Tartalomjegyzék

- 1 A második ZH
  - 1.1 Mintafeladatok
  - 1.2 Megoldások
  - 1.3 Mit kell tudni
    - 1.3.1 Mely beépített metódusokat kell ismerni az egyes osztályokból

A második ZH

Id?pont, hely: április 30 16-17, Ka.26

ZH papíron, használható a 2-oldalas Ruby Quick Ref

A ZH témája objektum-orientált programozás Ruby nyelven.

Mintafeladatok

1. Mit ad vissza a Ruby az alábbi kifejezésekre

```
1<2 && 2<4
2 & 7
"http://wiki.math.bme.hu#{n=6; n+=1}"http://wiki.math.bme.hu
'pi = %.5f' % Math::PI
?a
'a'[0]
'abcd'[0]
123.to_s
17.5.to_i
"http://wiki.math.bme.hu17.5"http://wiki.math.bme.hu.to_i
12.to_f
"http://wiki.math.bme.hualma korte szilva barack"http://wiki.math.bme.hu.gsub(/([^\s]+)/,'(\1)')
'10-99'.succ
(1..4).each {|i| print i}
(1..4).map {|i| i**2}
(1..4).inject {|s,i| s+i}
```

2. Írjuk a `nev` változóban megadott nevet egy több sorból álló `email` nev? sztring megadott helyére!

3. Hozzunk létre egy SajatString nev? osztályt, mely megörökli a String összes tulajdonságát, de a gsub metódus úgy m?ködik benne, mint a String osztályban, csak az eredmény végéhez hozzáírja, hogy "http://wiki.math.bme.hu(sajat valtozat)"http://wiki.math.bme.hu.

4. Írjunk programrészletet, mely egy tömbökb?l álló tömb elemeit kiírja az els? definiálatlan (nil) elemig. (catch/throw)

5. Írjunk `sorszamoz` nev? metódust, mely az arumentumában megadott fájlnev? fájlt beolvassa, és sorait kiírja megsorszámozva, a sorszámozást 1-gyel kezdve!

6. Írjunk Teglalap nev? osztályt, melyben a példányváltozók @ba, @jf, @szelesseg, @magassag a bal alsó és jobb fels? sarok koordinátái [x,y] alakban megadva, valamint a szélesség és a magasság. Egy új példányt kétféleképp is létre lehessen hozni, a két sarok, vagy a bal alsó sarok és a szélesség meg a magasság megadásával. Pl. Teglalap.new([1,2],[4,4]) vagy Teglalap([1,2], 3, 2). Ha az argumentumok száma nem 2 vagy 3, kapjunk hibaüzenetet!

7. Írj rekurzív metódust az Integer osztályba, mely kiszámítja az *n*-edik Fibonacci-számot!

8. Írj rekurzív metódust az Integer osztályba, mely kiszámítja az *n*-edik Fibonacci-számot *memoization*-t használva, vagyis a már kiszámolt Fibonacci-számokat egy tömbben tárolva!

9. Írj `sortores` nev? függvényt, mely egy sztringet megadott sorszélesség?re tördel. Ez azt jelenti, hogy egy sor sem lehet a megadott szélességnel szélesebb, hacsak nincs a sorban egyetlen szóköz vagy TAB karakter sem. Minden sor a lehet? legszélesebb legyen a megadott határon belül. A tördelés csak szóköz vagy TAB karakter helyén lehet (a korábbi újsor karaktereket meg kell tartani). Ha a sortörés helyén több szóköz/TAB karakter van, azokat csak egyetlen újsor karakterre cseréljük! A függvénynek két argumantuma legyen, els? egy sztring, a második opcionális, mely alapértelmezésben 70, és amely a sorszélességet adja meg. (Használjuk a gsub metódust, és reguláris kifejezéseket!)

