Tarea 3: Estructura de datos

Imanol Miguez

11/2/2021

RESPUESTAS DE LA TAREA

1. Cread un vector llamado "Harry" formado por la sucesión de números consecutivos entre el -10 y 27. Pedidle a R que os devuelva el elemento de índice 7. Escribid el resultado.

```
Harry <- seq(-10,27)
Harry
```

```
##  [1] -10  -9  -8  -7  -6  -5  -4  -3  -2  -1   0   1   2   3   4   5   6   7   8
## [20]   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27
```

```
Harry[7]
```

```
## [1] -4
```

Profesor

```
Harry=-10:27

Harry[7]
```

```
## [1] -4
```

2. Dad el máximo de la sucesión (100*2^n* -73^n) con n=0,. . . ,200 n=0:200

```
max(sapply(seq(0,200), FUN = function(n){100*5^n-7*3^n}))
```

```
## [1] 6.223015e+141
```

Profesor

```r
n=0:200

max(100*2^n-7*3^n)
```

```
## [1] 1499
```

3. Cread la sucesión de números consecutivos entre 0 y 40. A continuación, cread el vector (3*5^n - 1) con n=0,...,40. Ponedle como nombre x. Ahora, dad el subvector de los elementos que son estrictamente mayores que 3.5

```r
n = seq(0,40)
x <- sapply(n, FUN = function(x){3*(5^x)-1})
x[which(x>3.5)]
```

```
##  [1] 1.400000e+01 7.400000e+01 3.740000e+02 1.874000e+03 9.374000e+03
##  [6] 4.687400e+04 2.343740e+05 1.171874e+06 5.859374e+06 2.929687e+07
## [11] 1.464844e+08 7.324219e+08 3.662109e+09 1.831055e+10 9.155273e+10
## [16] 4.577637e+11 2.288818e+12 1.144409e+13 5.722046e+13 2.861023e+14
## [21] 1.430511e+15 7.152557e+15 3.576279e+16 1.788139e+17 8.940697e+17
## [26] 4.470348e+18 2.235174e+19 1.117587e+20 5.587935e+20 2.793968e+21
## [31] 1.396984e+22 6.984919e+22 3.492460e+23 1.746230e+24 8.731149e+24
## [36] 4.365575e+25 2.182787e+26 1.091394e+27 5.456968e+27 2.728484e+28
```

Profesor

```r
n=0:40

x=3*5^n-1

which(x>3.5)
```

```
##  [1]  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
## [26] 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41
```

4. Cread una función que os devuelva la parte real, la imaginaria, el módulo, el argumento y el conjugado de un número, mostrando solo 2 cifras significativas

```r
complejos = function(x) {
 r1 <- round(c(Re(x),Im(x),Mod(x),Arg(x)),2)
 r2 <- round(Conj(x),2)

 r = list(Real = r1[1],Imaginario = r1[2], Modulo = r1[3], Argumento = r1[4], Conjugado = r2)

  return(r)
}

complejos(2+3i)
```

```
## $Real
## [1] 2
##
## $Imaginario
## [1] 3
##
## $Modulo
## [1] 3.61
##
## $Argumento
## [1] 0.98
##
## $Conjugado
## [1] 2-3i
```

Profesor

```
info = function(x){print(c(Re(x),Im(x),Mod(x),Arg(x),Conj(x)),2)}
info(2+3i)
```

```
## [1] 2.00+0i 3.00+0i 3.61+0i 0.98+0i 2.00-3i
```

