Fogalomlista

by pts@fazekas.hu at Tue Jan 23 19:41:20 CET 2007 -- Tue Jan 23 21:45:53 CET 2007

Ha oktatóként jutottál ide, és jót akarsz a hallgatóknak, akkor az alábbi igen terjedelmes fogalomlistából válaszd ki azokat, amiket jónak látod, hogy a köv. félévben INFO2-b?l a hallgatók heti 45 perc el?adás és 45 perc gyakorlat alatt megtanuljanak. Vélhet?en az alábbiaknak több, mint a felét ki kell húzni id?hiány miatt. De melyik fele maradjon meg? Ha van ötleted, hogy mi maradjon meg, akkor írd a wiki felhasználóneved a megmaradó sorokba, és ne írd a nem megmaradókba.

- algoritmus
  - leállási probléma
- szoftver
  - operációs rendszer
  - alkalmazás
    - interaktív alkalmazás
    - szerver, daemon
    - kötegelt alkalmazás (batch job)
  - firmware (pl. mobiltelefonban, DVD-fevev?ben, videókamerában, gépkocsi elektronikájában)
- RAM-gép
  - processzor
  - memória
  - gépi utasítás
  - regiszter
    - utasításszámláló
    - akkumulátor
    - indexregiszter
    - általános célú regiszter
    - flagek
  - feltétel nélküli elágazás
  - feltételes elágazás
- fordító (compiler)
- interpreter
- szintaktikai elemek
  - kulcsszó
  - megjegyzés
  - konstans
    - egész szám
      - decimális (10-es számrendszerbeli)
      - oktális (8-as számrendszerbeli, 0...)
      - hexadecimális (16-os számrendszerbeli, 0xDeadBeaf)
      - méret jelzése (l, L)
      - el?jel nélküliség jelzése (u)
    - lebeg?pontos szám
      - lebeg?pontos szám tudományos jelöléssel (...e...)
      - méret jelzése (f, l, L)
    - karakter
    - string
      - backslash-es escape
        - \n
        - \r
        - \t
        - bájt beszúrása oktálisan: \0123
        - bájt beszúrása hexadeximálisan: \xFa

- \<soremelés>
- \"http://wiki.math.bme.hu
- \\

- azonosító (identifier)
- operátor
  - prefix operátor
  - postfix operátor
  - infix operátor
    - balra asszociatív infix operátor
    - jobbra asszociatív infix operátor
    - nem asszociatív infix operátor (ilyen C-ben nincs)
  - hármas operátor
    - ?:
  - lusta kiértékelés
    - &&
    - ||
    - ?:
  - operátor-precedencia
    - el?ször a postfix operátorok a tapadóval kezdve
    - majd a prefix operátorok a tapadóval kezdve
    - majd az infix és hármas operátorok, precedencia-sorrendben és az asszociativitásuknak megfelel?en
    - az asszociativitás csak azonos precedenciájú operátorok között számít
  - 3["http://wiki.math.bme.hunyami"http://wiki.math.bme.hu] == "http://wiki.math.bme.hunyami"http://wiki.math.bme.hu[3]

- C preprocesszor direktívák
  - lehet szóköz és megjegyzés a # után
  - folytathatók sorvégi \ után
  - makródefiníció
    - #define
      - példa #define isalpha(c) (((c)|32U)-'a'<='z'-'a'+0U)
    - #undef
  - feltételes fordítás
    - #if egész-kifejezés
    - #else
    - #elseif (nem szabványos)
    - #endif
    - #ifdef
    - #ifndef
  - más fájlokat betölt? preprocesszor-direktívák
    - #include <...>
    - #include "http://wiki.math.bme.hu..."http://wiki.math.bme.hu
    - #line n "http://wiki.math.bme.hu..."http://wiki.math.bme.hu
- egész szám
  - el?jel nélküli egész szám (unsigned)
  - el?jeles egész szám (signed)
  - kettes komplemens ábrázolás
  - egészszám-típusok C-ben
    - char (>=8 bit)
    - unsigned char
    - short (>=16 bit)
    - unsigned short
    - int (>=16 bit)

- ◊ unsigned int == unsigned
- ◊ long (>=32 bit)
- ◊ unsigned long
- ◊ long long (>= 64 bit, nem szabványos)
- ◊ unsigned long long

♦ bitenkénti m?velet
- ◊ és (and, &)
- ◊ vagy (or, |)
- ◊ kizáró vagy (xor, ^)
- ◊ nem (not, ~)
- ◊ balra eltolás (shl, <<)
- ◊ jobbra eltolás (shr, >>)