10. Szimuláljunk 6 kockadobást, és adjuk össze a dobott számokat!

11. Írjunk egy Fa nev? osztályt, egy fa konstruálására, és abba 2 metódust: <<, each. Az initialize metódust megadjuk. A << metódussal lehessen a Fa osztály egy példányában egy csúcshoz fát illeszteni, az each metódussal pedig be lehessen járni a fa minden csúcsát.

```
class Fa
  attr_reader :ertek

  def initialize( ertek )
    @ertek = ertek
    @gyerek = []
  end

  def <<( ertek )
    ...
  end

  def each
    ...
  end
end
```

Példaként megadunk egy párbeszédet a Fa osztály betöltése után:

```
>> load "http://wiki.math.bme.hufa.rb"http://wiki.math.bme.hu
>> f = Fa.new("http://wiki.math.bme.huszulo"http://wiki.math.bme.hu)
=> #<Fa:0xb7c5d6d8 @gyerek=[], @ertek="http://wiki.math.bme.huszulo"http://wiki.math.bme.hu>
>> gy1 = f << "http://wiki.math.bme.huelso gyerek"http://wiki.math.bme.hu
=> #<Fa:0xb7c596c8 @gyerek=[], @ertek="http://wiki.math.bme.huelso gyerek"http://wiki.math.bme.hu>
>> u11 = gy1 << "http://wiki.math.bme.hu1/1. unoka"http://wiki.math.bme.hu
=> #<Fa:0xb7c55bb8 @gyerek=[], @ertek="http://wiki.math.bme.hu1/1. unoka"http://wiki.math.bme.hu>
>> uu11 = u11 << "http://wiki.math.bme.huukunoka"http://wiki.math.bme.hu
=> #<Fa:0xb7c523dc @gyerek=[], @ertek="http://wiki.math.bme.huukunoka"http://wiki.math.bme.hu>
>> gy2 = f << "http://wiki.math.bme.humasodik gyerek"http://wiki.math.bme.hu
=> #<Fa:0xb7c4a060 @gyerek=[], @ertek="http://wiki.math.bme.humasodik gyerek"http://wiki.math.bme.
>> gy2 << "http://wiki.math.bme.hu2/1. unoka"http://wiki.math.bme.hu
=> #<Fa:0xb7c457cc @gyerek=[], @ertek="http://wiki.math.bme.hu2/1. unoka"http://wiki.math.bme.hu>
>> gy2 << "http://wiki.math.bme.hu2/2. unoka"http://wiki.math.bme.hu
=> #<Fa:0xb7c42130 @gyerek=[], @ertek="http://wiki.math.bme.hu2/2. unoka"http://wiki.math.bme.hu>
>> f.each {|i| puts i}
szulo
elso gyerek
1/1. unoka
ukunoka
masodik gyerek
```

```
2/1. unoka
2/2. unoka
=> [#<Fa:0xb7c596c8 @gyerek=[#<Fa:0xb7c55bb8 @gyerek=[#<Fa:0xb7c523dc @gyerek=[], @ertek="http://w
@ertek="http://wiki.math.bme.hu1/1. unoka"http://wiki.math.bme.hu>], @ertek="http://wiki.math.bme.
@ertek="http://wiki.math.bme.hu2/1. unoka"http://wiki.math.bme.hu>, #<Fa:0xb7c42130 @gyerek=[], @e
```

12. 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 cserélje fel a változók értékét!

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

Útmutatás: írja meg az `a=` és a `a=` metódusokat! `attr_accessor` helyett mit kell írni?

13. 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. A beolvasás elvégezhet? a `STDIN.each_line` iterátorral.

14. Írjon Ruby-programot *palindromlehet.rb* néven, amely beolvassa a bemenet sorait, és kiírja azokat a sorokat, melyekben a bet?k átrendezésével 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: vágja szét bet?kre/rendezze/ragassza össze a sztring karaktereit az `s=s.split().sort.join` hívással.

