Poseu el nom a tots els fulls
Entregueu els problemes en fulls separats
Les respostes han de ser justificades

Problema 1 *3 punts* Temps estimat: 30 min

L'empresa CONTROL S.A. vol conèixer quin dels seus empleats ha treballat més hores al llarg d'un periode de temps. Per això, ens demana un algorisme que a partir de les dades introduïdes per un operador ens tregui pel canal estàndar de sortida el nom de l'empleat amb més hores treballades, i quantes hores de més ha fet respecte de la mitjana d'hores treballades per tots els empleats.

Un operador introduirà les dades en un computador posant per cada empleat, el seu nom (com a molt 30 caràcters), seguit d'una seqüència de dades provinents d'un rellotge de control de presència que indica les seves entrades i sortides a l'empresa. La seqüència acabarà amb un empleat sentinella.

Dissenyeu l'algorisme demanat considerant que les següents accions i funcions ja estan dissenyades.
Notes: En el cas poc probable de coincidència d'hores, l'empleat afortunat serà el primer. Com a mínim hi haurà un empleat vàlid en la seqüència.

acció $llegirEmpleat($**sor** $e : tEmpleat)$
{**Pre:** $s \equiv e_1 \dots e_n e_f$, $pe(s) = \alpha$ i $\alpha = e_1 \dots e_{i-1}$, $pd(s) = e_i \cdot \beta$, i $\beta = e_{i+1} \dots e_f$, $1 \leq i \leq f$ }
{**Post:** e representa e_i, i $pe(S) = \alpha \cdot e_i$ i $pd(S) = \beta$}

funció $empleatSentinella($**ent** $e : tEmpleat)$ **retorna booleà**
{**Pre:** $e = E$}
{**Post:** empleatSentinella(e) és cert si E és l'empleat sentinella. Fals en cas contrari.}

acció $escriureEmpleat($**ent** $e : tEmpleat)$
{**Pre:** $e = E$}
{**Post:** Pel CSS surt N i H, on N és el nom de l'empleat E i H són les hores treballades d'E.}

funció $hores($**ent** $e : tEmpleat)$ **retorna real**
{**Pre:** $e = E$}
{**Post:** hores(e) és el nombre d'hores treballades per l'empleat E}

Problema 2 *4 punts* Temps estimat: 30 min

Un concessionari de cotxes de diverses marques i models d'automòbils vol conèixer la seva facturació anual. Per fer-ho disposa de dos fitxers. El primer, anomenat "vendes.dat', conté una seqüència de parelles on cadascuna conté el nom del model d'un cotxe, i el nombre de cotxes venuts. El segon fitxer, "llistaDePreus.dat", conté també una seqüència de parelles on cadascuna conté el nom del model d'un cotxe, i el seu preu de venda al públic. Les parelles d'ambdós fitxers estan ordenades segons el nom de models de cotxe. Ambdós fitxers acaben amb una parella que té com a model un mot sentinella.

Es demana que dissenyeu un algorisme eficient que donats el fitxer "vendes.dat" i el fitxer "llistaDePreus.dat", escrigui la facturació total de les vendes efectuades. El fitxer de vendes només podrà contenir models de cotxe ja existents en el fitxer de preus.

Podeu comptar amb les accions:

acció $llegirVendaFST($**entsor** $f : FST,$ **sor** $model : tMot,$ **sor** $n :$ **enter**$)$
{**Pre:** f es un FST de parelles de model cotxe i quantitat, i $mode(f) = lectura \ \wedge \ pe(f) = \alpha \ \wedge \ en\_curs(f) = \beta \ \wedge \ pd(f) = \gamma \cdot \delta \ \wedge \ pd(f) \neq \emptyset$}
{**Post:** $mode(f) = lectura \ \wedge \ pe(f) = \alpha.\beta \ \wedge \ en\_curs(f) = \gamma \ \wedge \ pd(f) = \delta \ \wedge$ m i n representen el model de cotxe i la quantitat respectivament de la parella γ, i $n \geq 0$}

acció $llegirPreuFST($**entsor** $f : FST,$ **sor** $model : tMot,$ **sor** $p :$ **real**$)$
{**Pre:** f es un FST de parelles de model cotxe i preu, i $mode(f) = lectura \ \wedge \ pe(f) = \alpha \ \wedge \ en\_curs(f) = \beta \ \wedge \ pd(f) = \gamma \cdot \delta \ \wedge \ pd(f) \neq \emptyset$}
{**Post:** $mode(f) = lectura \ \wedge \ pe(f) = \alpha.\beta \ \wedge \ en\_curs(f) = \gamma \ \wedge \ pd(f) = \delta \ \wedge$ m i p representen el model de cotxe i el preu respectivament de la parella γ, i $p \geq 0$}

funció $modelSentinella($**ent** $m : tMot)$ **retorna booleà**
{**Pre:** cert}
{**Post:** $modelSentinella(m)$ és cert si m és el model de cotxe sentinella. Si no ho és, retorna fals}

funció $motsIguals($**ent** $m1 : tMot,$ **ent** $m2 : tMot)$ **retorna booleà**
{**Pre:** $m1 = M1$ i $m2 = M2$}
{**Post:** $motsIguals(m1, m2)$ és cert si i només si $M1$ i $M2$ representen el mateix mot.}

funció $mot1AbansMot2($**ent** $m1 : tMot,$ **ent** $m2 : tMot)$ **retorna booleà**
{**pre:** $m1 = M1$ i $m2 = M2$}
{**post:** $mot1AbansMot2(m1,m2)$ és cert si i només si $M1$ va alfabèticament abans que $M2$ i $M1 \neq M2$.}

Per exemple, si els continguts dels fitxers són:

"vendes.dat"
Focus 40
Golf 50
Mondeo 30

"llistaDePreus.dat"
Astra 13000
Focus 10000
Galaxy 30000
Golf 18500
Laguna 8600
Leon 15000
Mondeo 12500
Neon 18000
Polo 9000
Serena 12000
Voyager 35000
Z3 12000

L'algorisme escriurà la següent sortida: 1700000

Problema 3 *3 punts* Temps estimat: 30 min

Suposem que definim el tipus següent:

tipus
$taulaEnters =$ **taula** $[1..100000]$ **de enter**
ftipus

Considereu el següent subprograma:
{Prec: $1 \leq n \leq 100000$}

```
funció Nhiha(ent t : taulaEnters, ent n : enter) retorna booleà
  var
    i, j, sum : enter
    trobat : booleà
  fvar
  i := 1;
  trobat := fals
  mentre i ≤ n ∧ ¬trobat fer
    j := 1
    sum := 0
    mentre j ≤ i − 1 fer
      sum := sum + t[j]
      j := j + 1
    fmentre
    si
      sum = t[i] → trobat := cert
    □ sum ≠ t[i] → i := i + 1
    fsi
  fmentre
  retorna trobat
ffunció
```

Es demana:

a) Calcular la complexitat assimptòtica en el cas pitjor d'aquest algorisme.

b)Modificar l'algorisme, sense variar l'especificació, a fi que la seva complexitat es redueixi, i calculeu-la.