Curso de Ingreso - Módulo Programación

Contents

Guía de ejercicios

1. Conceptos básicos de los programas imperativos
2. Estructuras de control - Gráficos

1 Introducción

Esta es la guía de ejercicios correspondiente a la clase 01 (ver diapositivas). Deberá entregar todos los ejercicios resueltos en un archivo .R. Cada ejercicio debe estar resuelto entre comentarios que indique secciones dentro del archivo (en esta ocasión pueden usar el siguiente archivo como *template* para resolverlo: template-01.R).

Algunos de los temas necesarios para resolver esta guía no fueron incluidos en la teórica, muchos se encuentran en este documento, mientras que otros deberán ser investigados (por ejemplo, buscando en internet). El ejercicio de buscar cómo abordar/resolver problemas en internet es *casi tan* importante como poder resolverlos.

Además de escribir los programas pedidos deberán probarlos y dejar constancia de las pruebas realizadas, además de explicitar si anduvo como era esperado o no.

Ejercicios

1. Se desea tener un programa que dada la variable `grados`, que representa la temperatura en grados Farenheit, calcule en otra variable el valor en Celsius. Más info https://www.lmgtfy.es/?q=formula+formula+fahrenheit+a+celsius.

```
grados <- 90

enCelsius <- (grados - 32) * 5/9

enCelsius
```

```
## [1] 32.22222
```

2. Escribir otro programa que se comporte a la inversa, es decir, que dada una variable que represente la temperatura en Celsius, calcule su equivalente en Farenheit.

```r
grados <- 25

enFarenheit <- grados * 9/5 + 32

enFarenheit
```

```
## [1] 77
```

3. Escribir un conversor de kilometros a millas.

```r
unosKilometros <- 50

unasMillas <- unosKilometros/1.609

unasMillas
```

```
## [1] 31.0752
```

4. Dado un cuadrado, que el largo de su base se encuentra guardado en una variable llamada `base`, calcular:
   a. El perímetro

```r
base <- 30

perimetro <- base * 4

perimetro
```

```
## [1] 120
```

   b. El área

```r
base <- 30

area <- base * base

area
```

```
## [1] 900
```

5. Idem anterior pero para un triangulo equilatero.

```r
base <- 30

perimetro <- base * 3

perimetro
```

```
## [1] 90
```

```r
base <- 30

area <- sqrt(3)/4 * base^2

area
```

```
## [1] 389.7114
```

6. Asumiendo que los años tienen siempre 365 días, calcular:
   a. cuántos días vas a cumplir tu próximo cumpleaños,

```r
edad <- 27

dias <- 27 * 365

dias
```

```
## [1] 9855
```

b. cuántas horas vas a haber vivido,

```r
horas <- dias * 60

horas
```

```
## [1] 591300
```

c. y cuántos segundos.

```r
segundos <- horas * 60

segundos
```

```
## [1] 35478000
```

7. Se tienen las notas de 3 materias en sus respectivas variables: `matematica`, `lengua`, `dibujo`. Calcular el promedio de dichas notas.

```r
matematica <- 6
lengua <- 7
dibujo <- 8

suma <- matematica + lengua + dibujo

promedio <- suma / 3

promedio
```

```
## [1] 7
```

8. Repetir el ítem anterior, pero ahora con los valores guardados en un vector llamado `notas`. Hint: se puede acceder a los elementos de un vector con `[]`. Ej: `c(4, 6, 88)[2]` nos da el valor `6`.

```r
notas <- c(6, 7, 8)

suma <- notas[1] + notas[2] + notas[3]

promedio <- suma / 3

promedio
```

```
## [1] 7
```

9. Si en el ítem anterior no usaste la función `lenght()` y `sum()`, volvé a resolverlo usandolas.

```r
notas <- c(6, 7, 8)

suma <- sum(notas)
cantidad <- length(notas)

promedio <- suma / cantidad
```

```
promedio
```

```
## [1] 7
```

10. Si en el ítem anterio al anterior no usaste la función `mean()`, volvé a resolverlo usandola.

```
promedio <- mean(notas)

promedio
```

```
## [1] 7
```

11. Dadas dos variables, `perro` y `gato` escribir un programa que intercambie los valores de ambas variables.

```
perro <- 70
gato <- -15

temporal <- perro
perro <- gato
gato <- temporal
```

```
perro
```

```
## [1] -15
```

```
gato
```

```
## [1] 70
```

```
# opcion sin variables extras
perro <- 70
gato <- -15

perro <- perro + gato
gato <- perro - gato
perro <- perro - gato
```

```
perro
```

```
## [1] -15
```

```
gato
```

```
## [1] 70
```

