Fundamentos de R

Víctor Arath Ramírez López

19/10/2020

1. (43 puntos) Operaciones básicas

(a) Crea un vector llamado contarpor5 que es una secuencia de números reales desde el 5 hasta el 100 en incrementos de 5. (3 puntos)

```
cortarpor5<-seq(5,100,5)
cortarpor5
```

```
##  [1]   5  10  15  20  25  30  35  40  45  50  55  60  65  70  75  80  85  90  95
## [20] 100
```

(b) Crea un vector llamado Tratamiento cuyas entradas se forman como sigue: "Tratamiento 1" aparece 20 veces, "Tratamiento 2" aparece 18 veces, y "Tratamiento 3" aparece 22 veces. (3 puntos)

```
Tratamiento<-c(rep("Tratamiento 1", 20), rep("Tratamiento 2", 18), rep("Tratamiento 3", 22))
Tratamiento
```

```
##  [1] "Tratamiento 1" "Tratamiento 1" "Tratamiento 1" "Tratamiento 1"
##  [5] "Tratamiento 1" "Tratamiento 1" "Tratamiento 1" "Tratamiento 1"
##  [9] "Tratamiento 1" "Tratamiento 1" "Tratamiento 1" "Tratamiento 1"
## [13] "Tratamiento 1" "Tratamiento 1" "Tratamiento 1" "Tratamiento 1"
## [17] "Tratamiento 1" "Tratamiento 1" "Tratamiento 1" "Tratamiento 1"
## [21] "Tratamiento 2" "Tratamiento 2" "Tratamiento 2" "Tratamiento 2"
## [25] "Tratamiento 2" "Tratamiento 2" "Tratamiento 2" "Tratamiento 2"
## [29] "Tratamiento 2" "Tratamiento 2" "Tratamiento 2" "Tratamiento 2"
## [33] "Tratamiento 2" "Tratamiento 2" "Tratamiento 2" "Tratamiento 2"
## [37] "Tratamiento 2" "Tratamiento 2" "Tratamiento 3" "Tratamiento 3"
## [41] "Tratamiento 3" "Tratamiento 3" "Tratamiento 3" "Tratamiento 3"
## [45] "Tratamiento 3" "Tratamiento 3" "Tratamiento 3" "Tratamiento 3"
## [49] "Tratamiento 3" "Tratamiento 3" "Tratamiento 3" "Tratamiento 3"
## [53] "Tratamiento 3" "Tratamiento 3" "Tratamiento 3" "Tratamiento 3"
## [57] "Tratamiento 3" "Tratamiento 3" "Tratamiento 3" "Tratamiento 3"
```

(c) Asigna los nombres "x" y "y" a los valores 5 y 7 respectivamente. Calcula x^y y a este resultado asígnale el nombre "z". ¿Cuál es el valor de z? (3 puntos)

```
x<-5; y<-7
z<- x^y
z #resultado
```

```
## [1] 78125
```

(d) Sean los vectores u = (1, 2, 5, 4) y v = (2, 2, 1, 1). Realiza lo siguiente:

i. Escribe y ejecuta el código para encontrar la componente de u que es igual a 5. No se pide la posición de este valor dentro del vector. (3 puntos)

```
u<-c(1,2,5,4); v<-c(2,2,1,1)

for (i in u) {
  if(i==5){
    print(i)
  }
}
```

```
## [1] 5
```

ii. Escribe y ejecuta el código para que proporcione los valores de v 2. (3 puntos)

```
v[v>=2]
```

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

iii. Define a la matriz X cuyos renglones son los vectores u y v, y a la matriz Y cuyas columnas son los vectores u y v multiplicados por una constante k = 2. Calcula el producto W = XY. (8 puntos)

```
k<-2
X <- matrix(c(u,v), nrow = 2, ncol = 4, byrow = T)
Y<- k*(matrix(c(u,v), nrow = 4, ncol = 2, byrow = F))

W<-X%*%Y
W
```

```
##      [,1] [,2]
## [1,]   92   30
## [2,]   30   20
```

iv. Escribe y ejecuta el código para calcular la inversa de la matriz W. Compara tu resultado con el obtenido mediante la aplicación directa de la función: solve(W). (8 puntos)

```
W1<-X%*%Y
W1<-matrix(c(W1[2,2],W1[1,2]*-1,W1[2,1]*-1,W1[1,1]), 2,2,T )
W1<-(1/det(W1))*W1; W1
```

```
##             [,1]        [,2]
## [1,]  0.02127660 -0.03191489
## [2,] -0.03191489  0.09787234
```

```
solve(W) ###Comrpobación
```

```
##             [,1]        [,2]
## [1,]  0.02127660 -0.03191489
## [2,] -0.03191489  0.09787234
```

