Matematika Diskretua
Laboratorio saioa: Zenbaki Teoria eta R

R softwareak "numbers"paketean ditu inplementatuta Zenbaki Teoriako funtzio asko. Paketea instalatu egin behar da `install.packages("numbers")` aginduaren bidez. Ondoren, kargatzeko `library("numbers")` idatz ezazu. `library(help="numbers")` aginduaren bidez "numbers"paketeko funtzio guztien zerrenda aterako zaizu. Informazio zabalagoa aurkituko duzu manualean: http://cran.r-project.org/web/packages/numbers/numbers.pdf

1. **Ariketa**. "numbers"paketeko hainbat funtzio erabiltzea. Funtzioak nola inplementatuak izan diren ikusteko, nahikoa da funtzioaren izena idaztea, parametrorik pasa gabe. Adibidez:

   ```
   isNatural
   ```

   Adi! isNatural funtzioak $2^{53} - 1$ zenbakia baino handiagorik ez du onartzen...

   - Zenbaki arruntak eta lehenak. Arrunta den egiaztatzeko: isNatural(n), karakterea: is.character(n), lehena: isPrime(n), zenbaki lehenen zerrenda bat sortzeko: Primes(n1,n2), emandako zenbakia baino handiagoa den hurrengo zenbaki lehena kalkulatzeko: nextPrime(n), aurrekoa: previousPrime(n)

     ```
     isNatural(8)
     2^53
     2^52
     isNatural(2^52)
     isNatural(2^53)
     # 15 zeroko zenbakia
     isNatural(1000000000000000)
     # 16 zeroko dituena
     isNatural(10000000000000000)
     is.character(8)
     is.character("8")
     abs(-8)
     isPrime(7)
     isPrime(8)
     Primes(1,20)
     Primes(100,200)
     nextPrime(2000)
     nextPrime(99999)
     previousPrime(100019)
     ```

   - Zenbaki lehen erlatiboak diren egiaztatzeko: coprime(n,m)

     ```
     coprime(1448,1001)
     coprime(1448,1002)
     ```

   - Zenbakien deskonposaketa zenbaki lehenetan: primeFactors(n)

     ```
     primeFactors(1448)
     primeFactors(1001)
     ```

   - Zatiketa Euklidearra. Emandako n eta m zenbakien arteko zatiketa euklidestarra egin eta zatidura lortzeko: div(n,m) edo n%/% m, hondarra lortzeko: mod(n,m) edo n%%m

```
div(7,2)
7%/%2
mod(7,2)
7%%2
```

- Zatitzaile komunetako handienaren kalkulua (greatest common divisor): GCD(n,m)

```
GCD(1448,1001)
GCD(1448,1002)
```

- Zatitzaile komunetako handienaren 5.propietatea (Bezout-en identitatea). Emandako zenbakien zkh kalkulatu eta zkh(a,b)=xa+yb konbinazio linealeko x eta y koefizienteak kalkulatzen ditu Euklidesen algoritmoa erabiliz: extGCD(a,b)

```
extGCD(1448,1001)
[1]    1 -421  609
> 1==-421*1448+609*1001
[1] TRUE
extGCD(1448,1002)
[1]    2 -164  237
> 2==-164*1448+237*1002
[1] TRUE
```

2. Ariketa. Bi zenbaki oso emanik, a eta b, zatitzaile komunetako handiena kalkulatuko duen funtzioa da ondoren daukazuna. ADA programazio-lengoaian programatuta dago. Funtzioaren inplementazioa Euklidesen algoritmoan oinarrituta dago.

```
function zkh(a, b : Positive) return Positive is
        --Aurre:
        --Post: zkh(a,b) bueltatzen du

      r, c, d : Integer;
    begin
        c := a;
        d := b;
        while d /= 0 loop
            r := c mod d;
            c := d;
            d := r;
        end loop;
        return c;
    end zkh;
```

2.1 Egoki ezazu funtzioa R programazio lengoaiarako. Ondorein dei egin funtzioari. Irteera zuzena dela egiaztatzeko `GCD(a,b)`, `coprime(a,b)` eta `extGCD(a,b)` funtzioak erabil ditzakezu.

```
## Deiak funtzioari. a=231, b=1820  -------------------------
zkh(231,1820)
coprime(231,1820)
GCD(231,1820)
extGCD(231,1820)
## Deiak funtzioari. a=1369, b=2597  ---------------------------
zkh(1369,2597)
coprime(1369,2597)
```

```
GCD(1369,2597)
extGCD(1369,2597)
## Deiak funtzioari. a=2689, b=4001  ----------------------------
zkh(2689,4001)
coprime(2689,4001)
GCD(2689,4001)
extGCD(2689,4001)
```

