A tárgy f?oldala: SZIMP2 A tárgy oktatásához felhasznált, http://wiki.math.bme.hu/ -n belüli wikioldalak GNU FDL licenc vagy (választás szerint) CC-BY-SA-2.0 licenc szerint szabadon használhatók és terjeszthet?k.

Tartalomjegyzék

8. el?adás (2007-04-06)

Az objektum-orientált programozás elméleti bevezet?je hangzott el, és a matmul feladat megoldása felkerült a táblára Ruby nyelven.

Néhány új fogalom:

- osztály (a C-beli struktúratípusnak felel meg)
- objektum (a C-beli struktúrának felel meg)
- attribútum (a C-beli struktúramez?nek felel meg)
- metódus (a C-beli függvénynek felel meg)

Ha a Ruby osztályhierarchiája hasonló a biológiai törzsfához. Például a gerincesek (osztály) törzsén belül a madarak (osztály) osztályának egy alosztálya a pacsirta (osztály) nev? faj, melynek egy példánya a Csipcsip nev? kismadarunk (objektum), aki nem mellesleg egy pacsirta. ?t jellemezhetjük különböz? tulajdonságai alapján; ezek az attribútumok. Például: él-e még? , mennyire éhes, hogy hívják a párját, stb. Tehát bizonyos tulajdonságait megadjuk, amik csak rá jellemz?ek.

Matmul Ruby nyelven:

```
a=[[1,2],                #megadjuk az 'a' mátrixot
   [3,4],
   [5,6]]
```

```
b=[[7],                  #megadjuk a 'b' mátrixot
   [8]]

ab=[]                    #létrehozzuk az üres szorzatmátrixot
i=0; while i<a.size;     # while ciklust nem véletlenül használunk, mivel Rubyban nincs for ciklus
  ujsor=[]
  j=0; while j<b[0].size;
    x=0
    k=0; while k<b.size
      x+=a[i][k]*b[k][j]
      ujsor[j]=x
   end
 ab[i]=ujsor
end
```

9. gyakorlat (2007-04-10, 2007-04-13)

lepcso.rb:

```
n=7
l=[[1],
   [2,3],
   [4,5,6],
   ...
  ]
# l.size==n
p l
```

9. el?adás (2007-04-13)

Objektum-orientáltan kezdtünk programozni Ruby nyelven, a Sikidom, BBox és Kor osztályok kerültek fel a táblára. Megismertük a szemétgy?jtést (garbage collection) is. Láttunk példát rá, hogy két változó mutathat ugyanarra az objektumra.

```
class BBox
  attr_accessor :llx, :lly, :urx, :ury
end

class Sikidom
  def bbox()
    fail  # még nem tudjuk megírni, a Sikidom túl absztrakt
  end
  def kerulet()
    fail  # még nem tudjuk megírni, a Sikidom túl absztrakt
  end
  def terulet()
    fail  # még nem tudjuk megírni, a Sikidom túl absztrakt
  end
end

class Kor <Sikidom
  attr_accessor :cx, :cy, :r
  def bbox()
    b=BBox.new
    b.llx=@cx-@r; b.lly=@cy-@r
    b.urx=@cx+@r; b.ury=@cy+@r
    return b  # a return fölösleges
  end
  def kerulet()
    Math::PI*2*@r
```

```
  end
  def terulet()
    Math::PI*@r*@r
  end
end
```

A szemétgy?jt? kipucolja a felesleges dolgokat a memóriából. A el?ször mindent lenulláz, azután pedig akire van mutató egy elérhet? helyr?l, azt megnöveli. Ha két szemét egymásra mutat, akkor egyiküket sem növeli, mert egymásból ugyan elérhet?ek de ?k maguk nem érhet?ek el egy elérhet? helyr?l. A szemétgy?jtés alatt a program lelassul.

A szemétgy?jtés szemléltetése egy példában:

```
class Gyerek                                        #létrehozunk egy Gyerek osztályt
  attr_accessor :nev, :kov                          #az elemei névvel(nev) és egy mutatóval(kov) fog
  def initialize(nev) @nev=nev; @kov=nil; end       #definiáljuk a konstruktort, amely egy nevet fog
end

def egyik()                                         #definiáljuk az egyik metódust
  pal=Gyerek.new("http://wiki.math.bme.huPal"http://wiki.math.bme.hu)                                  #p
  kata=Gyerek.new("http://wiki.math.bme.huKata"http://wiki.math.bme.hu)
  peter=Gyerek.new("http://wiki.math.bme.huPeter"http://wiki.math.bme.hu)
  pal.kov=kata; kata.kov=peter; peter.kov=pal       #létrehozzuk a mutatókat a csoporbeli viszonyok
end

def masik()
  sandor=Gyerek.new("http://wiki.math.bme.huSandor"http://wiki.math.bme.hu)
  jozsef=Gyerek.new("http://wiki.math.bme.huJozsef"http://wiki.math.bme.hu)
  benedek=Gyerek.new("http://wiki.math.bme.huBenedek"http://wiki.math.bme.hu)
  sandor.kov=jozsef; jozsef.kov=benedek
end

a=egyik
b=masik
# itt
a=nil
```

Ha a szemétgy?jtés *#itt* megy végbe, akkor a következ? történik:

Csak azok maradnak meg, akik ekkor az *a* és a *b* változókból valamilyen úton elérhet?k. Az *a* változó ekkor az *egyik* metódus utoljára használt elemére (objektumára) fog mutatni. (A mi esetünkben Pálra.) Pál, Kata és Péter egy ciklikus csoportot alkotnak, tehát bármelyikükb?l bármelyikük elérhet?, így az 'a' változóból is. Ezért a szemétgy?jtés során, egyik kis barátunknak sem fog baja esni. A *b* változó a fent leírtak alapján Benedekre fog mutatni. De Sándor csak Józsefre, József csak Benedekre mutat, más viszony nem áll fent a csoportban. Így a *b* változóból **CSAK** Benedek érhet? el, így a szemétgy?jtés során Sándortól és Józseft?l sajnos meg kell válnunk. Még akkor is, ha Sándorból József elérhet?, hiszen itt egy szemét mutat egy másik szemétre. (Sándortól és Józseft?l egyben elnézést is kérünk a kifejezésért!)

10. gyakorlat (2007-04-17, 2007-04-20)

A Szórt Spektrum a Háztet?n Csillagjós és Rádióamat?r Klub holnap Guinness-rekordkísérletre készül: annyi ?rturistával kívánják felvenni a kapcsolatot egy nap alatt, amennyivel csak lehet. Az internetr?l letöltötték a holnapi ?rturista-áthaladási id?pontokat. Minden áthaladáshoz három adat tartozik: az ?rturista neve (szóközt nem tartalmaz), mikortól (hány órától) kezdve tartózkodik a klub rádiójának vételkörzetében, és mikor

hagyja el a vételkörzetet. E két id?pont egy balról zárt, jobbról nyílt intervallum végpontjai. Példa áthaladási id?pontok (~/szimp2/urturista1.in):

```
Pirx_pilota 5 7
Leia_hercegno 7 9
Dalek_42 6 10
Kis_herceg 1 6
```

Írjon programot Ruby nyelven, ami beolvassa az áthaladási id?pontokat, és kiválaszt a benne lev? ?rturisták közül egy olyan maximális részhalmazt, melynek bármely két eleméhez tartozó intervallumok diszjunktak. (A diszjunktságot azért kell kikötni, mert a klubnak csak egy rádiója van, és szerencsétlen esetben el?fordulhat, hogy a teljes intervallumban próbálkozniuk kell, hogy végre összejöjjön a kapcsolat.)

Ötlet: Használja ki, hogy ha az intervallumok végpont szerint növekv? sorrendbe vannak rendezve, akkor a mohó algoritmus jó és optimális eredményt ad. A mohó algoritmus veszi sorra az intervallumokat, és egy intervallumot akkor választ ki, ha az ? metszete az utoljára kiválasztott intervallummal üres halmaz.

Mintamegoldás (Ivett és Viktor, ~/szimp2/urturista.rb):

```
class Intervallum
  attr_accessor :nev, :a, :b
end

p=[]
STDIN.each_line { |s|
  t=s.split(/\s+/)
  h=Intervallum.new
  h.nev=t[0]
  h.a=t[1].to_i
  h.b=t[2].to_i

  #p[p.size]=h
  #p.push(h)
  p << h
}

p.sort!{|ha,hb| ha.b<=>hb.b}

q=[]
h0=nil
p.each { |h|
  if (h0==nil || [h.a,h0.a].max >= [h.b,h0.b].min);
    q.push(h)
    h0=h
  end
}
q.each { |h|
  STDOUT.puts h.nev
}
```

10. el?adás (2007-04-20)

Megtanultuk:

- konstruktor (initialize)
- példányváltozó (@)
- metódus, amely beállít egy példányváltozót (attr_writer)
- metódus, amely visszaad egy példányváltozót (attr_reader)

- fekete doboz

11. gyakorlat (2007-04-24, 2007-04-27)

A keddi gyakorlat sportnap miatt elmaradt.