12. Calcular el índice de masa corporal (IMC) de una persona cuya altura es `1.78`m y su peso es `80`kg. (`IMC = peso / altura^2`)

```
peso <- 80
altura <- 1.78

imc <- peso / altura^2

imc
```

```
## [1] 25.24934
```

13. Si tenemos los pesos y las alturas de personas en 2 vectores, calcular el IMC para cada uno.

```
pesos <- c(80, 70, 75, 94, 67)
alturas <- c(1.68, 1.75, 1.85, 1.90, 1.68)
```

```r
imcs <- pesos / alturas^2

imcs
```

```
## [1] 28.34467 22.85714 21.91381 26.03878 23.73866
```

14. Sobre el cálculo del ejercicio anterior, encontrar el valor máximo, el mínimo, el promedio y la mediana de los IMCs.

```r
max(imcs)
```

```
## [1] 28.34467
```

```r
min(imcs)
```

```
## [1] 21.91381
```

```r
mean(imcs)
```

```
## [1] 24.57861
```

```r
median(imcs)
```

```
## [1] 23.73866
```

2 Estructuras de control - Gráficos

Esta es la guía de ejercicios correspondiente a la clase 02 (ver diapositivas). Deberá entregar al menos todos los ejercicios indicados con una estrella (★) en un archivo .R. Igualmente se recomienda realizar *todos* los ejercicios para ganar mayor habilidad en programación. Cada ejercicio debe estar resuelto entre comentarios que indique secciones dentro del archivo.

Algunos de los temas necesarios para resolver esta guía no fueron incluidos en la teórica, muchos se encuentran en este documento, mientras que otros deberán ser investigados (por ejemplo, buscando en internet). El ejercicio de buscar cómo abordar/resolver problemas en internet es *casi tan* importante como poder resolverlos.

Además de escribir los programas pedidos deberán probarlos y dejar constancia de las pruebas realizadas, además de explicitar si anduvo como era esperado o no.

2.1 Condicionales

(1) Completar el siguiente programa de manera tal que el valor de la variable `a_es_mas_grande` sea `TRUE` unicamente cuando la variable `a` es más grande que `b` y `FALSE` en caso contrario.

```r
a_es_mas_grande <- COMPLETAR
```

(2) ★ Completar el siguiente programa de manera tal que el valor de `nombre_mas_grande` sea el nombre de la variable cuyo valor es más grande entre `a` y `b`, en caso de ser iguales devolver cualquiera (puede ser siempre el mismo).

```r
if (COMPLETAR) {
  nombre_mas_grande <- COMPLETAR
} else {
  COMPLETAR
}
```

(3) Modificar el programa anterior para que en caso de que sean iguales devuelva `"iguales"`

```r
if (COMPLETAR) {
  nombre_mas_grande <- COMPLETAR
```

```
} else {
  if (COMPLETAR) {
    COMPLETAR
  } else {
    COMPLETAR
  }
}
```

(4) ★ Escribir un programa que dado el valor guardado en la variable `q`, si dicho valor no es múltiplo de 3, entonces lo múltiplique por 3. En caso de serlo no debe modificarse.

(5) ★ Completar el programa para que calcule un paso de la función de Collatz. Recordar que dice:

$$collatz(n) = \begin{cases} n/2 & \text{si n es par} \\ (3n+1) & \text{si no} \end{cases}$$

```
if (COMPLETAR) {
  nuevo_n <- COMPLETAR
} else {
  COMPLETAR
}
```

(6) ★ Dados 2 vectores de números `v1` y `v2` concatenarlos en una variable llamada `res` en el siguiente orden:

- si el primer elemento de `v1` es menor que el primero de `v2`, entonces `v1` y luego `v2`
- sino al revés

(7) ★ Dados 2 vectores de números `v1` y `v2` y una variable llamada `cuenta` escribir un programa que calcula en la variable `res` los siguiente:

- si `cuenta` vale la cadena de caracteres "promedio", el promedio de todos los valores de `v1` y `v2`,
- si `cuenta` vale la cadena de caracteres "minimo", el mínimo entre todos los valores de `v1` y `v2`,
- si `cuenta` vale la cadena de caracteres "minimo1", el mínimo entre todos los valores de `v1`,
- para cualquier otro valor devolver la suma de todos los valores.

(8) Dados 2 vectores de números `v1` y `v2` concatenarlos en una variable llamada `res` de manera tal que el que primero que aparezca sea aquel cuya suma de elementos sea menor o igual que el otro.

2.2 Ciclos

(9) ★ Completar el programa para que dado un vector de 10 posiciones llamado `v` cuente la cantidad de posiciones i cuyo valor es exactamente i.

```
res <- 0
for (i in 1:10){
  if (v[i] == COMPLETAR) {
    res <- res + COMPLETAR
  }
}
```

(10) Modificar el programa anterior para que acepte vectores de cualquier longitud. Pista: puede usar la función `length` para obtener la longitud del vector.