2.2 Inplementatu duzun funtzioa hobe ezazu. Parametro gisa pasatako a eta b zenbaki arruntak ez badira, errore mezua eman behar da. Horretarako `is.character(a)`, `isNatural(a)` eta `abs(a)` funtzioak erabil ditzakezu, eta mezuak pantailan bistaratzeko `print("")` funtzioa.

```
zkh(3.2,4)
# "a eta b parametroek zenbaki osoak izan behar dute"

zkh("a","b")
# "a eta b ezin dira karaktereak izan"
```

3. Ariketa. Zenbaki batekin lehen erlatiboa den beste zenbaki bat kalkulatuko duen funtzioa idatz ezazu. Funtzioa RSA gakoen kalkuluan beharko dugu. Funtzioaren bi bertsio inplementatuko ditugu:

3.1 Emandako m zenbakiarekin lehen erlatiboa den zenbaki oso positiborik txikiena itzultzea.

```
lehen_erlatibo_txiki <- function(m)
{
}
## Deiak funtzioari.
lehen_erlatibo_txiki(13797)
lehen_erlatibo_txiki(16974)
lehen_erlatibo_txiki(56970)
lehen_erlatibo_txiki(1000000000000000)
```

3.2 Emandako m zenbakiarekin lehen erlatiboa den eta t baino handiagoa den zenbaki oso positiboa itzultzea.

```
lehen_erlatibo <- function(m,t)
{
}
## Deiak funtzioari.
lehen_erlatibo(13797, 50)
lehen_erlatibo(16974, 54687)
lehen_erlatibo(56970,26378)
lehen_erlatibo(9674,26378)
lehen_erlatibo(1000000000000000,26378)
lehen_erlatibo(1000000000000000,2637800000)
```

4. **Ariketa**. ASCII kodeketa (**A**merican **S**tandard **C**ode for **I**nformation **I**nterchange) Informazioaren Elkartrukerako Kode Estatubatuar Estandarra da, alfabeto latinean oinarritutako karaktereen kodeketa bat. Gaur egungo sistema informatiko gehienek ASCII kodeketa darabilte testuak adierazi eta testua maneiatzen duten gailuen kontrolerako. http://eu.wikipedia.org/wiki/ASCII helbidean aurki dezakezu informazio gehiago. Bertan, *Karaktere inprimagarriak* ataleko taulan ikusiko duzu "kaixo"hitzeko bost hizkiei dagozkien ASCII kodeak 107 97 105 120 111 direla.

   ASCII kodeak lortzeko `strtoi(charToRaw("k"),16L)` agindua erabil dezakezu R softwarean. Karaktereak banaka eman beharrik ez dago. Egin proba!

```
strtoi(charToRaw("k"),16L)
strtoi(charToRaw("a"),16L)
strtoi(charToRaw("i"),16L)
strtoi(charToRaw("x"),16L)
strtoi(charToRaw("o"),16L)
strtoi(charToRaw("kaixo"),16L)
```

   ASCII kodeetatik hizkietara `rawToChar(as.raw(`n`))` agindua erabil dezakezu. Kodeak banaka eman nahi ez badituzu, bektore moduan antolatu beharko dituzu.

```
rawToChar(as.raw(107))
rawToChar(as.raw(97))
rawToChar(as.raw(105))
rawToChar(as.raw(120))
rawToChar(as.raw(111))
hitza<-c(107,97,105,120,111)
rawToChar(as.raw(hitza))
```

   Idatz ezazu funtzio bat testu bat emanik, testu sekretu bihurtuko duena, zuk nahi duzun eraldaketa aplikatuz.

```
testu_sekretu <- function(testua)
{

}
```

   Testua ASCII kode bihur dezakezu adibidez, ondoren kode guztiak +5 egin, kode horiei dagokien testua testu sekretu gisa itzultzeko. Hau lortu beharko zenuke:

```
testu_sekretu("kaixo")
[1] "pfn}t"
```

   Izan ere, "k"$\rightarrow$ 107 $\rightarrow$ 112 $\rightarrow$ "p".

   Horrela hasi zen kriptografia aintzina. Begira artikulu hauek:

   http://zientzia.eus/artikuluak/kode-sekretuak-i-ii-iii/