A gyakorlatot Bácsi László, LacKac (e-mail: lackac@) tartotta stringfeldolgozásról. Ezt a programot (~lackac/ruby/backup.rb) magyarázta:

```
require 'net/http'
require 'iconv'

class String
 def to_latin2
    Iconv.iconv('iso-8859-2', 'utf-8', self)[0]
  end
end

class Feed
  def initialize url
    @url = url
    @items = []
  end

  def get_content
    @content = Net::HTTP.get(URI.parse(@url))
  end

  def parse
    get_content
    @content.gsub(/<item.*?>.*?<\/item>/m) do |match|
      item = {}
      match =~ /<title>(<!\[CDATA\[)?(.*?)(\]\]>)?<\/title>/m
      item[:title] = $2
      match =~ /<link>(.*?)<\/link>/m
      item[:link] = $1
      match =~ /<description>(<!\[CDATA\[)?(.*?)(\]\]>)?<\/description>/m
      item[:desc] = $2
      @items << item
    end
  end

  def parse_by_xml
    require 'rexml/document'
    get_content
    xml = REXML::Document.new(@content)
    xml.elements.each('//item') do |match|
      item = {}
      item[:title] = match.elements['title'].text
      item[:link]  = match.elements['link'].text
      item[:desc]  = match.elements['description'].text
      @items << item
    end
  end

  def list
    @items.each do |item|
      puts item[:title].to_latin2
      puts item[:link].to_latin2
      l=0
      item[:desc].split.each do |w|
        print "http://wiki.math.bme.hu  "http://wiki.math.bme.hu if l == 0
```

```
        print w.to_latin2+"http://wiki.math.bme.hu "http://wiki.math.bme.hu
        l+=w.length+1
        if l > 60
          print "http://wiki.math.bme.hu\n"http://wiki.math.bme.hu
          l = 0
        end
      end
      print "http://wiki.math.bme.hu\n"http://wiki.math.bme.hu
    end
  end
end

feed = Feed.new(ARGV[0])
feed.parse_by_xml
feed.list
```

Így próbáltuk:

```
$ ruby backup.rb 'http://24ora.index.hu/?rss&keres=&szerzokeres=&rovatkeres=osszes&order=1'
$ ruby backup.rb 'http://origo.hu/contentpartner/rss/hircentrum/origo.xml'
```

11. el?adás (2007-04-27)