(11) ★ Modificar el programa anterior para que además, en caso de que no cumpla con que la posición i valga i, reemplace dicho valor por un cero.

(12) ★ El siguiente programa recorre un vector `v` hasta encontrar un elemento que cumpla con tener como valor la posición. Dicho valor queda en la variable `i`. Experimentar con este programa teniendo en cuenta 2 casos: que existe y que no una posición que cumpla.

```
i <- 1
while (v[i] != i){
  i <- i + 1
}
```

(13) Arreglar el programa anterior para que en caso de que no exista, el programa termine sin dar error. Pista: antes de la condición diga `v[i]` debemos asegurarnos que `i` es una posición válida, es decir, es menor o igual que la longitud del vector.

```
i <- 1
while (COMPLETAR & v[i] != i){
  i <- i + 1
}
```

(14) Podemos verificar si existe o no dicho elemento en el ejercicio anterior mirando el valor de `i`. Si el valor es una posición válida de `v` entonces sí existe. Completar el programa para que en una variable `existe_i` determine con `TRUE/FALSE` la existencia de dicho valor.

```
i <- 1
while (COMPLETAR & v[i] != i){
  i <- i + 1
}

existe <- FALSE
if (COMPLETAR) {
  existe <- TRUE
}
```

(15) ★ Dados dos dados (de 6 caras que valen de 1 a 6), uno azul y uno rojo, calcular con un programa cuantas combinaciones posibles de valores hay. Pista: debería dar 6*6=36. ;-)

```
contar <- COMPLETAR
for (dado_azul in 1:6){
  for (dado_rojo in COMPLETAR){
    contar <- contar + COMPLETAR
  }
}
```

(16) ★ Con el mismo par de dados, calcular con un programa cuantas combinaciones tienen los 2 dados iguales.

(17) Con el mismo par de dados, calcular con un programa cuantas combinaciones suman 10.

(18) ★ Dado un vector de los primeros n naturales, escribir un programa que calcule para cada elemento el cuadrado de dicho valor en un nuevo vector.

```
valores_n <- COMPLETAR
valores_cuadrado <- c()

for (i in COMPLETAR) {
  valores_cuadrado <- c(valores_cuadrado,  COMPLETAR)
}
```

(19) Dado un vector de los primeros n números pares naturales, escribir un programa que calcule para cada elemento el cuadrado de dicho valor, si es múltiplo de 4, sino que lo divida por 2.

2.3 Gráficos

(20) Dados los valores calculados en el ejecicio anterior, graficarlos con el comando `plot`

(21) ★ Escribir un programa que calcule y grafique los primeros 10 términos de las siguientes sucesiones:

a. $a_n = \frac{1}{\sqrt{n}} + \left(\frac{1}{2}\right)^n$

b. $d_n = (-1)^{n+5}$

(22) ★ Retomando el ejercicio de las combinaciones de los dados, escribir un programa que inicie con un vector llamado `cantidad` de doce posiciones en cero (explorar el comando `rep`) y que para cada posición calcule cuántas combinaciones de los dados suman el valor indicado por la posición (ej. `cantidad[11]` vale 2, que son: (5,6) y (6,5)). Graficar los datos obtenidos.

(23) Hacé un gráfico que refleje la evolución de la temperatura del agua a lo largo del tiempo atendiendo a la siguiente descripción:

Saqué del fuego una cacerola con agua hirviendo. Al principio, la temperatura bajó con rapidez, de modo que a los 5 minutos estaba en 60 grados. Luego, fue enfriándose con más lentitud. A los 20 minutos de haberla sacado estaba en 30 grados y 20 minutos después seguía teniendo algo más de 20 grados, temperatura que se mantuvo, pues era la temperatura que había en la cocina.

(24) ★ Graficar la función $\rho(x) = x^2$, para $x \in [-10, 10]$.

```
x <- seq(COMPLETAR, COMPLETAR, 0.01)
valores <- c()

for (i in x){
  nuevo_valor <- COMPLETAR
  valores <- COMPLETAR
}

plot(n, valores, type = "p")
```

(25) Graficar la función $\rho(x) = |x|$, para $x \in [-10, 10]$.

(26) ★ Considerar $\rho_k : \mathbb{R} \to \mathbb{R}$, definida de la siguiente manera

$$\rho_k(x) = \begin{cases} x^2 & \text{si} \quad |x| \leq k \\ 2k|x| - k^2 & \text{si} \quad |x| > k \end{cases}$$

Calcular y graficar la la función ρ_k, con $k = 5$, para $x \in [-10, 10]$.

(27) En el mismo gráfico agregar el valor de la función ρ_k para $k = 2$ y $k = 8$ utilizando un color diferente para cada valor de k