Capítol 1

Repàs: Funcions, Vectors i Esquemes de Recorregut i Cerca

Objectius: Recordar i enfortir els coneixements adquirits a l'assignatura d'Introducció a la Informàtica, que seran bàsics per a poder continuar amb aquesta assignatura.

Primer de tot:

- Creeu una carpeta (o working directory) que es digui `Repas` on emmagatzemareu tots els scripts que codifiquen les diferents funcions d'aquesta part.
- Recordeu d'anar a aquest directori utilitzant `setwd("..../Repas")`
- Aconsellem quardar cada funció dins el directori `Repas` en un fitxer que es digui igual que la funció seguit d'un punt i `R`. Per exemple si la funcio es diu `mi_fu` enmmagatzemeu en un script `mi_fu.R`.

1.1 Exercicis

1. Fes una funció que donat un vector d'enters, retorni quin és el primer múltiple de 3 que hi ha a la seqüència, o 0 si no hi ha cap.
2. Dissenya una funció que donat un vector d'enters retorni un altre vector d'enters amb els elements del primer en ordre invers.
3. Dissenya una funció que donats dos vectors d'enters v1 i v2 ordenats, retorni un tercer vector que conté tots els elements de v1 i v2 també ordenats (si hi ha repetits es repeteixen també en el vector resultat).
4. Fes una funció que donat un vector d'enters i un enter, retorni TRUE si i només si l'enter donat és divisor d'algun dels elements del vector.

5. Fes una funció que donades dues cadenes de caràcters que representen dues paraules o frases diferents, retorni TRUE si aquestes dues paraules són bifronts o falsos palíndroms, és a dir que es llegeixen igual una del dret i l'altra del revés. Per exemple: roma - amor.

6. Fes una funció que donades dues cadenes de caràcters que representen dues paraules, retorni TRUE si aquestes dues paraules són anagrames. Una paraula és anagrama d'una altra si i només si, els caràcters que tenen totes dues paraules són els mateixos, encara que estiguin ordenats de forma diferent. Per exemple: roma - amor - omar - mora, són anagrames totes 4 entre sí, i arbol - labor - borla, també ho són.

7. Tenim la informació dels primers i segons classificats al Campionat Nacional de Lliga de futbol des dels seus inicis (any 1928) en diversos retalls de paper. Però aquesta informació, encara que cronològicament és correcta: sabem qui va ser el primer campió de la primera lliga, i després apareix en algun moment el campió de la 2a lliga abans del campió de la 3a lliga, però estan barrejats amb els subcampions.

   Així doncs l'ordre de les lligues es manté. Primer apareix la informació del campió o subcampió d'una lliga abans de les dades de la següent lliga, però estan barrejats els campions i subcampions.

   Feu un programa que, donada una seqüència de les dades de les lligues, en format de nom de l'equip i posició (1 o 2), que acabarà en un equip fictici "FI", s'ha de generar la taula dels campions i subcampions de lliga de forma cronològica però aparellant correctament els campions i els subcampions del mateix any. La informació és correcta i hi ha el mateix nombre de campions i de subcampions.

   Per exemple, si l'entrada fos:

```
Barça 1
Ath-Bilbao 2
Madrid 2
At-Madrid 2
València 1
Espanyol 1
At-Madrid 1
Barça 2
FI
```

   El resultat seria:

```
1er i 2on CLASSIFICATS DE LES LLIGUES
-------------------------------------
Campions         Subcampions
--------         ------------
Barça            Ath-Bilbao
València         Madrid
Espanyol         At-Madrid
At-Madrid        Barça
```

8. Escriu una funció que, donat un vector d'enters, escriu per pantalla, en ordre invers a com arriben, tots els enters que siguin senars havent-los multiplicat abans pel senar mínim dels nombres del vector. Per exemple:

   Si el vector és: `3 5 4 8 7 9 10 12 5 2 6 31 22 3 9 0`

   El resultat a la sortida ha de ser: `27 9 93 15 27 21 15 9` *(ja que el mínim senar és el 3)*

9. *Quicksort.* (Un clàssic). La funcio `qs(v)` ha d'ordenar una taula de nnombres, per exemple

```
> v<-c(1,5,2,3,4,7,10,9, 6)
> v
[1]  1  5  2  3  4  7 10  9  6
> qs(v)
[1]  1  2  3  4  5  6  7  9 10
```

   Us demanem que completeu el codi:

```
qs <- function(vec){
  if(length(vec) > 1){
    pivot <- vec[1]
    low <- qs(vec[vec < pivot])
    mid <- vec[vec == pivot]
    high <- ....)
    c(low, ..., ....)
  }
  else vec
}
```

   Recordeu que cal emmagatzemar la funció dins `Repas` en un scrip (que recomenem) es digui `qs.R`