Az el?adást Bácsi László, LacKac (e-mail: lackac@) tartotta stringfeldolgozásról. Az IRB-be ezt írta be:

```
$ irb
irb(main):001:0>
irb(main):002:0*
irb(main):003:0* n=42
=> 42
irb(main):004:0> n.class
=> Fixnum
irb(main):005:0> f=3.14
=> 3.14
irb(main):006:0> f.class
=> Float
irb(main):007:0> s="http://wiki.math.bme.hualma"http://wiki.math.bme.hu
=> "http://wiki.math.bme.hualma"http://wiki.math.bme.hu
irb(main):008:0> s.class
=> String
irb(main):009:0>
irb(main):010:0*
irb(main):011:0*
irb(main):012:0* s.methods
=> ["http://wiki.math.bme.humethods"http://wiki.math.bme.hu, "http://wiki.math.bme.huinstance_eval
"http://wiki.math.bme.hu, "http://wiki.math.bme.hustrip"http://wiki.math.bme.hu, "http://wiki.math
, "http://wiki.math.bme.husucc!"http://wiki.math.bme.hu, "http://wiki.math.bme.huinstance_of?"http
t"http://wiki.math.bme.hu, "http://wiki.math.bme.hufind_all"http://wiki.math.bme.hu, "http://wiki.
leton_methods"http://wiki.math.bme.hu, "http://wiki.math.bme.hutr"http://wiki.math.bme.hu, "http:/
ip"http://wiki.math.bme.hu, "http://wiki.math.bme.hufrozen?"http://wiki.math.bme.hu, "http://wiki.
?"http://wiki.math.bme.hu, "http://wiki.math.bme.hucapitalize!"http://wiki.math.bme.hu, "http://wi
gsub"http://wiki.math.bme.hu, "http://wiki.math.bme.huprotected_methods"http://wiki.math.bme.hu, "
p"http://wiki.math.bme.hu, "http://wiki.math.bme.hurstrip"http://wiki.math.bme.hu, "http://wiki.ma
"http://wiki.math.bme.huswapcase!"http://wiki.math.bme.hu, "http://wiki.math.bme.huljust"http://wi
"http://wiki.math.bme.huclass"http://wiki.math.bme.hu, "http://wiki.math.bme.huobject_id"http://wi
ods"http://wiki.math.bme.hu, "http://wiki.math.bme.hu=="http://wiki.math.bme.hu, "http://wiki.math
 "http://wiki.math.bme.huuntaint"http://wiki.math.bme.hu, "http://wiki.math.bme.husucc"http://wiki
nd"http://wiki.math.bme.hu, "http://wiki.math.bme.huindex"http://wiki.math.bme.hu, "http://wiki.ma
 "http://wiki.math.bme.huchomp"http://wiki.math.bme.hu, "http://wiki.math.bme.hu=~"http://wiki.mat
ypt"http://wiki.math.bme.hu, "http://wiki.math.bme.hudelete!"http://wiki.math.bme.hu, "http://wiki
to_f"http://wiki.math.bme.hu]
irb(main):013:0> s+"http://wiki.math.bme.hu korte"http://wiki.math.bme.hu
```

```
=> "http://wiki.math.bme.hualma korte"http://wiki.math.bme.hu
irb(main):014:0> s
=> "http://wiki.math.bme.hualma"http://wiki.math.bme.hu
irb(main):015:0> s.methods.class
=> Array
irb(main):016:0> s.methods.find_all {|m| m =~ /case/}
=> ["http://wiki.math.bme.hudowncase"http://wiki.math.bme.hu, "http://wiki.math.bme.hudowncase!"ht
irb(main):017:0>
irb(main):018:0*
irb(main):019:0*
irb(main):020:0* a=[1,2,3,4,5,6]
=> [1, 2, 3, 4, 5, 6]
irb(main):021:0> a.clas
NoMethodError: undefined method `clas' for [1, 2, 3, 4, 5, 6]:Array
        from (irb):21
        from :0
irb(main):022:0> a.class
=> Array
irb(main):023:0> a.methods
=> ["http://wiki.math.bme.humethods"http://wiki.math.bme.hu, "http://wiki.math.bme.huinstance_eval
, "http://wiki.math.bme.hupack"http://wiki.math.bme.hu, "http://wiki.math.bme.hureverse_each"http:
assoc"http://wiki.math.bme.hu, "http://wiki.math.bme.huinstance_of?"http://wiki.math.bme.hu, "http
 "http://wiki.math.bme.hureject!"http://wiki.math.bme.hu, "http://wiki.math.bme.huflatten"http://w
e"http://wiki.math.bme.hu, "http://wiki.math.bme.hureplace"http://wiki.math.bme.hu, "http://wiki.m
 "http://wiki.math.bme.hufrozen?"http://wiki.math.bme.hu, "http://wiki.math.bme.huinstance_variabl
lect"http://wiki.math.bme.hu, "http://wiki.math.bme.huto_a"http://wiki.math.bme.hu, "http://wiki.m
tion"http://wiki.math.bme.hu, "http://wiki.math.bme.huindices"http://wiki.math.bme.hu, "http://wik
_to?"http://wiki.math.bme.hu, "http://wiki.math.bme.hureject"http://wiki.math.bme.hu, "http://wiki
nsert"http://wiki.math.bme.hu, "http://wiki.math.bme.huobject_id"http://wiki.math.bme.hu, "http://
_"http://wiki.math.bme.hu, "http://wiki.math.bme.humember?"http://wiki.math.bme.hu, "http://wiki.m
, "http://wiki.math.bme.hulast"http://wiki.math.bme.hu, "http://wiki.math.bme.husend"http://wiki.m
=~"http://wiki.math.bme.hu, "http://wiki.math.bme.hueach_index"http://wiki.math.bme.hu, "http://wi
to_ary"http://wiki.math.bme.hu, "http://wiki.math.bme.huequal?"http://wiki.math.bme.hu, "http://wi
irb(main):024:0> n.size
=> 4
irb(main):025:0> a.size
=> 6
irb(main):026:0> a.nitems
=> 6
irb(main):027:0> n
=> 42
irb(main):028:0>
irb(main):029:0*
irb(main):030:0* a=[1,2,3,4]
=> [1, 2, 3, 4]
irb(main):031:0> b=["http://wiki.math.bme.huegy"http://wiki.math.bme.hu,"http://wiki.math.bme.huke
irb(main):032:1"http://wiki.math.bme.hu "http://wiki.math.bme.hu
irb(main):033:1> b=["http://wiki.math.bme.huegy"http://wiki.math.bme.hu,"http://wiki.math.bme.huke
irb(main):034:1> 10000000.size
irb(main):035:1> 10_000_000.size
irb(main):036:1> "http://wiki.math.bme.hu
irb(main):037:1"http://wiki.math.bme.hu "http://wiki.math.bme.hu
irb(main):038:1>
irb(main):039:1*
irb(main):040:1*
irb(main):041:1* n=10_000_000_000
irb(main):042:1> n.class
irb(main):043:1> SyntaxError: compile error
(irb):31: parse error, unexpected tSTRING_BEG, expecting kDO or '{' or '('
b=["http://wiki.math.bme.huegy"http://wiki.math.bme.hu,"http://wiki.math.bme.huketto"http://wiki.m
                    ^
        from (irb):42
        from (null):0
LacKac@LacKacMB:~> irb
```

```
irb(main):001:0> n=10_000_000_000
=> 10000000000
irb(main):002:0> n.size
=> 8
irb(main):003:0> n=1_000_000_000
=> 1000000000
irb(main):004:0> n.size
=> 4
irb(main):005:0> a=[1,2,3,4]
=> [1, 2, 3, 4]
irb(main):006:0> b=["http://wiki.math.bme.huegy"http://wiki.math.bme.hu,"http://wiki.math.bme.huke
=> ["http://wiki.math.bme.huegy"http://wiki.math.bme.hu, "http://wiki.math.bme.huketto"http://wiki
irb(main):007:0> a+b
=> [1, 2, 3, 4, "http://wiki.math.bme.huegy"http://wiki.math.bme.hu, "http://wiki.math.bme.huketto
irb(main):008:0> [1,2,3]+[3,4,5]
=> [1, 2, 3, 3, 4, 5]
irb(main):009:0> [1,2,3]|[3,4,5]
=> [1, 2, 3, 4, 5]
irb(main):010:0> [1,2,3]-[3,4,5]
=> [1, 2]
irb(main):011:0> [1,2,3]^[3,4,5]
NoMethodError: undefined method `^' for [1, 2, 3]:Array
        from (irb):11
        from :0
irb(main):012:0> [1,2,3]&[3,4,5]
=> [3]
irb(main):013:0> [1,2,3]*3
=> [1, 2, 3, 1, 2, 3, 1, 2, 3]
irb(main):014:0> [1,2,3]==[1,2,3]
=> true
irb(main):015:0> [1,2,3]==[2,1,3]
=> false
irb(main):016:0> a=[1,2,3,4,5]
=> [1, 2, 3, 4, 5]
irb(main):017:0> a[0]
=> 1
irb(main):018:0> a[4]
=> 5
irb(main):019:0> a[-1]
=> 5
irb(main):020:0> a[-3]
=> 3
irb(main):021:0> a[-3]="http://wiki.math.bme.huharom"http://wiki.math.bme.hu
=> "http://wiki.math.bme.huharom"http://wiki.math.bme.hu
irb(main):022:0> a
=> [1, 2, "http://wiki.math.bme.huharom"http://wiki.math.bme.hu, 4, 5]
irb(main):023:0> a[23423]
=> nil
irb(main):024:0> a.sort
ArgumentError: comparison of Fixnum with String failed
        from (irb):24:in `sort'
        from (irb):24
        from :0
irb(main):025:0> a.sort_by^C
irb(main):025:0> a=[3,6,2,89,42
irb(main):026:1> ]
=> [3, 6, 2, 89, 42]
irb(main):027:0> a=[3,6,2,89,42]
=> [3, 6, 2, 89, 42]
irb(main):028:0> a.sort
=> [2, 3, 6, 42, 89]
irb(main):029:0> b
=> ["http://wiki.math.bme.huegy"http://wiki.math.bme.hu, "http://wiki.math.bme.huketto"http://wiki
irb(main):030:0> b.sort
```

```
=> ["http://wiki.math.bme.huegy"http://wiki.math.bme.hu, "http://wiki.math.bme.huharom"http://wiki
irb(main):031:0> c=a.map! {|i| i.to_s}
=> ["http://wiki.math.bme.hu3"http://wiki.math.bme.hu, "http://wiki.math.bme.hu6"http://wiki.math.
irb(main):032:0> a.has?(4)
NoMethodError: undefined method `has?' for ["http://wiki.math.bme.hu3"http://wiki.math.bme.hu, "ht
        from (irb):32
        from :0
irb(main):033:0> a.include?(4)
=> false
irb(main):034:0> a.include?(89)
=> false
irb(main):035:0> a
=> ["http://wiki.math.bme.hu3"http://wiki.math.bme.hu, "http://wiki.math.bme.hu6"http://wiki.math.
irb(main):036:0> a.include?("http://wiki.math.bme.hu89"http://wiki.math.bme.hu)
=> true
irb(main):037:0> a.sort
=> ["http://wiki.math.bme.hu2"http://wiki.math.bme.hu, "http://wiki.math.bme.hu3"http://wiki.math.
irb(main):038:0> a.sort_by {|i| i.to_i}
=> ["http://wiki.math.bme.hu2"http://wiki.math.bme.hu, "http://wiki.math.bme.hu3"http://wiki.math.
irb(main):039:0> a
=> ["http://wiki.math.bme.hu3"http://wiki.math.bme.hu, "http://wiki.math.bme.hu6"http://wiki.math.
irb(main):040:0> b
=> ["http://wiki.math.bme.huegy"http://wiki.math.bme.hu, "http://wiki.math.bme.huketto"http://wiki
irb(main):041:0> b=["http://wiki.math.bme.hualma"http://wiki.math.bme.hu,"http://wiki.math.bme.huk
=> ["http://wiki.math.bme.hualma"http://wiki.math.bme.hu, "http://wiki.math.bme.hukorte"http://wik
irb(main):042:0> b.sort_by {|s| s.length}
=> ["http://wiki.math.bme.hudio"http://wiki.math.bme.hu, "http://wiki.math.bme.hualma"http://wiki.
irb(main):043:0> b.sort_by {|valami| valami.length}
=> ["http://wiki.math.bme.hudio"http://wiki.math.bme.hu, "http://wiki.math.bme.hualma"http://wiki.
irb(main):044:0> s="http://wiki.math.bme.hualma korte dio"http://wiki.math.bme.hu
=> "http://wiki.math.bme.hualma korte dio"http://wiki.math.bme.hu
irb(main):045:0> s.length
=> 14
irb(main):046:0> s[6]
=> 111
irb(main):047:0> s[6].class
=> Fixnum
irb(main):048:0> s[6] == "http://wiki.math.bme.hua"http://wiki.math.bme.hu
=> false
irb(main):049:0> s[6] == "http://wiki.math.bme.huk"http://wiki.math.bme.hu
=> false
irb(main):050:0> s[6] == ?k
=> false
irb(main):051:0> ?k
=> 107
irb(main):052:0> ?o
=> 111
irb(main):053:0> s[6] == ?o
=> true
irb(main):054:0>
irb(main):055:0* s
=> "http://wiki.math.bme.hualma korte dio"http://wiki.math.bme.hu
irb(main):056:0> s.split
=> ["http://wiki.math.bme.hualma"http://wiki.math.bme.hu, "http://wiki.math.bme.hukorte"http://wik