Megoldások

1. Másoljuk be az irb-be!

2. Egy lehetséges megoldás:

```
nev = "http://wiki.math.bme.huKati"http://wiki.math.bme.hu
email = <<END
Kedves #{nev}!
Szeretettel gondolunk...
...
Pista
END
```

3.

```
class SajatString < String
  def gsub(*args)
    return "http://wiki.math.bme.hu#{super} (sajat valtozat)"http://wiki.math.bme.hu  # el?hívjuk
  end
end
```

4.

```
a=[[1,nil,3],[2,3,4],[0,2,5]]
catch :kiugrunk do
  for i in 0...a.size do
    for j in 0...a[i].size do
      throw :kiugrunk unless a[i][j]
```

```
      print a[i][j]
    end
    print "http://wiki.math.bme.hu\n"http://wiki.math.bme.hu
  end
end
```

5.

```
def sorszamoz(filenev)
  File.open(filenev) do |f|
    f.each_with_index do |sor,i|
      print "http://wiki.math.bme.hu#{i+1}: #{sor}"http://wiki.math.bme.hu
    end
  end
end
```

6.

```
class Teglalap
  attr_accessor :ja, :bf, :szelesseg, :magassag
  def initialize(*args)
    case args.size
    when 2
      @ba, @jf = args
      @szelesseg = @jf[0] - @ba[0]
      @magassag = @jf[1] - @ba[1]
    when 3
      @ba, @szelesseg, @magassag = args
      @jf = [@ba[0]+@szelesseg, @ba[1] + @magassag]
    else
      raise ArgumentError, "http://wiki.math.bme.hu2 vagy 3 argumentumot kell megadni!"http://wiki
    end
  end
end
```

7.

```
class Integer
  def fib()
    return self if self<2
    return (self-1).fib + (self-2).fib
  end
end
```

8.

```
class Integer
  @@fib_tab = [0,1]
  def fib()
    @@fib_tab[self] ||= (self-1).fib + (self-2).fib
  end
end
```

9.

```
def sortores( s, sorszelesseg = 70 )
  s.gsub(/(.{1,#sorszelesseg}})(\s+|\z)/,"http://wiki.math.bme.hu\\1\n"http://wiki.math.bme.hu)
end
```

10.

```
(Array.new(6) {rand(6) + 1}).inject {|s,i| s + i}
```

vagy

```
a = Array.new(6) {rand(6) + 1}
a.inject {|s,i| s + i}
```

11.

12. A feladatban megadott kód eredményeként létrejön egy "http://wiki.math.bme.hua"http://wiki.math.bme.hu, "http://wiki.math.bme.hub"http://wiki.math.bme.hu, "http://wiki.math.bme.hua="http://wiki.math.bme.hu és egy "http://wiki.math.bme.hub="http://wiki.math.bme.hu metódus is. Teszteljük le irb-ben:

```
i=Intervallum.new(2,4)
i.methods
i.a
i.a=1
i.a
```

A feladat megoldása:

```
class Intervallum
  attr_reader :a, :b    # ide most elég attr_reader
  def initialize(a, b)
    if a < b
      @a=a; @b=b
    else
      @a=b; @b=a
    end
  end
  def a=(a)
    if a <= @b
      @a=a
    else
      @a=@b
      @b=a
    end
  end
  def b=(b)
    if @a <= b
      @b=b
    else
      @b=@a
      @a=b
    end
  end
end
```

13.

```
STDIN.each_line do |s|
  s.chomp!
  if s == s.reverse
    puts s
  end
end
```

14. 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
}
```

Mit kell tudni

m?veletek számokkal, karakterláncokkal Osztály, osztálypéldány, objektum, konstruktor, objektum azonosító, példány változó, példány metódus fogalma if, else, elsif, if kifejezésekben, ?:, unless, case/when/else, === és szerepe a case-ben, while, until, for/in, iterátorok: times, upto, downto, each, map/collect, inject, each_with_index break/next, blokkok redo/retry, catch/throw

Mely beépített metódusokat kell ismerni az egyes osztályokból

- Object
  - class (pl. 5.class)
  - methods (5.methods)
- Float
  - abs + - * < <= == != >= > <=> ** (a <=> az összahasonlítás után -1/0/1 értéket ad vissza)
- Fixnum
  - abs + - * < <= == != >= > <=> << >> **
- Bignum
  - (semmit)
- Integer
  - (semmit)
- String
  - + =~ !~ chomp chomp! reverse upcase upcase! downcase downcase! reverse reverse!
- Array
  - [] []= + - each size << max min sort sort! include?
- Class
  - superclass
- TrueClass
  - && || ! == !=
- FalseClass

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

p, print, puts.

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