr2_cels(i) << std::endl
};

}

3

** int *p = &step;
↳ amikor*p = 15,
↳ elrendelásint *const ccp = &x;
(*ccp) += 6;

föld : const int * const ccp = & have működ?



const int N=47
 int x=10;
 const int * ccp = &N;
 /* ccp = 14;
 ccp = &x;

3. ctt gyarapítat

3/3

üline := ötlet, rezervált azon! \rightarrow letölteni
- körlektér ajanlás a copy-on-write

összeg:

Class Foo

```
{  
public: int x;  
void set(int i) {  
    x = i;  
}  
};
```

[$\wedge \rightarrow \text{ctrl} + \text{C}$]

Jeladat: Hasoljuk a standard inputot a std::cin karakterláncba másolni!

abcd\n
abcd\n
:
(eof)

\rightarrow abc* : cat \rightarrow ^D \wedge D (ctrl+D) \downarrow
Win/DOS: copy con 12

(nézés) ez visz lezárt körön
rögzít masol

\rightarrow karakterdől kezdődik

```
/* C-ban: */  
#include <stdio.h>  
int main()  
{  
    char ch;  
    while !(fch = getchar() != EOF)  
    {  
        putchar(ch);  
    }
}
```

a+b+2*(n/4),

int a, b;

a=b
- a+=b

(baz eredmény)

if (a=b)

1. a-hez előre elrendelt
adjázz b-t
2. b!=0

{f(b!=0)}

Diff. zw. | 3/5

Class. cast

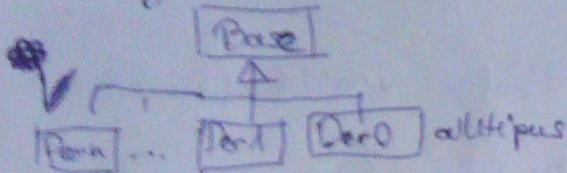
Oft

- static - cast

Static cast: allgemeines calli Gute verwendet

Jahrtaf

- dynamic - cast



Der0's pointer → deinen tipus Letzte mögliche Wahl ist korrekt

Base* bp = new Der0();

②
delete bp;

④ if (Der0* p = dynamic - cast < Der0* (bp))

{}
- //punkt auf Der0
{} mindestens is
erwähnt

void set (int *v, int x)

{ for (int i=0; i < sizeof(v) / sizeof(N[0]); i++)
 { v[i] = x } }

pointer vezető

OppGyár

4.2.1

Probléma: ha a tömb merek bejárás negy minden a pointer meghatározott.



void set (int *v, const int x)

{ for (int i=0; i < 5; i++)
 { v[i] = x; } }

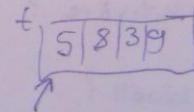
keres

int v[5] = { 0, 1, 3, 7 }

set (N, sizeof(v) / sizeof(N[0]), 0)

{ ezt pointerkort használja, erre oda kell ~;
 figyelni a fu hivatalnál, ha használunk
 akárjuk a meretet }

int t[] = { 5, 8, 3, 9 };
 int *p = t; &t[0]

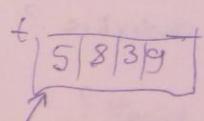


OppGyár 4.2.2

ellenőrözhető a teljes betűk számaval csak az összeg 6

{ ezt pointerkort használja, erre oda kell ~;
 figyelni a fu hivatalnál, ha használunk
 akárjuk a meretet }

int t[] = { 5, 8, 3, 9 };
 int *p = t; &t[0]

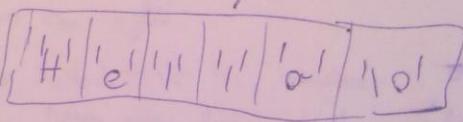


OppGyár 4.2.2

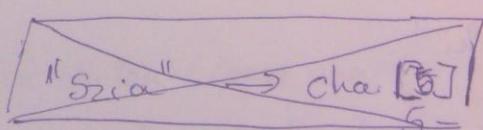
ellenőrözhető a teljes betűk számaval csak az összeg 6

Regi C-s stringezés

char v[5] = "Hello";
 strlen(v) == 5



ezt a 10-ig cs megnövezzük
 hogy elérjük van beine.