irb(main):057:0> mac="http://wiki.math.bme.hu16:ef:3d:3a:56:dc"http://wiki.math.bme.hu
=> "http://wiki.math.bme.hu16:ef:3d:3a:56:dc"http://wiki.math.bme.hu
irb(main):058:0> mac.split("http://wiki.math.bme.hu:"http://wiki.math.bme.hu)
=> ["http://wiki.math.bme.hu16"http://wiki.math.bme.hu, "http://wiki.math.bme.huef"http://wiki.mat
irb(main):059:0> mac="http://wiki.math.bme.hu16:ef 3d.3a:56.dc"http://wiki.math.bme.hu
=> "http://wiki.math.bme.hu16:ef 3d.3a:56.dc"http://wiki.math.bme.hu
irb(main):060:0> mac.split("http://wiki.math.bme.hu:"http://wiki.math.bme.hu)
=> ["http://wiki.math.bme.hu16"http://wiki.math.bme.hu, "http://wiki.math.bme.huef 3d.3a"http://wi
irb(main):061:0> mac.split("http://wiki.math.bme.hu:. "http://wiki.math.bme.hu)
=> ["http://wiki.math.bme.hu16:ef 3d.3a:56.dc"http://wiki.math.bme.hu]
```

```
irb(main):062:0> mac.split(/[:\. ]/)
=> ["http://wiki.math.bme.hu16"http://wiki.math.bme.hu, "http://wiki.math.bme.huef"http://wiki.mat
irb(main):063:0> "http://wiki.math.bme.hualma"http://wiki.math.bme.hu+"http://wiki.math.bme.hukort
=> "http://wiki.math.bme.hualmakorte"http://wiki.math.bme.hu
irb(main):064:0> "http://wiki.math.bme.hualma"http://wiki.math.bme.hu+"http://wiki.math.bme.hu "ht
=> "http://wiki.math.bme.hualma korte"http://wiki.math.bme.hu
irb(main):065:0> 'alma'
=> "http://wiki.math.bme.hualma"http://wiki.math.bme.hu
irb(main):066:0> print "http://wiki.math.bme.hualma\n"http://wiki.math.bme.hu
alma
=> nil
irb(main):067:0> print "http://wiki.math.bme.hualma"http://wiki.math.bme.hu
alma=> nil
irb(main):068:0> print "http://wiki.math.bme.hualma\n"http://wiki.math.bme.hu
alma
=> nil
irb(main):069:0> print "http://wiki.math.bme.hualma\tkorte\tdio\n"http://wiki.math.bme.hu
alma    korte   dio
=> nil
irb(main):070:0> print 'alma\tkorte\tdio\n'
alma\tkorte\tdio\n=> nil
irb(main):071:0>
irb(main):072:0* n=42
=> 42
irb(main):073:0> s="http://wiki.math.bme.huegy"http://wiki.math.bme.hu
=> "http://wiki.math.bme.huegy"http://wiki.math.bme.hu
irb(main):074:0> s="http://wiki.math.bme.huertelme"http://wiki.math.bme.hu
=> "http://wiki.math.bme.huertelme"http://wiki.math.bme.hu
irb(main):075:0> print "http://wiki.math.bme.huaz elet "http://wiki.math.bme.hu+s+"http://wiki.mat
SyntaxError: compile error
(irb):75: parse error, unexpected tIDENTIFIER, expecting $
        from (irb):75
        from :0
irb(main):076:0> print "http://wiki.math.bme.huaz elet "http://wiki.math.bme.hu+s+"http://wiki.mat
TypeError: can't convert Fixnum into String
        from (irb):76:in `+'
        from (irb):76
        from :0
irb(main):077:0> print "http://wiki.math.bme.huaz elet "http://wiki.math.bme.hu+s+"http://wiki.mat
az elet ertelme 42=> nil
irb(main):078:0> print "http://wiki.math.bme.huaz elet "http://wiki.math.bme.hu+s+"http://wiki.mat
az elet ertelme 42
=> nil
irb(main):079:0> print "http://wiki.math.bme.huaz elet #{s} #{n}\n"http://wiki.math.bme.hu
az elet ertelme 42
=> nil
irb(main):080:0> print "http://wiki.math.bme.huaz elet #{s.upcase} #{n}\n"http://wiki.math.bme.hu
az elet ERTELME 42
=> nil
irb(main):081:0>
irb(main):082:0*
irb(main):083:0*
irb(main):084:0* s.upcase
=> "http://wiki.math.bme.huERTELME"http://wiki.math.bme.hu
irb(main):085:0> s="http://wiki.math.bme.huLacKac"http://wiki.math.bme.hu
=> "http://wiki.math.bme.huLacKac"http://wiki.math.bme.hu
irb(main):086:0> s.upcase
=> "http://wiki.math.bme.huLACKAC"http://wiki.math.bme.hu
irb(main):087:0> s.downcase
=> "http://wiki.math.bme.hulackac"http://wiki.math.bme.hu
irb(main):088:0> s.swapcase
=> "http://wiki.math.bme.hulACkAC"http://wiki.math.bme.hu
irb(main):089:0> s
=> "http://wiki.math.bme.huLacKac"http://wiki.math.bme.hu
```

```
irb(main):090:0> s.upcase!
=> "http://wiki.math.bme.huLACKAC"http://wiki.math.bme.hu
irb(main):091:0> s
=> "http://wiki.math.bme.huLACKAC"http://wiki.math.bme.hu
irb(main):092:0> s.methods
=> ["http://wiki.math.bme.humethods"http://wiki.math.bme.hu, "http://wiki.math.bme.huinstance_eval
"http://wiki.math.bme.hu, "http://wiki.math.bme.hustrip"http://wiki.math.bme.hu, "http://wiki.math
, "http://wiki.math.bme.husucc!"http://wiki.math.bme.hu, "http://wiki.math.bme.huinstance_of?"http
t"http://wiki.math.bme.hu, "http://wiki.math.bme.hufind_all"http://wiki.math.bme.hu, "http://wiki.
leton_methods"http://wiki.math.bme.hu, "http://wiki.math.bme.hutr"http://wiki.math.bme.hu, "http:/
ip"http://wiki.math.bme.hu, "http://wiki.math.bme.hufrozen?"http://wiki.math.bme.hu, "http://wiki.
?"http://wiki.math.bme.hu, "http://wiki.math.bme.hucapitalize!"http://wiki.math.bme.hu, "http://wi
gsub"http://wiki.math.bme.hu, "http://wiki.math.bme.huprotected_methods"http://wiki.math.bme.hu, "
p"http://wiki.math.bme.hu, "http://wiki.math.bme.hurstrip"http://wiki.math.bme.hu, "http://wiki.ma
"http://wiki.math.bme.huswapcase!"http://wiki.math.bme.hu, "http://wiki.math.bme.huljust"http://wi
"http://wiki.math.bme.huclass"http://wiki.math.bme.hu, "http://wiki.math.bme.huobject_id"http://wi
ods"http://wiki.math.bme.hu, "http://wiki.math.bme.hu=="http://wiki.math.bme.hu, "http://wiki.math
 "http://wiki.math.bme.huuntaint"http://wiki.math.bme.hu, "http://wiki.math.bme.husucc"http://wiki
nd"http://wiki.math.bme.hu, "http://wiki.math.bme.huindex"http://wiki.math.bme.hu, "http://wiki.ma
 "http://wiki.math.bme.huchomp"http://wiki.math.bme.hu, "http://wiki.math.bme.hu=~"http://wiki.mat
ypt"http://wiki.math.bme.hu, "http://wiki.math.bme.hudelete!"http://wiki.math.bme.hu, "http://wiki
to_f"http://wiki.math.bme.hu]
irb(main):093:0> s.reverse
=> "http://wiki.math.bme.huCAKCAL"http://wiki.math.bme.hu
irb(main):094:0> a
=> ["http://wiki.math.bme.hu3"http://wiki.math.bme.hu, "http://wiki.math.bme.hu6"http://wiki.math.
irb(main):095:0> a.reverse
=> ["http://wiki.math.bme.hu42"http://wiki.math.bme.hu, "http://wiki.math.bme.hu89"http://wiki.mat
irb(main):096:0> "http://wiki.math.bme.hubacsi laszlo"http://wiki.math.bme.hu.capitalize
=> "http://wiki.math.bme.huBacsi laszlo"http://wiki.math.bme.hu
irb(main):097:0> nev = "http://wiki.math.bme.hubacsi laszlo istvan"http://wiki.math.bme.hu
=> "http://wiki.math.bme.hubacsi laszlo istvan"http://wiki.math.bme.hu
irb(main):098:0> nev.splite
NoMethodError: undefined method `splite' for "http://wiki.math.bme.hubacsi laszlo istvan"http://wi
        from (irb):98
        from :0
irb(main):099:0> nev.split
=> ["http://wiki.math.bme.hubacsi"http://wiki.math.bme.hu, "http://wiki.math.bme.hulaszlo"http://w
irb(main):100:0> nev.split.map {|i| i.capitalize}
=> ["http://wiki.math.bme.huBacsi"http://wiki.math.bme.hu, "http://wiki.math.bme.huLaszlo"http://w
irb(main):101:0> nev.split.map {|i| i.capitalize}.join
=> "http://wiki.math.bme.huBacsiLaszloIstvan"http://wiki.math.bme.hu
irb(main):102:0> nev.split.map {|i| i.capitalize}.join("http://wiki.math.bme.hu "http://wiki.math.
=> "http://wiki.math.bme.huBacsi Laszlo Istvan"http://wiki.math.bme.hu
irb(main):103:0> a=["http://wiki.math.bme.hualma"http://wiki.math.bme.hu, "http://wiki.math.bme.hu
=> ["http://wiki.math.bme.hualma"http://wiki.math.bme.hu, "http://wiki.math.bme.hukorte"http://wik
irb(main):104:0> a.each {|i| puts i}
alma
korte
szilva
=> ["http://wiki.math.bme.hualma"http://wiki.math.bme.hu, "http://wiki.math.bme.hukorte"http://wik
irb(main):105:0> a.each_with_index {|v,i| puts "http://wiki.math.bme.hu#{i}: #{v}"http://wiki.math
0: alma
1: korte
2: szilva
=> ["http://wiki.math.bme.hualma"http://wiki.math.bme.hu, "http://wiki.math.bme.hukorte"http://wik
irb(main):106:0> a.each_with_index {|v,i| print "http://wiki.math.bme.hu#{i}: #{v}"http://wiki.mat
0: alma1: korte2: szilva=> ["http://wiki.math.bme.hualma"http://wiki.math.bme.hu, "http://wiki.mat
irb(main):107:0> a.each_with_index {|v,i| print "http://wiki.math.bme.hu#{i}: #{v}\n"http://wiki.m
0: alma
1: korte
2: szilva
=> ["http://wiki.math.bme.hualma"http://wiki.math.bme.hu, "http://wiki.math.bme.hukorte"http://wik
irb(main):108:0> s="http://wiki.math.bme.hualma\nkorte\nszilva"http://wiki.math.bme.hu
```

```
=> "http://wiki.math.bme.hualma\nkorte\nszilva"http://wiki.math.bme.hu
irb(main):109:0> print s
alma
korte
szilva=> nil
irb(main):110:0> s.each_line {|l| puts l}
alma
korte
szilva
=> "http://wiki.math.bme.hualma\nkorte\nszilva"http://wiki.math.bme.hu
irb(main):111:0>
irb(main):112:0*
irb(main):113:0*
irb(main):114:0* s
=> "http://wiki.math.bme.hualma\nkorte\nszilva"http://wiki.math.bme.hu
irb(main):115:0> s="http://wiki.math.bme.hualma korte szilva"http://wiki.math.bme.hu
=> "http://wiki.math.bme.hualma korte szilva"http://wiki.math.bme.hu
irb(main):116:0> s.split
=> ["http://wiki.math.bme.hualma"http://wiki.math.bme.hu, "http://wiki.math.bme.hukorte"http://wik
irb(main):117:0> a=["http://wiki.math.bme.hualma"http://wiki.math.bme.hu, "http://wiki.math.bme.hu
irb(main):118:1"http://wiki.math.bme.hu ["http://wiki.math.bme.hu]
SyntaxError: compile error
(irb):117: parse error, unexpected tSTRING_BEG, expecting kDO or '{' or '('
a=["http://wiki.math.bme.hualma"http://wiki.math.bme.hu, "http://wiki.math.bme.hukorte"http://wiki
                           ^
(irb):118: parse error, unexpected ']', expecting $
        from (irb):118
        from :0
irb(main):119:0> a=["http://wiki.math.bme.hualma"http://wiki.math.bme.hu, "http://wiki.math.bme.hu
=> ["http://wiki.math.bme.hualma"http://wiki.math.bme.hu, "http://wiki.math.bme.hukorte"http://wik
irb(main):120:0> a= %w{ alma korte szilva }
=> ["http://wiki.math.bme.hualma"http://wiki.math.bme.hu, "http://wiki.math.bme.hukorte"http://wik
irb(main):121:0> a= %w{alma korte szilva}
=> ["http://wiki.math.bme.hualma"http://wiki.math.bme.hu, "http://wiki.math.bme.hukorte"http://wik
```