♦ sizeof operátor C-ben
♦ kerekítés a nulla felé (C-ben)

• logikai érték (bool, boolean)
♦ logikai m?velet
- ◊ és (and, &&)
- ◊ vagy (or, ||)
- ◊ kizáró vagy (xor)
- ◊ nem (not, !)
- ◊ lusta kiértékelés (&&, ||)
- ◊ feltételes kiértékelés (?:)

• lebeg?pontos szám
♦ lebeg?pontos szám el?jele
♦ lebeg?pontos szám törtrésze (mantissza)
♦ lebeg?pontos szám kitev?je (karakterisztika)
♦ normálás
♦ lebeg?pontos számok összeadása
♦ lebeg?pontos számok szorzása
♦ túlcsordulás (Inf, -Inf)
♦ alulcsordulás (+0, -0)
♦ eredmény nincs értelmezve (NaN)
♦ legeb?pontosszám-típusok C-ben
- ◊ float
- ◊ double
- ◊ long double (nem szabványos)

♦ kerekítés
- ◊ kerekítés a plusz végtelen felé
- ◊ kerekítés a mínusz végtelen felé
- ◊ kerekítés a nulla felé
- ◊ kerekítés a legközelebbi ábrázolható érték felé

• karakter
♦ Unicode-karakter wchar_t
♦ 8-bites karakter (char, unsigned char)

• string (karakterlánc)
♦ null-terminált string
♦ hosszával megadott string

• változó
♦ globális változó
♦ lokális változó
- ◊ a lokális változó elrejti a globálisat

♦ ciklusváltozó

• típuskonverzió

- automatikus típuskonverzió
- explicit (kézi) típuskonverzió

- be- és kimenetkezelés C-ben
  - #include <stdio.h>
  - fájlkezelés C-ben
    - FILE*
    - fopen()
    - fclose()
    - stdin
    - stdout
    - stderr
  - kimenet C-ben
    - putchar(), putc()
    - printf(), fprintf()
      - változó argumentumszám
      - formátumstring
        - %d
        - %-5.3g
    - sprintf()
    - fwrite()
  - bemenet C-ben
    - getchar(), getc()
    - fgets()
    - scanf(), fscanf()
      - változóértékek helyett mutatókat vár
    - fread()
- gyors stringkezelés C-ben
  - #include <string.h>
  - strlen()
  - strcpy()
  - strcmp()
  - strncmp()
  - memcpy()
  - memcmp()
- néhány matematikai függvény C-ben
  - #include <math.h>
  - gcc -lm
  - M_PI (nem szabványos)
    - helyette 2*atan2(1,0)
  - osztás nullával
    - egész számok esetén
    - lebeg?pontos számok esetén
  - sqrt()
  - fabs()
    - == összehasonlítás helyett különbség abszolút értékének figyelése
  - abs()
  - cbrt() (mindig ábrázolható az eredménye)
  - sin()
  - cos()
  - tan()
  - atan2()
  - exp()
  - log()

- fpclassify() (POSIX szabványos)
- ceil(), ceill(), ceilf()
- floor(), floorl(), floorf()
- trunc(), truncl(), trncf()
- pow()

- adatszerkezet
  - struktúra (struct, record)
  - tömb (array)
    - indexelés
    - többdimenziós tömb
    - tömb méretének tárolása a tömbön kív?l
    - tömb deklarásása C-ben
    - többdimenziós tömb deklarálása C-ben
    - tömb definiálása C-ben
    - null-terminált karaktertömb definiálása C-ben
    - indexelés 1-t?l (vagy egyéb, nem 0 kezd?értékt?l) C-ben
    - tömb méretének dinamikus növelése
      - realloc()
  - láncolt lista
    - sima, egy irányba láncolt lista
      - NULL
      - lekódolni: minimális elem megkeresése egy láncolt listában
      - lekódolni: minimális elem megkeresése egy rendezett láncolt listában
    - kétirányba körbeláncolt lista
  - verem (LIFO)
    - futási verem (runtime stack)
    - verem megvalósítása tömmbel
    - verem megvalósítása sima, egy irányba láncolt listával
  - várakozási sor (queue, FIFO)
    - várakozási sor megvalósítása kétirányba körbeláncolt listával
  - prioritásos várakozási sor
    - kupac (heap)
      - prioritásos várakozási sor megvalósítása kupaccal
      - kupacrendezés
      - eratoszthenészi szita kupaccal
  - gráf
    - incidencia mátrix
    - adjacencia (szomszédossági) mátrix
    - komponensek számának meghatározása mélységi bejárással
    - legrövidebb út keresése
      - szélességi bejárás
      - Dijkstra algoritmusa
  - asszociatív tömb
    - asszociatív tömb megvalósítása struktúrák körbeláncolt listájával
    - asszociatív tömb megvalósítása keres?fákkal
    - asszociatív tömb megvalósítása hasheléssel
- rendezett tömb
  - bináris keresés rendezett tömbben
    - lekódolni: eldönteni, hogy egy érték eleme-e egy rendezett listának
    - miért nem alkalmazható láncolt lista esetén?
  - stabil rendezés
  - tömb stabil rendezése összefésül? rendezéssel
  - azonos elemek keresése