(e) Resuelve el siguiente sistema de ecuaciones lineales con 5 incógnitas:

$$
\begin{aligned}
x1 + 2x2 + 3x3 + 4x4 + 5x5 &= 7 \\
2x1 + x2 + 2x3 + 3x4 + 4x5 &= -1 \\
3x1 + 2x2 + x3 + 2x4 + 3x5 &= -3 \\
4x1 + 3x2 + 2x3 + x4 + 2x5 &= 5 \\
5x1 + 4x2 + 3x3 + 2x4 + x5 &= 17
\end{aligned}
$$

mediante la ecuación matricial apropiada $Ax = y$. (12 puntos)

```
A<-matrix(c(1,2,3,4,5,2,1,2,3,4,3,2,1,2,3,4,3,2,1,2,5,4,3,2,1), 5,5, TRUE)
Y<-matrix(c(7,-1,-3,5,17))
X<-solve(A)%*%Y
X
```

```
##      [,1]
## [1,]   -2
## [2,]    3
## [3,]    5
## [4,]    2
## [5,]   -4
```

2. (31 puntos) Funciones

(a) Programa una función que calcule el factorial de cualquier número entero n, esto es, n!. Aplícala para obtener 8!. (8 puntos)

```
fac <- function(x){
  fac <- x
  while (x > 1){
      x <- x - 1
      fac <- fac*x
  }
  fac
}
fac(8)
```

```
## [1] 40320
```

(b) Programa una función para convertir cualquier temperatura medida en grados Celsius a grados Kelvin y a grados Fahrenheit. Aplícala para obtener las equivalencias de 23°C. (8 puntos)

```
gradosc<-function(x){
  k<-x
  f<-x
  k<-k+273.15
  f<-(f*(9/5))+32
  cat("De",x,"grados Celsius","a",k,"grados Kelvin\n")
  cat("De",x,"grados Celsius","a",f,"grados Fahrenheit")
}
gradosc(23)
```

```
## De 23 grados Celsius a 296.15 grados Kelvin
## De 23 grados Celsius a 73.4 grados Fahrenheit
```

(c) Una caminata aleatoria simple y sim´etrica que empieza en el origen está definida como sigue.Supónngase que X1, X2, X3, . . . , Xn son variables aleatorias i.i.d, esto es, independientes e idénticamente distribuidas con la distribución:

```
caminata<-function(n){
  s0<-0
  j<-c(-1,1)
  s<-sample(j, n, replace = T, prob = c(0.5, 0.5))
  s<-cumsum(s)
  sn<-c(s0,s)
  return(sn)
}
caminata(10)
```

```
##  [1] 0 1 2 3 4 5 6 5 6 7 6
```

(a) Programa la funci´on factorial usando un ciclo for. Aplícala para obtener 8!. (7 puntos)

```
facfor<-function(n){
  fac<-1
  for (i in 1:n) {
    fac <-fac*i
  }
  return(fac)
}
facfor(8)
```

```
## [1] 40320
```

(b) Imagina que tienes una deuda de $20000 que debes pagar al banco. Cada mes, abonas $1250 hasta que la liquidas. Escribe un ciclo while que te permita imprimir en cada iteración tu nuevo total (es decir, la cantidad que resta luego de cada abono). ¿En cuántas mensualidades terminarás de pagar? (9 puntos)

```
x<-20000
i<-0
while (x>0) {
  x<-x-1250
  i<-i+1
  cat("Mes: ",i,"Deuda: ", x,"\n")
}
```

```
## Mes:  1 Deuda:  18750
## Mes:  2 Deuda:  17500
## Mes:  3 Deuda:  16250
## Mes:  4 Deuda:  15000
## Mes:  5 Deuda:  13750
## Mes:  6 Deuda:  12500
## Mes:  7 Deuda:  11250
## Mes:  8 Deuda:  10000
## Mes:  9 Deuda:  8750
## Mes:  10 Deuda:  7500
## Mes:  11 Deuda:  6250
## Mes:  12 Deuda:  5000
## Mes:  13 Deuda:  3750
## Mes:  14 Deuda:  2500
## Mes:  15 Deuda:  1250
## Mes:  16 Deuda:  0
```

4. (10 puntos) Código R. Ejecuta lo siguiente:

```
genero <- factor(c(rep("mujer", 91), rep("hombre", 92)))
table(genero)
```

```
## genero
## hombre  mujer
##     92     91
```

```
genero <- factor(genero, levels=c("hombre", "mujer"))
table(genero)
```

```
## genero
## hombre  mujer
##     92     91
```

```
genero <- factor(genero, levels=c("Hombre", "mujer"))
table(genero)
```

```
## genero
## Hombre  mujer
##      0     91
```

```
rm(genero)
```

a) ¿Encuentras algun error en el código? Si es así, indica cuál(es). (3 puntos)

No encuentro algún error de sintaxis pero si en la salida del tercer table(genero) ya que aroja como resultado 0 en Hombres.

b) Considera la última tabla de género, table(genero). El resultado de esta línea es:

Hombre mujer

0 91

Explica el por qué de estos números. (7 puntos)

La función factor clasifica en niveles a aquellas palabras que son iguales, al ingresar factor(genero, levels=c(“Hombre”, “mujer”)), pedimos que busque en genero los niveles que se llaman Hombre y mujer,sim embargo Hombre no es un nivel de genero, por lo que al buscarlo no encuentra ninguna referencia y es por eso da como resultado 0.