12. gyakorlat (2007-05-01, 2007-05-04)

A keddi, május elsejei gyakorlat az ünnep miatt elmaradt. Énekszó és tánc köszöntse. A pénteki gyakorlat anyaga következik.

A tanult C beépített függvények:

- printf()
- scanf()
- putchar()
- getchar()
- abort()

A tanult Ruby beépített osztályok:

- Object
- Float
- Fixnum
- Bignum
- Integer
- String
- Array
- Module

- Class
- TrueClass
- FalseClass
- NilClass
- Hash
- Regexp

Próbálkozások:

```
$ irb
1.2.class
42.class
42.class.class
42.class.class.class
2**42.class  # hiba
(2**42).class
(2**61).class
(2**62).class
"http://wiki.math.bme.hu"http://wiki.math.bme.hu.class
.class
[].class
{}.class
42.class.superclass
42.class.superclass.superclass
42.class.superclass.superclass.superclass
42.kind_of?(Fixnum)   #: true
42.kind_of?(Integer)  #: true
42.kind_of?(String)   #: false
true.class
false.lass
nil.class
Float.superclass
String.methods
```

Az összes beépített osztály listázása (pl. *String*):

```
Object.constants.sort.select { |c| Object.const_get(c).kind_of?(Class) }
```

Az összes beépített modul listázása (pl. *Math*):

```
Object.constants.sort.select { |c| Object.const_get(c).kind_of?(Module) &&
  !Object.const_get(c).kind_of?(Class) }
```

Osztályhiearachia-részlet:

```
Object
  String
  Array
  Hash
  Regexp
  Numeric
    Float
    Integer
      Fixnum
      Bignum
```

A program (*listastszamol.rb*), amely megszámolja, hogy az aktuális felhasználó levelesládájában a mat06 levelezési listára ki hány levelet küldött, és a küldések számának növekv? sorrendjében nyomtatja ki a feladó e-mail címeket:

```
r=/^List-Id: *mat06[.]lists[.]math[.]bme[.]hu$/
# p r.class #: Regexp
f=File.open(ENV["http://wiki.math.bme.huMAIL"http://wiki.math.bme.hu])
listas=false; fejlecben=false
froml=nil
h={}
f.each_line { |l|
  if fejlecben && l =~ r; then
    listas=true
  end
  if fejlecben && l =~ /^From: *(.*)/
    froml=$1
    if froml=~/<(.*?)>/;
      froml=$1
    end
    if froml !~/[@]/;
      froml=nil
    end
  end
  if fejlecben && l =~ /^$/;
    if listas and froml;
      if h.has_key?(froml);
        h[froml]+=1
      else
        h[froml]=1
      end
      #p froml
    end
    froml=nil; listas=false; fejlecben=false
  end
  if l =~ /^From /;
    fejlecben=true
  end
}

h.sort { |a,b|
  a[1]<=>b[1]
}.each { |k,v|
  print "http://wiki.math.bme.hu#{v} #{k}\n"http://wiki.math.bme.hu
}
```

12. el?adás (2007-05-04)

Kivételkezelésr?l volt szó.

```
begin
  f=File.open("http://wiki.math.bme.hubableves.txt"http://wiki.math.bme.hu)
  p f.readline
  p f.readline
rescue Errno::ENOENT
  print "http://wiki.math.bme.huA fájl nem létezik!\n"http://wiki.math.bme.hu
rescue SystemCallError => e
  print "http://wiki.math.bme.huRendszerhiba\n"http://wiki.math.bme.hu
  p e
ensure
  f.close if f && !f.closed?
end
```

Osztályhiererchia-részlet:

```
 Object
   Exception
```

```
    StandardError
      SystemCallError
        Errno::ENOENT
```

Saját kivétel definiálása:

```
class NemPozitivHiba <Exception; end

def feldolgoz(sor)
  f=sor.to_f
  raise NemPozitivHiba, "http://wiki.math.bme.huvalami üzenet"http://wiki.math.bme.hu if f<=0.0
  print "http://wiki.math.bme.hu#{Math::sqrt(f)}\n"http://wiki.math.bme.hu
end

STDIN.each_line { |l|
  begin
    feldolgoz(l)
  rescue NemPozitivHiba
    print "http://wiki.math.bme.huNEM POZITIV\n"http://wiki.math.bme.hu
  end
}
```

13. gyakorlat (2007-05-08, 2007-05-11)

Ezen a pénteken volt a ZH. A gyakorlaton a ZH-feladatokhoz hasonló feladatokat oldottunk meg.

13. el?adás (2007-05-11)

A gyors prímtényez?kre bontás algoritmusa volt. Pontosabban: egy olyan algoritmus hangzott el, ami egy *j* jegy? pozitív egész számnak polinomid?ben (O(*j*^*k*) db alapm?velettel) meghatározza egy valódi osztóját, vagy visszaadja, hogy a szám prím. Sajnos az algoritmus a részletszámítások során az eredeti számnál jóval többjegy? számokkal végez m?veleteket, és mivel nagy számokkal lassú számolni, ezért hiába végez kevés m?veletet, összességében lassú lesz, legalábbis a mai számítógépeken polinomid?nél lassabb lesz.

Jó tanácsok

(Ezt a szakaszt Szabó Viktor vitte fel részben saját órai jegyzetei, részben pts-sel folytatott levelezése alapján.)

Az óra második felében különböz? tanácsok hangzottak el, melyek hasznosak lehetnek az egyetemi tanulmányok során és kés?bb, a munkavégzéskor is:

- CARPE DIEM - Élj a mának!
- SAPERE AUDEM - Merj gondolkodni!
- A munkát mindenképpen el kell végezni; rosszul bárki el tudja. -

Miért ne végeznénk jól? (Görög Ibolya tanácsa a Protokoll c. könyvéb?l.)

- Minden munka az ember javára válik.
- A munka nemesít.
- Tedd közkinccsé munkád eredményét!
- ?Az én vezérem bels?mb?l vezérel? (József Attila)
- Ha nem megy, a legjobbtól kérj segítséget!
- Fogadd el a segítséget, és köszönd is meg!
- Adni öröm. -- Segíts másoknak!

- Lehet?leg úgy segíts, hogy a másik ezt felhasználva egyedül

boldoguljon! (pts által optimálisnak tartott eset: kb. x id? segítség, utána x id? önálló munka.)

- Ne várj viszonzást!
- ?Hiába döngetek kaput, falat?? (Ady Endre)
- Szánd rá az id?t!
- Rossz munkához is id? kell.

Szoftverjogok

A különböz? szoftverjogokról is szó esett: arról, hogy mennyire jogos és erkölcsös egy ötletet, találmányt vagy felfedezést szabadalmaztatni, s hogy mennyire akadályozza ez a tudomány fejl?dését.

Egy középkori szerzetes példáját hallhattuk, aki a repülés területén jelent?s felfedezést tett (több mint 100 métert sikerült repülnie), ám kísérletezés közben egyik alkalommal eltörte a lábát, és ezért az apát eltiltotta a további kutatástól valamint az eddigi tudás továbbadásától. A repülés során szerzett tapasztalatok az apátság iratai közé kerültek, melyek évszázadokig nem kerültek publikálásra. Ha publikálták volna ?ket, akkor az emberiség sokkal hamarabb építhetett volna repül?gépet.