- ♦ tömbben el?forduló különböz? értékek megszámolása
- egyszer? számelméleti algoritmusok
  - ♦ euklideszi algoritmus legnagyobb közös osztó megkeresésére
    - ◊ törtszám egyszer?sítése
    - ◊ racionális (tört)számok összeadása
  - ♦ átváltás számrendszerek közt leosztogatással
  - ♦ moduláris hatványozás négyzetre emeléssel és szorzással
  - ♦ prímség ellen?rzése négyzetgyökig próbálgatással
  - ♦ prímfelbontás kiírása próbálgatással
  - ♦ eratoszthenészi szita
  - ♦ négyzetgyökvonás bináris kereséssel
  - ♦ négyzetgyökvonás Newton-iterációval
- rekurzió
  - ♦ kett?hatványok számolása
  - ♦ Fibonacci-számok számolása
  - ♦ rekurzió gyorsítása tabellázással
    - ◊ binomiális együtthatók rekurzív számolása
    - ◊ leghosszabb közös részsorozat (részstring) keresése
    - ◊ nyer? stratégia keresése a „csak a szélér?l vehetsz el *játékhoz*
  - ♦ összefésül? rendezés (szerepel máshol is)
- ciklus
  - ♦ ciklusváltozó
  - ♦ a ciklusutasítás részei
    - ◊ kezdeti értékadás
    - ◊ továbbmenési feltétel
    - ◊ leállási feltétel
    - ◊ továbblépési utasítás
    - ◊ ciklusmag
    - ◊ azonnali kilépés (break)
    - ◊ folytatás a következ? iterációval (continue)
  - ♦ elöltesztel?s ciklus
    - ◊ while
    - ◊ for
  - ♦ hátultesztel?s ciklus
    - ◊ do ... while
  - ♦ végtelen ciklus
    - ◊ bizonyítás keresése program leállására
      - · el?rehaladást jelz?, csökken? természetes szám érték? változó keresése
    - ◊ a leállási probléma
  - ♦ példák ciklusra
    - ◊ négyzetszámok kiírása szorzással
    - ◊ négyzetszámok kiírása összeadással
    - ◊ a gép gondol egy egész számot, a játékos tippelhet
    - ◊ a játékos gondol egy egész számot, a gép tippelhet
    - ◊ string helyben megfordítása
    - ◊ fájlnév kiterjesztésének keresése
    - ◊ fájlnév elérési útjának keresése
    - ◊ két rendezett lista összefésülése
    - ◊ bemenet karaktereinek megszámolása
      - · a wc program
- algoritmusok leírása
  - ♦ követelményspecifikáció, interfész
    - ◊ algoritmus neve

- ◊ bemeneti adatok és típusaik
- ◊ kimeneti változók és típusaik
- ◊ hívás el?tt teljesül? feltétel (precondition)
- ◊ hívás után teljesül? feltétel (postcondition)
- ◊ pontos megadása annak, hogy mit csinál (az nem, hogy hogyan)
- ◊ hibalehet?ségek listája
- ◊ hibajelzési mód specifikálása
- ◊ letöltéséhez, lefordításához és futtatásához szükséges információk

♦ a használt adatszerkezetek leírása

♦ a m?ködés leírásának módjai

- ◊ informális leírás (receptszer?en, folyamatos szövegben)
- ◊ precíz formális leírás
  - · folyamatábra (flow chart)
    - az elvégzend? m?veletek leírása
    - a m?veleti sorrend kijelölése
    - UML aktivitás-diagram (activity diagram)
  - · stuktogramm (stuki)
  - · leírás egyéb matematikai formalizmussal
  - · leírás informatikai formalizmussal
  - · a forráskód megadása
- ◊ a m?ködés megértését segít? leírás
  - · segédábra, szemléltet? ábra
  - · UML-diagramok
  - · szimuláció

