Jorge Fuertes

C:\Users\jorge\Desktop\JORGE\CUNEF\MASTER\PROGRAMACIÓN R\R\datos guardados\BLOQUE 3 TEMA 1...

```
### EJERCICIO 1 - Programación de Scripts y Funciones ###


######## saber si un numero es par.

par<-function(n){
if (n%%2==0) {
sol<-"el numero es par"}
else  {sol<-"el numero es impar"}
return(sol)
}
par(10)
par(28)
par(13)
```

R Console

```
R version 3.4.1 (2017-06-30) -- "Single Candle"
Copyright (C) 2017 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R es un software libre y viene sin GARANTIA ALGUNA.
Usted puede redistribuirlo bajo ciertas circunstancias.
Escriba 'license()' o 'licence()' para detalles de distribucion.

R es un proyecto colaborativo con muchos contribuyentes.
Escriba 'contributors()' para obtener más información y
'citation()' para saber cómo citar R o paquetes de R en publicaciones.

Escriba 'demo()' para demostraciones, 'help()' para el sistema on-line de $
o 'help.start()' para abrir el sistema de ayuda HTML con su navegador.
Escriba 'q()' para salir de R.

[Previously saved workspace restored]

> par<-function(n){
+ if (n%%2==0) {
+ sol<-"el numero es par"}
+ else  {sol<-"el numero es impar"}
+ return(sol)
+ }
> par(10)
[1] "el numero es par"
> par(28)
[1] "el numero es par"
> par(13)
[1] "el numero es impar"
>
```

```
######### ver si es par (1) o impar (0)

par2<-function(n) {
if (n%%2==0) {
sol<-1}
else {sol<-0}
return(sol)
}

par2(27)
par2(28)
```

```
> ######### ver si es par (1) o impar (0)
>
> par2<-function(n) {
+ if (n%%2==0) {
+ sol<-1}
+ else {sol<-0}
+ return(sol)
+ }
>
> par2(27)
[1] 0
> par2(28)
[1] 1
>
```

```
######### contar el numero de pares que hay en el vector

v<-c(7,3,2,6,5,9)
contarpares<-function(v){
contador<-0
for (i in (1:length(v))){
      if (par2(v[i])==1) {contador <- contador+1}}
return (contador)
}
contarpares(v)
```

```
> v<-c(7,3,2,6,5,9)
> contarpares<-function(v){
+ contador<-0
+ for (i in (1:length(v))){
+ if (par2(v[i])==1) {contador <- contador+1}}
+ return (contador)
+ }
> contarpares(v)
[1] 2
>
```

```
######### calcula el factorial de un numero

### ¿Factorial de 8? / ¿Factorial de 231?

factorial<-function(n){
### ese numero debe ser un numero natural
if (n<0) (f<-"el número debe ser positivo o cero")
else {
f<-1
while(n>0)
{
f<-f*n
n<-n-1
}}
return(f)
}

factorial(8)
factorial(0)
factorial(-5)
factorial(231)
```

```
> factorial<-function(n){
+ ### ese numero debe ser un numero natural
+ if (n<0) (f<-"el número debe ser positivo o cero")
+ else {
+ f<-1
+ while(n>0)
+ {
+ f<-f*n
+ n<-n-1
+ }}
+ return(f)
+ }
>
> factorial(8)
[1] 40320
> factorial(0)
[1] 1
> factorial(-5)
[1] "el número debe ser positivo o cero"
> factorial(231)
[1] Inf
>
```

```
######### Construir una agenda de teléfonos
##### almacenada en un data frame con dos campos: nombres y telefonos

nueva.agenda<-function(){
###esta función crea una agenda
###que almacenará en un dataframe con dos campos: nombres, telefonos
nombres<-"Jorge"
telefonos<-"654824569"
continuar<-"S";
while (continuar =="S") {
nombre<-readline(prompt="Dime el nombre: ")
telefono<-readline(prompt="Dime el teléfono: ")
nombres<-c(nombres,nombre)
telefonos<-c(telefonos,telefono)
continuar <- readline(prompt="¿Continuar(S/N)? ")
}
mdf<-data.frame(nombres, telefonos)
mdf$nombres<-as.character(mdf$nombres)
mdf$telefonos<-as.character(mdf$telefonos)
return (mdf)
}
miagenda<-nueva.agenda()
```

```
> nueva.agenda<-function(){
+ ###esta función crea una agenda
+ ###que almacenará en un dataframe con dos campos: nombres, telefonos
+ nombres<-"Jorge"
+ telefonos<-"654824569"
+ continuar<-"S";
+ while (continuar =="S") {
+ nombre<-readline(prompt="Dime el nombre: ")
+ telefono<-readline(prompt="Dime el teléfono: ")
+ nombres<-c(nombres,nombre)
+ telefonos<-c(telefonos,telefono)
+ continuar <- readline(prompt="¿Continuar(S/N)? ")
+ }
+ mdf<-data.frame(nombres, telefonos)
+ mdf$nombres<-as.character(mdf$nombres)
+ mdf$telefonos<-as.character(mdf$telefonos)
+ return (mdf)
+ }
> miagenda<-nueva.agenda()
Dime el nombre: Raul