"Szia" → const char [5] → C++ ban

char *s = "blabla"; → pointer formázás el C-ba

ez ezt követően "kövesd" volt használni,

C++ →

s[0] = 'x'; → hiba: adott pont. ponttól lefelé pontozott

C++ 4. osztály / 4/3

Pointer aritmetika



t [5 | 8 | 3 | 9]

P ↗

int t[] = {5, 8, 3, 9};

int *p = &t[0];

++p; } ↗ összehasonlítás indexi
p++ ; elemre lepít

p--; } elöl indexi
--p; } előre lepít

*p = 3; → Raktár - többlet
P = 2; Lépni eggyel előre

t[2] = 6; // *(&t[2]) = 6;
Pointer + szám

2[t] = 6; ↗ *(&t[1]) = 6;

{ int *q = t+3

P = t;

p == q; ↗ utgyanarra az indexre mutat-e?

p < q; ↗ p kevésbé indexűre mutat-e?

p > q; ↗ p utgyanra a kevésbé indexűre?

q-p // => 3 Hány elem van a kettő között?

const char str[] = "alma";

std::cout << str[0]; // l

<< str[1]; // l

<< str[2]; // ma → hogys jött el ki?

[a | l | m | a | l | o]
↓

int strlen (char *s)

{

char *p = s;

while (*p != '\0')

{

++p;

}

return p-s;

}

S

↓

[l | s | t | r | i | h | g | (\0)]

megszem hossz

eddig hossz (előtérben?)

Lehetőség leghosszú pozícióhoz (nincs más pont) mi van ha null pointer? Null pontozik nem lehet elérhető!

futás ideje hossz → utazásra meg akartunk!
nagy tömör használásban ellentmondani kellene,

Mi van akkor ha az lepsit is hibaazon?

int strlen (char* s)

$$\{ \text{char} *p = s$$

while (*p != '\0') {
++p;

} return p - s;

str

str

QH 4. gyak

4/4

CPP Gyak

4/4.1

(1) Strlen ("xyz"); → Noha const törtezőtől elérhető a címlak.

Segmentation fault.

(2) char str[] = "abcd"; // →

strlen (str);

std::cout << str;

// ⇒ számmal nem ír. mivel de lekérdezésre

címzésben a [b,c,d]

⇒ negyedik, nem visszatér

[a|b|c|d|\0|]

| s str → felülről az elso char '0' -ra

| mi az (*p = '\0') eredménye

| '\0' → integer ami false értéket ad vissza.

| ⇒ std::cout << str;

↓

megoldás

// ⇒ számmal nem ír. mivel de lekérdezésre

címzésben a [b,c,d]

⇒ negyedik, nem visszatér

CPP Gyak

4/4.1

'\0' → integer ami false értéket ad vissza.

⇒ std::cout << str;

↓

megoldás

while ('\0' != *p) → van ismétel

while ('\0' == *p) → hibát ír ki, mivel

nem jön leírni hibás betű, hanem konstansval
használtsági össze, ami a (bal oldalra)
lehetőségek sorában harossul írjuk

CPP Gyak

4/4.2

int strlen (const char*s)

{ const char *p = s;

while (*p != '\0')

→ Noha konstansot olvashatunk itt.

{ ++p;

} return p - s;

}

Karakter konverzid

char conv (char ch) :

t	→	g
A	→	B
K	→	m
...		

OppGau2
5.1.1

- faktoriális időtű opt.
- memória használat opt.
- karbantartásra szánt optimalizálás

1) char conv (char ch)

{ static const char c[] = { ... };
return c [ch & 0xFF]; }

Σ 0, 1, 2, ..., k-1, k
m | ↗ ↘ ↗ ↘ ↗ ↘
[ch & 0xFF]

Futási idő-c

Static: Cokoládás várhatóan hosszú lesz

az legalább negatívakor leírásban az általános törököléshez következőképpen végez a futási haneur megmarad.

OppGau2
5.1.2

return c [ch & 0xFF]; }

[ch & 0xFF]

Futási idő-c

Static: Cokoládás várhatóan hosszú lesz

az legalább negatívakor leírásban az általános törököléshez következőképpen végez a futási haneur megmarad.

OppGau2
5.1.2

0xFF → Integer 255

F	F
0..0 1111 1111	

8bit pozitív bináris leírás

char → signed char c[128, 127]

→ a signed char

→ az adott gépen előforduló a földelések, ha a most aktuálisan meghagyottak, de nem földelések

0..0 1111 1111 1111 1111 1111 1111 1111 [0, 255]

0..0	Ch..	Ch..	Ch..
↓	↓	↓	↓

new negatív

Befelvételi es művelet
fej

ch & 255 - összehasonlítás be $\frac{0377}{255} = 255 = 0xFF$

char conv (char ch)

{ switch (ch)

{ case 't': return 'g'; break;

:

default: return ch;

{ if (ch == ch)

return 'g';

else if ('h' == ch)

{

...}

Ungewöhnlich „geprißt“ gewaltsam (e
rsatz eingesetzte Werte hin)

Cgal Cpp
5.2.1

// Java compiler kann angeben, dass hier
// ein return mit einem alten Wert vorgenommen wird

Was passiert hier genau?

