# Examen parcial de Programació 2015-2016

## 5/4/2016 8:30 Durada: 1:40

## 1 Exercici 1: Taulell anagramàtic *(4 punts)*

Un **anagrama sintàctic** d'una seqüència de lletres $p$ és una altra seqüència de lletres $p'$ que resulta de la transposició de les lletres que apareixen en $p$. Fixeu que, segons aquesta definició, tota seqüència de lletres és anagrama de sí mateixa. Per exemple:

```
AMOR - ROMA - OMAR - MORA - RAMO - ARMO - MARO - ORAM
ARBOL - LABOR - BORLA
MONJA - JAMON - MOJAN
ESPONJA - JAPONES
ALEXIS - XISELA
```

Un **taulell anagramàtic** associat a una seqüència de lletres $p$ de llargada $k$ és una matriu de $k \times k$ lletres de forma tal que a cada fila i a cada columna hi apareix un anagrama de $p$. Per exemple, en els dos casos següents (a i b) tindríem

```
a)  ROMA          b)  LABOR
    AMOR              BROAL
    MARO              RBROA
    ORAM              AOLBR
                      OLARB
```

el cas `a)` és un taulell anagramàtic associat a la seqüència de lletres "AMOR", nentre que el cas `b)` no és un taulell anagramàtic associat a la seqüència de lletres "ARBOL"perquè les columnes 4 i 5, per exemple, no són anagrames de "ARBOL".

Escriu una funció amb que donats un vector de lletres `p` i una matriu qualsevol `t`, retorni TRUE si `t` és un taulell anagramàtic associat a `p` i FALSE en cas contrari.

Per a resoldre l'exercici podeu fer servir la funció:

```
1  #precondicio:
2  #p1 i p2 son vectors de lletres majuscules amb la mateixa llargada
3  anagrames ← function(p1,p2) {
4    n ← length(p1)
5    p1sort ← sort(p1)
6    p2sort ← sort(p2)
7    iguals ← TRUE
8    i ← 1
9    while ((i≤n) && (iguals) {
10     if (p1sort[i] ≠ p2sort[i]) {
11       iguals ← FALSE
12     }
13     i ←  i + 1
14   }
15   return (iguals)
16 }
```

**Solució 1** Solució que fa servir dos *whiles* un per a comprovar totes les files i l'altre per a comprovar totes les columnes.

```
1  taulellanagramatic ← function(p,t){
2    k ← length(p)
3    es_anag ← TRUE
4    if (nrow(t) == k && ncol(t) == k) {
5      #comprova files
6      i ← 1
7      while(i ≤ nrow(t) && es_anag){
8        es_anag ← anagrames(vp,t[i,])
9        i ← i+1
10     }
11     #comprova columnes
12     i ← 1
13     while(i ≤ ncol(t) && es_anag) {
14       es_anag ← anagrames(vp,t[,i])
15       i ← i+1
16     }
17   }
18   else {
19     es_anag ← FALSE
20   }
21   return (es_anag)
22 }
```

**Solució 2** Solució que només fa servir un *while* i a la i-èssima iteració comprova si la fila i la columna `i` són anagrames.

```
1  taulellanagramatic ← function(p,t){
2    k ← length(p)
3    es_anag ← TRUE
4    if (nrow(t) == k && ncol(t) == k) {
5      i ← 1
6      while(i ≤ nrow(t) && es_anag){
7        #comprova la fila i la columna i
8        es_anag ← anagrames(vp,t[i,]) && anagrames(vp,t[,i])
9        i ← i+1
10     }
11   }
12   else {
13     es_anag ← FALSE
14   }
15   return (es_anag)
16 }
```

## 2 Exercici 2: Diccionari nou *(3 punts)*

Disposem d'un diccionari que se'ns ha quedat antiquat, i volem canviar les seves definicions per les més noves. Per a això, comparem el nostre diccionari amb un de nou però només volem modificar les paraules que ja teníem, no volem afegir-ne més.

Així doncs, es demana que feu una funció que donats dos lists (`L1` i `L2`) on `L1` és el nostre diccionari i `L2` és el nou, es modifiqui el list `L1` (el nostre diccionari) amb les definicions noves (les de `L2`) però no afegirem aquelles paraules que només estan en `L2` i no en `L1`.

Per exemple:

```
L1 <- list (casa = "lloc on viure",
            lloguer = "diners a pagar per un servei",
            mitjana = "valor promig",
            moneda = "tros de metall amb valor, cada país en té una diferent",
            sabata = "tros de material per a cobrir el peu")
L2 <- list (casa = "vivenda, habitatge, lloc on viure",
            llibre = "conjunt de pàgines per llegir",
            moneda = "tros de metall amb valor",
            sabata = "calçat que cobreix el peu",
            videojoc = "joc electrònic")
```

el resultat ha de ser:

```
L1 <- list (casa = "vivenda, habitatge, lloc on viure",
            lloguer = "diners a pagar per un servei",
            mitjana = "valor promig",
            moneda = "tros de metall amb valor",
            sabata = "calçat que cobreix el peu")
```

```
1  NouDiccionari ← function (L1, L2)
2  {
3      noms ← names(L2)
4      for (n in noms) {
5          if (!is.null(L1[[n]])) {
6              L1[[n]] ← L2[[n]]
7          }
8      }
9      return (L1)
10 }
```

## 3 Exercici 3: Què calcula la funció `funcio`? *(3 punts)*

Tenim la següent funció de matrius ja implementada:

```
funcio ← function(mat) {
    nf = nrow(mat)
    nc = ncol(mat)
    sortida = vector()
    for (i in 1:nc) {
        a ← 0
        b ← mean(mat[,i])
        for (j in 2:nf) {
            if (mat[j,i] < b) {
                a ← a + 1
            }
        }
        sortida = c(sortida, a)
    }
    return (sortida)
}
```

Si la matriu d'entrada d'aquesta funció és la següent:

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 12 | 22 | 5 | 12 | 4 | 3 |
| 15 | 20 | 8 | 15 | 2 | 11 | 2 |
| 7 | 8 | 10 | 10 | 2 | 5 | 15 |
| 8 | 0 | 2 | 10 | 2 | 7 | 10 |
| 10 | 5 | 3 | 0 | 2 | 33 | 10 |

quin serà el vector sortida de la funció?

1 3 3 1 4 3 1