Dime el teléfono: 124234235
¿Continuar(S/N)? S
Dime el nombre: 214332423
Dime el teléfono: 32413423
¿Continuar(S/N)? S
Dime el nombre: Joaquin
Dime el teléfono: 23423553
¿Continuar(S/N)? N
> miagenda
    nombres telefonos
1     Jorge 654824569
2      Raul 124234235
3 214332423  32413423
4   Joaquin  23423553
```

```
####### En esta funcion se añaden nombres y teléfonos a una agenda que esta
#### almacenada en un data frame con dos campos: nombres y telefonos

#####Construir una agenda de teléfonos###############
add.agenda<-function(agenda.ant){
###esta función añade nombres y teléfonos a una agenda
###que está almacenada en un dataframe con dos campos: nombres, telefonos
nombres<-agenda.ant$nombres
telefonos<-agenda.ant$telefonos
continuar<-"S";
while (continuar =="S") {
nombre<-readline(prompt="Dime el nombre: ")
telefono<-readline(prompt="Dime el teléfono: ")
nombres<-c(nombres,nombre)
telefonos<-c(telefonos,telefono)
continuar <- readline(prompt="¿Continuar(S/N)? ")
}
return (data.frame(nombres, telefonos))
}

miagenda2<-add.agenda(miagenda)
```

```
> #####Construir una agenda de teléfonos###############
> add.agenda<-function(agenda.ant){
+ ###esta función añade nombres y teléfonos a una agenda
+ ###que está almacenada en un dataframe con dos campos: nombres, telefonos
+ nombres<-agenda.ant$nombres
+ telefonos<-agenda.ant$telefonos
+ continuar<-"S";
+ while (continuar =="S") {
+ nombre<-readline(prompt="Dime el nombre: ")
+ telefono<-readline(prompt="Dime el teléfono: ")
+ nombres<-c(nombres,nombre)
+ telefonos<-c(telefonos,telefono)
+ continuar <- readline(prompt="¿Continuar(S/N)? ")
+ }
+ return (data.frame(nombres, telefonos))
+ }
>
> miagenda2<-add.agenda(miagenda)
Dime el nombre: Miriam
Dime el teléfono: 2342346425
¿Continuar(S/N)? S
Dime el nombre: Maria
Dime el teléfono: 56645747
¿Continuar(S/N)? N
> miagenda2
    nombres  telefonos
1     Jorge  654824569
2      Raul  124234235
3 214332423   32413423
4   Joaquin   23423553
5    Miriam 2342346425
6     Maria   56645747
>
```

```
##Construir un vector que contenga los n números pares siguientes a m (par)

construye<-function(m,n){
if (m%%2==1){ v<-"error"}
else {
v<-m+2
m<-m+2
while (n>1)
{
m<-m+2
v<-c(v,m)
n<-n-1
}}
return (v)
}
construye(30,7)
#### Donde m<-30 y n<-7, construyendo un vector con los 7 numeros pares siguientes
#### a 30
```

```
> ##Construir un vector que contenga los n números pares siguientes a m (par)
>
> construye<-function(m,n){
+ if (m%%2==1){ v<-"error"}
+ else {
+ v<-m+2
+ m<-m+2
+ while (n>1)
+ {
+ m<-m+2
+ v<-c(v,m)
+ n<-n-1
+ }}
+ return (v)
+ }
> construye(30,7)
[1] 32 34 36 38 40 42 44
> #### Donde m<-30 y n<-7, construyendo un vector con los 7 numeros pares siguientes
> #### a 30
>
```

```r
##EJERCICIO 3

MAX INT (número más grande que puede representar un ordenador)
#### apartado a
v <- c(6, 3, 15, 0, 9, 1, -2)
maximo <- function (v) {
Max <- v[1]
N<-length(v)
for (i in 2:N){
      if (v[i]>Max) {Max <-v[i]}}
return(Max)
}
maximo(v)
```

```
> #### apartado a
> v <- c(6, 3, 15, 0, 9, 1, -2)
> maximo <- function (v) {
+ Max <- v[1]
+ N<-length(v)
+ for (i in 2:N){
+ if (v[i]>Max) {Max <-v[i]}}
+ return(Max)
+ }
> maximo(v)
[1] 15
>
```

```
#### apartado b

###
v <- c(6,5,15,3,5,3,0,5,0,0,1,6,6,5,0)
frecuencia <- function (v, x) {
Frec <- 0
N<-length(v)
for (i in 2:N){
      if (v[i]==x) (Frec <-Frec+1)}
return(Frec)
}
frecuencia(v, 5)
```

```
> v <- c(6,5,15,3,5,3,0,5,0,0,1,6,6,5,0)
> frecuencia <- function (v, x) {
+ Frec <- 0
+ N<-length(v)
+ for (i in 2:N){
+ if (v[i]==x) (Frec <-Frec+1)}
+ return(Frec)
+ }
> frecuencia(v, 5)
[1] 4
>
```

```
#### apartado c

# Tengo un vector v
m<-maximo(v); m
f<-frecuencia(v,m); f
sprintf("El maximo de v es: %d y aparece %d veces en el vector", m, f)
```

```
> #### apartado c
>
> # Tengo un vector v
> m<-maximo(v); m
[1] 15
> f<-frecuencia(v,m); f
[1] 1
> sprintf("El maximo de v es: %d y aparece %d veces en el vector", m, f)
[1] "El maximo de v es: 15 y aparece 1 veces en el vector"
>
>
```