• mutató

♦ mutatók összehasonlítása

♦ mutató növelése és csökkentése

- ◊ az 1-gyel növelés hatása a mutatóra
- ◊ preinkrementálás (++p)
- ◊ posztinkrementálás (p++)

♦ void*

♦ NULL

• memóriakezelési hibák

♦ túl korai felszabadítás

♦ túl kés?i felszabadítás

♦ elmaradt felszabadítás (memory leak)

♦ többször felszabadítás

♦ túl rövid terület foglalása

- ◊ puffertúlcsordulás (buffer overflow)
- ◊ puffertúlcsordulásos támadás

♦ érvénytelen területre (pl. NULL-ra) való hivatkozás

♦ iniciálizálatlan érték felhasználása

♦ memóriakezelési hibajelenségek

- ◊ nehezen reprodukálható hibák
- ◊ a program értelmetlenül viselkedik
- ◊ a program értelmetlen kimenetet produkál
- ◊ Segmentation fault

• függvény (function) == szubrutin

♦ argumentum == paraméter

♦ argumentum típusa

♦ visszatérési érték

♦ visszatérési érték típusa

♦ függvénytörzs

- paraméterátadás fajtái
  - érték szerinti paraméterátadás
  - név szerinit paraméterátadás == cím szerinti paraméterátadás
    - megvalósítása mutatókkal C-ben
- return mögötte kifejezéssel
- a függvénytörzs végén kötelez? return (kivéve eljárás esetén)
- eljárás (procedure)
  - void visszatérési érték
  - return üresen
- két szomszédos tömbrész megcserélése helybeni megfordítással
  - void visszatérési típus
- több érték visszadása
  - struktúra visszadása
  - visszadás cím szerinti paraméterátadással
- változó argumentumszámú függvények
  - #include <stdarg.h>
  - va_list
  - va_start
  - va_arg
  - va_end
- függvénymutató
  - számok rendezése a qsort() függvénnyel
  - kis- és nagybet?ket figyelmen kívül hagyó rendezés a qsort() függvénnyel
- egymásba ágyazott függvényt C-ben nem lehet

- deklarált függvény, típus, változó honnan érhet? el (storage class)
  - static
  - módosító nélküli
  - extern (opcionális)
  - (auto)
  - (register)
- deklaráció
  - többszörös deklaráció
  - el?re deklaráció (forward declaration)
- definíció C-ben
  - típusdefiníció
    - typedef
  - függvénydefiníció
  - változódefiníció kezdeti értékkel (inicializálással)
- utasításblokk
  - lokális változó
- feltétel nélküli vezérlésátadás
  - goto
  - goto kiküszöbölési lehet?ségei
    - új, kis függvények
    - while, break, continue
- feltételes elágazás
  - if
  - else
  - újabb if az else ágban
  - switch
  - case
- típusdefiníció C-ben
  - egyszer? típuskifejezések

- ◊ pl. char(*)(int*[],unsigned short(*)[42])
- ♦ egyszer? változódeklaráció
  - ◊ pl. char(*f)(int*[],unsigned short(*)[42]);
- ♦ typedef
- ♦ struct
  - ◊ változódeklarációval kombinálva
  - ◊ typedef-fel kombinálva
  - ◊ __attribute__ ((packed)) nélkül extra helyet hagyhat ki, pl. struct { char a; long b; }
- ♦ enum
- ♦ union

• fejlécfájl (header, .h)
- ♦ #include
- ♦ szabványos C fejlécfájlok
  - ◊ ctype.h
  - ◊ (setjmp.h)
  - ◊ math.h
  - ◊ string.h
  - ◊ stdio.h
  - ◊ stdlib.h
  - ◊ time.h

• hibakeresési módszerek
- ♦ szigorúbb fordítási mód használata
  - ◊ gcc -ansi -pedantic
- ♦ szigorúbb fordítási figyelmeztetések kérése
  - ◊ gcc -W -Wall -Wunused
  - ◊ gcc -Wstrict-prototypes -Wmissing-prototypes -Wmissing-declarations
- ♦ okos szövegszerkeszt? használata
  - ◊ szintaxis-színezés (syntax highlighting)
  - ◊ zárójel párjának mutatása (show matching brace)
  - ◊ azonosítók automatikus kiegészítése
- ♦ kiíró utasítások beillesztése
  - ◊ kiíró utasítások feltételessé tétele
- ♦ memóriakezelés ellen?rzése
  - ◊ valgrind
- ♦ nyomkövetés (debug)
  - ◊ utasításléptetés (step)
  - ◊ töréspont (breakpoint)
  - ◊ változófigyelés (watch)
- ♦ meghívott rendszerhívások mutatása
  - ◊ strace