5. Cread una función que resuelva ecuaciones de segundo grado (de la forma Ax^2+Bx+C=0). No importa, por ahora, que tengáis en cuenta las que no tienen solución

```
ecuacion = function(A,B,C){
  r <- c((-B+sqrt(B^2-4*A*C))/(2*A),(-B-sqrt(B^2-4*A*C))/(2*A))
  if (is.na(r)){
    return("No hay solución")
  }
  return(r)
}
ecuacion(1,1,-2) # x^2 + x - 2 = 0
```

```
## Warning in if (is.na(r)) {: la condición tiene longitud > 1 y sólo el primer
## elemento será usado
```

```
## [1]  1 -2
```

```
ecuacion(2,-5,3) # 2x^2 - 5x + 3 = 0
```

```
## Warning in if (is.na(r)) {: la condición tiene longitud > 1 y sólo el primer
## elemento será usado
```

```
## [1] 1.5 1.0
```

```r
ecuacion(1,-1,1) # x^2 - x + 1 = 0
```

```
## Warning in sqrt(B^2 - 4 * A * C): Se han producido NaNs

## Warning in sqrt(B^2 - 4 * A * C): Se han producido NaNs

## Warning in if (is.na(r)) {: la condición tiene longitud > 1 y sólo el primer
## elemento será usado

## [1] "No hay solución"
```

Profesor

```r
solu2 = function(A,B,C){c((-B+sqrt(B^2-4*A*C)/(2*A)),(-B-sqrt(B^2-4*A*C)/(2*A)))}
solu2(1,1,2)
```

```
## Warning in sqrt(B^2 - 4 * A * C): Se han producido NaNs

## Warning in sqrt(B^2 - 4 * A * C): Se han producido NaNs

## [1] NaN NaN
```

6.Tomando el vector vec = c(0,9,98,2,6,7,5,19,88,20,16,0), dad 3 opciones diferentes para calcular el subvector c(9,19,20,16)

Tomando el vector vec definido en el apartado anterior, buscad

- qué entradas son pares

- qué entradas no son pares y mayores que 20

- dónde toma vec su valor máximo

- dónde toma vec sus valores mínimos

```r
vec = c(0,9,98,2,6,7,5,19,88,20,16,0)
subvec = c(9,19,20,16)

vec[c(2,8,10,11)]                                    # Opción 1
```

```
## [1]  9 19 20 16
```

```r
vec[-c(1,3,4,5,6,7,9,12)]                            # Opción 2
```

```
## [1]  9 19 20 16
```

```r
vec[vec == 9 | vec == 19 | vec == 20 | vec == 16] # Opción 3
```

```
## [1]  9 19 20 16
```

```r
vec[vec >= 9 & vec <= 20]                         # Opción 4
```

```
## [1]  9 19 20 16
```

```r
vec[which(vec%%2 == 0)]            # Entradas pares, contando 0 como par
```

```
## [1]  0 98  2  6 88 20 16  0
```

```r
vec[which(vec%%2 != 0 & vec > 20)] # Entradas impares y mayores de 20
```

```
## numeric(0)
```

```r
which.max(vec)             # Lugar donde toma el valor máximo
```

```
## [1] 3
```

```r
which(vec == min(vec))             # Lugar donde toma los valores mínimos
```

```
## [1]  1 12
```

EJERCICIOS DEL PDF

EJERCICIO 1

```r
A = rbind(c(1,3),c(2,4))

A2 = A %*% (A + A) %*% A
A2[2,2]
```

```
## [1] 236
```

EJERCICIO 2

```r
B = rbind(c(1,4,-6),c(0,0,3),c(0,-2,5))

eigen(B)$values
```

```
## [1] 3 2 1
```

EJERCICIO 3

```r
C = rbind(c(-48,35,-12),c(-134,95,-32),c(-194,133,-44))

round(eigen(C)$vectors,3)
```

```
##       [,1]  [,2]   [,3]
## [1,] 0.371 0.169  0.098
## [2,] 0.743 0.507 -0.195
## [3,] 0.557 0.845 -0.976
```

EJERCICIO 4

```r
D = rbind(c(-2,-8,-2,3),c(-3,-6,-1,2),c(-9,-22,-3,7),c(-18,-44,-8,15))

qr(D)$rank
```

```
## [1] 3
```