Mit tegyünk, hogy termékünket más is szabadon felhasználhassa?

(A következ? szakasz pts emailjéb?l származik:)

Minden szellemi termék szerz?jét megilleti a szerz?i jog, ezen felül egyes szellemi termékekhez (pl. találmányok) lehet kölön kérni szabadalmi oltalmat is. Mind a szerz?i jog, mind a szabadalmak akadályozzák, hogy a szellemi termék sok emberhez gyorsan eljusson. Ha tehát a szerz? lehet?vé kívánja tenni a gyors és szabad publikálást, akkor az alábbiakat érdemes tennie:

- A szellemi termékhez szabad felhasználást lehet?vé tev? licenct mellékel.

(Ha nem mellékel semmit, az azt jelenti, hogy ?minden jog fenntartva?, vagyis minden másoláshoz és terjesztéshez a szerz? engedélyét kell kérni, ami lassítja a publikálást.) Általában nem kell saját licenct kitalálni, többtucat jól meggondolt, létez? licenc van, egyszer?bb, ha a szerz? ezek közül választ, és feltünteti egy mondatban, melyiket választotta.

- ♦ Szoftverekhez tipikus a GNU GPL és a BSD licenc.

Az utóbbi nagyobb szabadságot ad a felhasználónak, viszont kevésbé biztosítja, hogy ne lehessen a szerz?t kisemmizni a szellemi termék eltulajdonításával. A fentieken kívül többtucat szabad szoftverlicenc van. Azok közül érdemes választani, amelyet az OSI (Open Source Initiative) jóváhagyott (approved).

- ♦ Szövegalapú szellemi alkotásokhoz a CC (Creative Commons) vagy a

GNU FDL (Free Documentation License) ajánlott.

- ♦ Egyéb m?vészeti alkotásokhoz (pl. festmény, zene, szobor) a CC ajánlott.
- ♦ Példamondatok licencelésre:
  - ◊ „Ez a szoftver/fájl a **GNU GPL** szerint terjeszthet?.?
  - ◊ „Ezt a fájl a **BSD licenc** szerint lehet terjeszteni.?
  - ◊ „Ez a dokumentum a **GNU FDL** szerint terjeszthet?.?
  - ◊ „Ezt a (bármit) a **CC** (creative commons) szabályai szerint

lehet terjeszteni.?

- Nem kér szabadalmi oltalmat a termékre.
- Könnyen hozzáférhet?vé teszi, melynek a legjobb módja, hogy a webr?l

letölthet?vé teszi. Ha nincs pénze honlap üzemeltetésére, használhatja az ingyenes szolgáltatókat, például szabad szoftverek fejlesztését és publikálását segít? oldal a http://www.sourceforge.net/

- A megértéshez és módosításhoz szükséges információt (szoftver esetén a

forráskód és a fordítást intéz? szkriptek, épület esetén a részletes tervek) is elérhet?vé teszi, módosítható formátumban is (tehát pl. nem PDF-ben).

- A részletes dokumentációt is elérhet?vé teszi.

A szerz?i jog és a szabadalom

A szabadalom és a szerz?i jog közti különbség megvilágítására tekintsük az alábbi példát. Az X szoftverben van egy ikon, amelyre ha rákattintunk, történik valami, és ha duplán kattintunk, valami más történik.

Szabad-e a szoftvert lemásolni, és akárhány gépen egyszerre használni? A válaszhoz utána kell nézni, hogy a licencben a szerz? milyen felhasználást engedélyezett. (Ha a szerz? nem mellékelt licenct, akkor ?minden jog fenntartva?, az általános jogszabályok az irányadók.) Kereskedelmi szoftvernél általában másolni és több gépen egyszerre használni tilos, szabad szoftvernél általában korlátozás nélkül szabad. (Szabad szoftvernek nevezzük azt a szoftvert, amelynek licence megfelel néhány kritériumnak, sz?kebb értelemben pedig a licencét az OSI jóváhagyta, mint szabad szoftver licenc.)

Szabad-e írni egy ugyanolyan vagy ekvivalens szoftvert? Általában igen, de pl. ha az ikon kinézetét képpontról képpontra le szeretnénk másolni, akkor a licencben utána kell nézni, szabad-e. Szabad szoftvereknél általában szabad, kereskedelmieknél általában nem szabad. Tegyük fel, hogy a szoftverben a duplakattintás funkciót szabadalmi oltalom védi. Ebben az esetben senki más nem írhat olyan szoftvert, ami kezeli a duplakattintást (!). Ez abszurd (a szoftverszabadalmak értelmetlenek, csak néhány nagy szoftvercég gazdagodását szolgálják, az ipar és a tudomány fejl?dését pedig gátolják), de jogilag így van. Egy csomó tömörítési algoritmust (pl. a hang- és videótömörítésben) és fájlformátumot véd szabadalom. Érdemes megfigyelni az egyes PDF-nézeget?k, kép- vagy videószerkeszt?k indulásakor megjelen? ?splash screen?-eket, melyek felsorolják a program által használt szabadalmakat.

14. gyakorlat (2007-05-15, 2007-05-18)

A terv az volt, hogy nyomkövetésr?l és hibakeresésr?l lesz szó mind C, mind Ruby nyelven. printf, kdevelop, valgrind, strace.

Ezek közül megvalósítottuk a hibakeresést C nyelven printf és valgrind segítségével, továbbá volt szó a veremkialakításról C-ben, a puffertúlcsordulásos hibákról és kihasználásukról.

Mi a hiba az alábbi C programban? ~/szimp2/fibjavit0.c:

```
#include <stdio.h>

int fib(int n) {
  printf("http://wiki.math.bme.huFIB %d\n"http://wiki.math.bme.hu, n);
  if (n==0) return 0;
  return fib(n-1, t)+fib(n-2);
}
```

```
int main(void) {
  printf("http://wiki.math.bme.hu%d\n"http://wiki.math.bme.hu, fib(10));
  return 0;
}
```

Mi a hiba az alábbi C programban? ~/szimp2/fibjavit1.c:

```
#include <stdio.h>

int fib(int n, int *t) {
  int f;
  if (n<2) f=n;
  else f=fib(n-1, t)+fib(n-2, t);
  t[n]=f;
  return f;
}

int main(void) {
  int t[2];
  printf("http://wiki.math.bme.hu%d\n"http://wiki.math.bme.hu, fib(10, &t[0]));
  return 0;
}
```

Mi a hiba az alábbi C programban? ~/szimp2/fibjavit2.c:

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

int fib(int n, int *t) {
  int i;
  t[0]=0;
  if (n<=0) return t[0];
  t[1]=1;
  if (n<=1) return t[1];
  for (i=2;i<=n;i++) {
    t[i]=t[i-1]+t[i-2];
  }
  return t[n];
}

int main(void) {
  int *t=malloc(100001);
  printf("http://wiki.math.bme.hu%d\n"http://wiki.math.bme.hu, fib(100000, &t[0]));
  free(t);
  return 0;
}
```

Tekintsük az alábbi C programot (~/szimp2/fibt2.c):

```
#include <stdio.h>

int fib(int n, int *t) {
  int f;
  if (n<2) f=n;
  else f=fib(n-1, t)+fib(n-2, t);
  t[n]=f;
  return f;
}

int main(void) {
  int t[2];
  printf("http://wiki.math.bme.huHello, World! %d\n"http://wiki.math.bme.hu, fib(22, t));
```

```
  return 0;
}
```

Kipróbálás:

```
$ gcc -s -O2 -W -Wall -o fibt2 fibt2.c
$ ./fibt2
17711
Szegmens hiba
```

Kiírja a helyes eredményt (17711), majd szegmens hiba lesz. Azért, mert a t tömb csak 2 elem?, és 22 elemet rak bele a fib. Az extra elemek felülírják a main() visszatérési címét.

Indulás után egy kicsivel így néz ki a verem:

```
 fib(2) visszatérési címe
 n: 2
--------------------------
 fib(3) visszatérési címe
 n: 3
--------------------------
 ...
--------------------------
 fib(22) visszatérési címe
 n: 22
--------------------------
 t[0]==0
 t[1]==1
 main() visszatérési címe
--------------------------
```

Mikor a fib-ek visszatérnek, így néz ki a verem:

```
 t[0]==0
 t[1]==1
 t[2]==1 (a main() visszatérési címe felülírva)
 t[3]==2
 ...
 t[22]==17711
```

Ezután a main() már nem tud visszatérni, mert a visszatérési címét felülírták.

Próbáljuk valgrinddel:

```
$ gcc -g -W -Wall -o fibt2g fibt2.c
$ ./valgrind ./fibt2g
...
Hello, World! 17711
==29070== Jump to the invalid address stated on the next line
==29070==    at 0xD00000008: ???
==29070==  Address 0xD00000008 is not stack'd, malloc'd or (recently) free'd
==29070==
==29070== Process terminating with default action of signal 11 (SIGSEGV)
==29070==  Bad permissions for mapped region at address 0xD00000008
==29070==    at 0xD00000008: ???
...
```

A valgrind se mondja meg, hol a hiba (stack trace), mert a visszatérési cím alapján jelezne.

14. el?adás (2007-05-18)

Mivel a március 15. el?tti szombaton plusz egy el?adást tartottunk, ezért a félévben egy el?adást ki kell hagyni. Ez lesz az. Pontosabban: az el?adás meg lesz tartva, és továbbmegyünk az anyagban, de nem kötelez? eljönni. Ennek az el?adásnak az anyaga nem lesz számonkérve (nem is lehet, mivel a ZH-t az el?z? héten már megírtuk). Az el?adás el?tti gyakorlat természetesen kötelez?.

A második ZH

Id?pontok:

- elmaradt keddi gyakorlatok pótlása: 2006. május 8. kedd, 12:00--14:15.
- konzultáció: 2007. május 10. csütörtök, 14:14--19:19. H.27. A konzultáció korábban is véget érhet a meghirdetettnél: ha 15 percen belül nem hangzik el kérdés.
- ZH papíron: 2007. május 11. péntek, 12:00--14:00. K.mf.65. Felügyel: Sisak Áron. Javítja: Sisak Áron.
- ZH programozás (opcionális): 2007. május 11. péntek, 14:00--16:00, H.57. Felügyel: Sisak Áron.
- A pótZH 7 nappal a ZH után lesz, a napon belül azonos id?beosztás szerint. Terem: J.208.
- A gyakIV 14 nappal a ZH után lesz, lásd a saját fejezetében.

A ZH témája objektum-orientált programozás Ruby nyelven. A ZH-n 1 db, kézzel, kék tollal írt A4-es lapnyi puskát lehet használni. A ZH-n kb. 50% programozási és 50% elméleti feladat lesz. Aki a programozási feladatokat számítógéppel szeretné írni, az jelentkezzen e-mailben a tárgy el?adójánál. A számítógépes megoldás lehet?ségét technikai okokból csak abban az esetben tudjuk biztosítani, ha max. 10 jelentkez? lesz.

Mely beépített metódusokat kell ismerni

A tanult Ruby beépített osztályok és metódusok:

- Object
  - ♦ class
  - ♦ methods
- Float
  - ♦ abs + - * < <= == != >= > <=> **
- Fixnum
  - ♦ abs + - * < <= == != >= > <=> << >> **
- Bignum
  - ♦ (semmit)
- Integer
  - ♦ (semmit)
- String
  - ♦ + =~ !~ split split! chomp chomp! reverse $1 $2 <=> upcase upcase! downcase downcase! reverse reverse!
- Array
  - ♦ [] []= + - | & join each size << push max min find_all sort sort! sort_by sort_by! include?
- Module
  - ♦ (semmit)
- Class
  - ♦ superclass
- TrueClass
  - ♦ && || ! == !=
- FalseClass

        - ♦ && || ! == !=
    - NilClass
        - ♦ == !=
    - Hash
        - ♦ [] []= has_key? keys values size delete each_pair
    - Regexp
        - ♦ (semmit)
    - Math
        - ♦ sqrt PI

Van még: STDIN.each_line, p, print, puts.

Hogy a fentiek mit csinálnak, annak itt lehet utánanézni: http://www.ruby-doc.org/core/

Mintafeladatok

Az alábbiakhoz hasonló és másféle feladatok is várhatók a második ZH-n.

Elméleti feladatok:

- Az `x.foo()` hívás esetén mely osztályokban és milyen sorrendben keresi a Ruby a meghívandó metódust?
- Hozzon létre egy olyan *x* objektumot, melyre `x.kind_of?(Alma) && x.class!=Alma`
- Mikor áll a C-beli `int i=1; while (i!=0) i*=2;` ciklus? Mikor áll le a Ruby-beli `i=1; while i!=0; i*=2; end` ciklus?
- Írjon egy olyan osztályt Ruby nyelven, amely paraméteres konstruktorral rendelkezik, és hozzon létre két példányt. Hányszor hívódik meg a konstruktor?
- Mutasson példát arra, hogy az `x.leszed()` hívás nem a *Gyumolcs* osztályon belül definiált *leszed* metódust hívja, noha `x.kind_of?(Gyumolcs)`
- Mit ad vissza a `self` egy metóduson belül?
- Mi a különbség egy metóduson belül a `foo=5`, `@foo=5` és a `self.foo=5` értékadások között?
- Írjon egy olyan metódust, melyben a `p foo` hívás kétszer szerepel, de csak az egyik esetben írja ki a *foo* nev? lokális változó értékét, a másik esetben mást ír ki.
- Írja át az alábbi kódot úgy, hogy `@a<=@b` mindig teljesüljön. Ha a hívó olyan helyzetet akarna el?idézni, amely megsérti a fenti feltételt, akkor dobjon kivételt `fail`-lel. (Segítség: 3 db metódusba kell `fail`-t tenni.) mintamegoldás

```
class Intervallum
  attr_accessor :a, :b
  def initialize(a, b)
    @a=a; @b=b
  end
end
```

- Az alábbi kód végrehajtása után a a Ruby lefuttatja a szemétgy?jtést. Rajzolja fel az `# itt` sor végrehajtásakor az objektumokat, és azt, hogy az *a* és *b* változók mely objektumra mutatnak. Mely objektumok maradnak meg, és melyeket szabadítja fel a szemétgy?jtés?

```
class Gyerek
  attr_accessor :nev, :kov
  def initialize(nev) @nev=nev; @kov=nil; end
end

def egyik()
  pal=Gyerek.new("http://wiki.math.bme.huPal"http://wiki.math.bme.hu)
```

```
  kata=Gyerek.new("http://wiki.math.bme.huKata"http://wiki.math.bme.hu)
  peter=Gyerek.new("http://wiki.math.bme.huPeter"http://wiki.math.bme.hu)
  pal.kov=kata; kata.kov=peter; peter.kov=pal
end

def masik()
  sandor=Gyerek.new("http://wiki.math.bme.huSandor"http://wiki.math.bme.hu)
  jozsef=Gyerek.new("http://wiki.math.bme.huJozsef"http://wiki.math.bme.hu)
  benedek=Gyerek.new("http://wiki.math.bme.huBenedek"http://wiki.math.bme.hu)
  sandor.kov=jozsef; jozsef.kov=benedek
end

a=egyik
b=masik
# itt
a=nil
```

Programozási feladatok:

- Írjon Ruby-programot *sorhosszcsokkeno.rb* néven, amely beolvassa a bemenet sorait, majd a sorhossz szerinti csökken? sorrendben kiírja a sorokat. Az egyenl? hosszúságú sorok sorrendje tetsz?leges.
- Írjon Ruby-programot *abszoluterteknovekvo.rb* néven, amely beolvassa a bemenet sorait (a sorok egész számokat tartalmaznak), és kiírja a sorokat az abszolút értékek növekv? sorrendjében. Az egyenl? abszolútérték? számok sorrendje tetsz?leges.
- Írjon Ruby-programot *palindrom.rb* néven, amely beolvassa a bemenet sorait, és kiírja a palindrom sorokat. Egy string palindrom, ha a megfordítása önmaga. Segítség: beolvasás után, de még összehasonlítás el?tt az `s.chomp!` hívással törölje a soremelést a string végér?l.
- Írjon Ruby-programot *palindromlehet.rb* néven, amely beolvassa a bemenet sorait, és kiírja azokat a sorokat, melyekben a bet?k átvarálásával palindrom string készíthet?. Egy string palindrom, ha a megfordítása önmaga. 1. segítség: beolvasás után, de még összehasonlítás el?tt az `s.chomp!` hívással törölje a soremelést a string végér?l. 2. segítség: rendezze a string karaktereit az `s=s.split().sort.join` hívással. Mintamegoldás:

```
def palindromlehet(s)
  s=s.chomp.split(//).sort.join
  # i db karaktert vizsgáltunk meg
  # a megvizsgált karakterek küzül az utolsó j db azonos
  i=1; j=1; k=0;
  while i<s.size
    if (s[i]==s[i-1])
      j+=1
    else
      if (j%2==1)
        k+=1
      end
      j=1
    end
    i+=1
  end
  if (j%2==1)
    k+=1
  end
  k<2
end