• szkriptnyelv

• memóriakezelés
- ♦ fix memóriaterület használata (pl. TeX)
- ♦ dinamikus memóriakezelés: memóriaterület kérése az operációs rendszert?l
  - ◊ kézi memóriafoglalás és -felszabadítás
    - · malloc()
    - · free()
    - · realloc()
    - · calloc()
  - ◊ kézi memóriafoglalás, automatikus memóriafelszabadítás
    - · hivatkozásszámlálás (reference counting)
      - • körkörös hivatkozások
    - · szemétgy?jtés

- megjelöl és kiseper (mark and sweep) szemétgy?jt?
- többgenerációs (generational) szemétgy?jt?
- másoló szemétgy?jt?

- parnacssori argumentumok és a környezet kezelése C-ben
  - argc
  - argv
  - #include <stdlib.h>
  - environ
  - getenv()
  - putenv()
- Ruby változóérvényesség-fajták (scope)
  - Ruby globális változó
  - Ruby lokális változó
  - tagváltozó (példányváltozó, attribútum)
  - osztályváltozó (statikus változó)
  - Ruby konstans
- modul
  - egymásba ágyazott modulok
- objektum-orientált programozás
  - objektum
  - osztály
  - metódus (tagfüggvény)
    - nem virtuális metódus
    - virtuális metódus
      - virtuállismetódus-tábla
    - absztrakt metódus
  - konstruktor
  - destruktor
    - miért nincs Rubyban destruktor
  - osztálymetódus
  - örökl?dés
    - egyszeres örökl?dés
    - többszörös örökl?dés
    - super
    - a konstruktor nem örökl?dhet
      - szül? osztály konstuktorának hívása
  - interfész
    - interfész többszörös öröklése
  - absztrakt osztály
  - egységbe zárás (encapsulation)
  - információrejtés
    - public
    - protected
    - private
    - friend
  - polimorfizmus
    - virtuálismetódus-hívás
    - függvény többszörös definiálása (function overloading)
    - operátor többszörös definiálása (operator overloading)
  - kompozíció
    - b?víthet? programkód
    - újrafelhasználható programkód
- iterálás

    - iterátor
    - blokk
    - blokkparaméterek
    - yield
- reguláris kifejezés (regular expression, regexp)
    - illesztés reguláris kifejezésre (regexp match)
    - végignézés reguláris kifejezéssel (regexp scan)
    - csere reguláris kifejezéssel (regexp substitution)
    - darabolás reguláris kifejezéssel (regexp split)
- példa: racionális számok osztálya
- példa: vektorok és mátrixok osztálya
    - mátrixszorzás
- kivételkezelés
    - kivétel
    - raise
    - begin ... rescue ... end
    - begin ... ensure ... end
    - Exception#backtrace
    - a kivételkezelés memóriafelszabadítási problémája C++-ban
- kötés ideje
    - korai kötés
    - kés?i kötés
- típusosság foka
    - statikus, méret ismert
    - dinamikus: a változóba bármit bele lehet tölteni (a Ruby ilyen)
    - automatikus: dinamikus, és bármi bármivé konvertálódik (a Ruby nem ilyen)
- típusellen?rzés ideje
    - fordításkor
    - értékadáskor
    - az érték felhasználásakor

TODO:

- Ruby beépített osztályok, p()
- Ruby hatékony programozás
- standard C reference kétoldalas PDF fekete-fehér nyomtatása, kiosztása
- hangadók
- el?adás célcsoportja
- ne üljenek bambán feladatmegoldáskor

Rácz Balázs ötletei:

- cout-ot jobb tanítani, mint a printf-et
- C++ programban hibakeresés legyen
- vector<int> -et tudják használni, ami biztonságosabb, mint a malloc()

pts öteletei:

- ne tanítsuk meg azt, ami kés?bb Algoritmuselméletb?l tananyag lesz
- Cékla házi feladatok kiírása (jól gyakorlható velük a rekurzió)
- ZH-n lehessen használni a standard C reference-t
- ZH-ra összeállíthassanak maguknak egy Ruby-puskát. Aki olyan puskát

tesz fel a wikibe, ami alapján 5-ösre meg lehet írni a Ruby ZH-t, az 5-öst kap a Ruby ZH-jára (be se kell jönnie megírni)

Ötlet: els? ZH C-b?l, második ZH Ruby-b?l