HelloWorld!

Hello Upper
Hello

char conv (char ch)

{ static const char from [] = { 't', 'h', 'k' };

static const char to [] = { 'g', 'B', 'm' };

for (int i=0; i < sizeof (from) / sizeof (from []) ; ++i)

 return to [i];

 return ch;

}

Cgal Cpp
5.2.2

char conv (char ch)

{ static const pair c [] =

 { { 't', 'g' }, { 'h', 'B' }, { 'k', 'm' } }

 { for (int i=0; i < sizeof (c) / sizeof (c []) ; ++i)

 if (c [i].from == ch)

 return c [i].to;

 return ch;

}

Struct Data

{ char from,

char to,

Pair(char, char)

{ from = f

to = t,

}

}

char conv(char ch)

{ static const Pat c[3] =

{ Pair('t', 'g'), Pair('h', 'B')

Pair('d', 'k') },

for (int i=0; i<3; i++) {

if (c[i].from == ch)

return c[i].to;

}

Ponto contínuo (versão 2 de struct struktur (types))

Exemplo:

int v[] = {3, 8, 9, 1, 4};

int *p = &v;

int *q = v+3;

p < q

p != q

q - p (= 3)

Elle feltetet:
 q es p ugymaraz
 tömb elnevezése
 ugyanazon

std::cout << p; // Alexander csinál

Iha nem ugyanaz; betűkkel!

de ezek hasonlók, de

szintén hasonlók, de

Definíció a ref. esetén: ~~ezt minden~~ az működési területen az érték változókhoz hozzájárulhatnak.

Pointerek

int x

int *p = &x

p = 0;

nullpoint: minden olyan adattípusban

használható a pontor minden mintet szennyez

vállalható kezdeti értéke.

- sehol semmit.

→ null pontot nem szabad elérni (le farabol, hibás)

integer 0;

NULL → preprocessor minden

nyelvtől függően: minden jelentése

C C++

0 == (void*)0

mashop van abban C-ban os C++ban az ez hibásnak tekinthető.

void f(int)

3

void f (const char *s)

3 ...

f(0);

if (p != 0)
{ *p = 5; }

mint csinál f(0);? Definíció miatt hiba

f(0); → integer

g(NULL); C++ → C-ban nem írható

C++11: nullptr → a pontortól fog megelőzni

g(nullptr);

NULL
(void*)0

↳ bárminyer típusú adatot mutthat
nemcsént pointer aritmetikája

B

Függvények ponterekkel

$\sum_{i=0}^N a_i$

$a_i: N \rightarrow R$
int double

int (*a) (double)

double (*a)(int)

double sum(int N; double(*a)(int))

{
double s=0;
for(int i=0; i<N; ++i)
s+=a(i);
return s;

dyn függvény mintett, minden ha minden pont
paraméterrel, double-t ad vissza.

double f

double fu(int i)

{ return 1.1(i); }

}

std::cout << sum(20, * &fu);

11.7.2021. 11:11
ellen

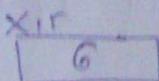
std::cout << sum(20, &fu); → Ellen hozza ki, ez attól

Miért nem működik a fü?

pointer hozzájárhat az objektus rögzítéséhez, de működésben nem működik, mivel az objektus elérhető.

Referenciák

int x = 6;

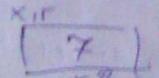


int &x = x;

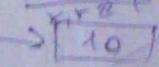
(ír referencia) nem működik elérhető az objektus rögzítéséhez
+ nem foglal fel területet

(megelőzés fájtható objektus)

+ n



5 = 10;



(az n is később működik)

Hagyományos a dologban az objektus

Ilyen a kölönbség a pointer és a referencia között?

Pointer

Referencia

memóriaelérés

megvalósítható, hozzá hozzájárhat az objektus
pointerrel nem hozzájárhat, mivel a pointer



null pointer

~ nincs null referencia

(0x0 -ban van)

* , &

(dereferálás, címkezés)

pointer arithmetik

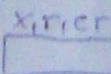
~ nincs erre szükséges művek

~ nincs felirat

(Param. átadáskor jön a referencia)

const int& cr = x;

Pointer arithmetika



→ a hozzájárható terület

cr - eredménytelen (dereferálás) művek

(++cr, cr++) -nem!