STDIN.each_line { |s|
  if palindromlehet(s)
    print s
  end
```

```
}
```

- Írjon Ruby-programot *rendszamtobbszor.rb* néven, amely beolvassa a bemenet sorait, és kiírja azokat a rendszámokat, melyek 2-szer vagy többször szerepelnek. A rendszám három bet?b?l, egy köt?jelb?l és három számjegyb?l áll. Segítség: egy sort szavakra lehet bontani a `s.split` hívással, ami stringekb?l álló tömböt ad vissza.
- Írjon Ruby-programot *egyszer.rb* néven, amely beolvassa a bemenet sorait, és csak azokat a sorokat írja ki (tetsz?leges sorrendben), melyek pontosan egyszer szerepelnek.
- Melyik az a legkisebb kett?hatvány, ami már nem *Fixnum*? Írjon programot (*marnemfixnum.rb* néven), ami megadja a választ.
- Adott két objektum: *a* és *b*. Írjon Ruby-függvényt, amely visszaadja azt a legmélyebb osztályt, amely mindkét objektumnak szül?osztálya. Deklaráció: `def kozosos(a,b)`
- Készítsen egy *Teglalap* nev? osztályt Ruby nyelven a *teglalap.rb* fájlba, amely egy tengélypárhuzamos téglalapot tartalmaz. Létrehozására példa: `t=Tegalalap.new(3,4,1,2)`, ahol a bal alsó sarok a (3,4), a szélessége 1, a magassága 2, tehát a jobb fels? sarok a (4,6).
    - Tegye lehet?vé a széls? koordináták lekérdezését, például `p t.llx, t.lly` a bal alsó sarkot `p t.urx, t.ury` jobb fels? sarkot írja ki.
    - Tegye lehet?vé a téglalap méretének lekérdezését, például `p t.width, t.height` a szélességet, majd a magasságot írja ki.
    - Tegye lehet?vé a téglalap méretének megváltoztatását, például `p.width=42` a téglalap szélességét, a `p.height=137` pedig a téglalap magasságát változtatja meg. A változtatás a bal alsó sarkot helybenhagyja, a többi csúcsot pedig értelemszer?en mozgatja.
    - Tegye lehet?vé az egyes csúcspontok lekérdezését: `t.ul` a bal fels?, `t.ur` a jobb fels?, `t.lr` a jobb alsó, `t.ll` a bal alsó sarkot adja vissza. Egy csúcspont egy kételem? tömb: `[x,y]`.
    - Tegye lehet?vé a téglalap eltolását: a `t.eltol!(dx,dy)` hívás *dx*-et hozzáad minden pont *x* koordintájához, és *dy*-ot adjon hozzá minden pont *y* koordinátájához, a `t.eltol(dx,dy)` hívás pedig egy új téglalapot adjon vissza, amely *t*-hez képest (dx,dy)-nal van eltolva.
    - Tegye lehet?vé a súlypont lekérdezését: a `t.sulypont` hívás adja vissza a súlypontot `[cx,cy]` tömbként, a `t.cx` hívás adja vissza a súlypont x koordinátáját, a `t.cy` hívás pedig adja vissza a súlypont y koordinátáját.
    - Tegye lehet?vé a téglalap eltolását a súlypont mozgatásával. A `t.cx=42` hívás x irányban, a `t.cy=137` hívás pedig irányban mozgassa a téglalapot úgy, hogy a súlypont a megadott koordinátára kerüljön.
    - Tegye lehet?vé a téglalap nagyítását: a `t.nagyit!(m)` hívás a tömegközéppontból *m*-szeresére nagyítsa a téglalapot, a `t.nagyit(m)` hívás pedig egy új téglalapot adjon vissza, amely az eredetihez képest a fenti módon van kinagyítva.
    - A téglalap méretét megváltoztató `t.width=()` és `t.height=()` metódusokat módosítsa úgy, hogy negatív méretet ne lehessen megadni. Definiáljon egy kivételosztályt *NegativMeretHiba* néven, és dobja ezt a kivételt, ha a hívó negatív méretet akar beállítani.
    - A téglalap létrehozásakor meghívott függvényt módosítsa úgy, hogy negatív méret? téglalapot ne lehessen megadni. Dobjon *NegativMeretHiba* kivételt, ha a hívó negatív méret? téglalapot akar létrehozni.
    - Tegye lehet?vé a téglalap széls? koordinátáinak megváltoztatását. A megvalósítandó hívások: `t.llx=1; t.lly=2; t.urx=3; t.ury=4`. Ha a hívó olyan értéket akar beállítani, hogy a téglalap mérete negatív lenne, dobjon *NegativMeretHiba* kivételt.
    - Írjon függvényt *sulypontok* néven, amely soronként beolvassa a bemenetet, minden sorban 4 számot várjon (elemekre bontás: `s.split` hívással), ezek a számok a bal alsó sarok x majd y koordinátája, a szélesség és a magasság. A kimenetre a téglalap súlypontját írja (soremeléssel lezárva), a *sulypont* metódus hívásával. A *NegativMeretHiba* kivételt kapja el, ekkor a kimenetre egy N bet? és egy soremelés kerüljön.

- Készítsen *Szamegyenes* néven osztályt Rubyban. A számegyenes olyan, mint egy tömb, de nem csak pozitív, hanem negatív irányban is tetsz?legesen b?víthet?. Az adatok tárolásához használjon 2 tömböt: *@p* és *@n*. Például a számegyenes 5-ödik elemét *@p[5]*-ben, a -5-ödik elemét pedig *@n[4]*-ben tárolja. Kezdetben mindkét tömb legyen üres.
  - Írja meg a `def [](i); ... end` metódus törzsét, mely visszaadja a számegyenes i-edik elemét.
  - Írja meg a `def []=(i,uj); ... end` metódus törzsét, mely a számegyenes i-edik elemét megváltoztatja *uj*-ra, és visszaadja *uj*-t.
  - Írja meg a `def size(); ... end` metódus törzsét, amely visszaadja a számegyenes elemeinek összszámát, vagyis a *@p* és *@n* tömbök összméretét.
  - Írja meg a `def tukroz!(); ... end` metódus törzsét, amely a számegyest tükrözi az origóra, vagyis minden i-re az i-edik elem helyet cserél a *-i*-edik elemmel.
- Készítsen *KorlatosTomb* néven osztályt Rubyban. A KorlatosTomb egy olyan tömb, amelynek kezdeti mérete kés?bb nem változtatható meg. Megvalósításához a *@t* Ruby tömböt (*Array*) használja.
  - Írja meg a konstruktort, amely paraméterben várja azt az *m* méretet, melyre korlátozva van a tömb.
  - Írja meg a paraméter nélküli *size* metódust, amely visszaadja a konstruktornak átadott *m* méretet.
  - Írja meg a `def [](i); ... end` metódus törzsét, mely visszaadja a korlátos tömb i-edik elemét. Ha az *i* kívül esik az intervallumon, kivételt dob (`fail`).
  - Írja meg a `def []=(i,uj); ... end` metódus törzsét, mely a korlátos tömb i-edik elemét megváltoztatja *uj*-ra, és visszaadja *uj*-t. Ha az *i* kívül esik az intervallumon, kivételt dob (`fail`).

Mintafeladatok megoldásai

- Intervallum

```
class Intervallum
  attr_reader :a, :b
  def initialize(a, b)
    fail unless a <= b
    @a=a; @b=b
  end
  def a=(a)
    fail unless a <= @b
    @a=a
  end
  def b=(b)
    fail unless @a <= b
    @b=b
  end
end
```

- Háromszög és szemétgy?jtés
  - A 2. ZH 1. feladatának kicsit módosított változata:
    - nincs attr_accessor, mert felesleges,
    - `#itt` és `#ott` és `#amott` ki is írjuk, mi van az objektumokban.

```
class Haromszog
  def initialize(a, b, c)
    fail unless a+b>c and a+c>b and b+c>a
    @a=a; @b=b; @c=c
  end
end
class SzabalyosHaromszog < Haromszog
```

```
  def initialize(a)
    super(a, a, a)
  end
end

begin
  h = Haromszog.new(3,4,5); # haromszog1
  sz = SzabalyosHaromszog.new(6); # szabalyosharomszog1
  h2 = SzabalyosHaromszog.new(5); # szabalyosharomszog2
  # itt
  p '# itt h: ' + h.to_s + ' sz: ' + sz.to_s
  h = Haromszog.new(1,2,3); # haromszog2
rescue
  h2 = nil
  # ott
  p '# ott h: ' + h.to_s + ', sz: ' + sz.to_s
  sz = h
ensure
  h = sz
  # amott
  p '# amott h: ' + h.to_s + ', sz: ' + sz.to_s
end
```

- Készítünk 3 objektumot, `#itt` nem szabadítható fel semmi, `h` `haromszog1` és `sz` `szabalyosharomszog1`.
- A `haromszog2` létre sem jön (háromszög-egyenl?tlenség miatt)
  - így `h` marad a korábbi, hiszen az értékadás csak a konstruktorhívás után kerül(ne) sorra,
  - és átkerülünk a `rescue` blokkba.
- A `rescue` blokkban el?ször `h2` változik, emiatt `#ott` `h` marad `haromszog1` és `sz` marad `szabalyosharomszog1`, ellenben `szabalyosharomszog2` felszabadítható,
  - kés?bb `sz`-be is `haromszog1` kerül.
- Az `ensure` blokk mindig lefut,
  - `h`-ba `sz`, azaz `haromszog1` kerül (nem mintha eddig nem az lett volna),
  - végül `#amott` felszabadítható `szabalyosharomszog1` is.

GyakIV

A gyakIV 2007. május 25-én, pénteken 12:00-tól 16:00-ig kerül megrendezésre a K.A.62-ben. A gyakIV a teljes féléves anyagból az, ami a wikiben meg van említve.