Para el lunes 2 de marzo, preparad el 1, 3, 6, 7, 8 y 9

1. [1 punt]

Considereu l'algorisme seg¨uent, camuflat misteriosament:

```
void misteri(vector<>& v, i)
{ (i==0) i=1; (i<v.size())
{ (v[i-1]<= v[i]) misteri(v,i+1); {
swap(v[i-1],v[i]);
misteri(v,i-1);
} } }

Void misteri(vector<>& v)
{
misteri(v,0);
}
```

Es demana:

a) Quin ´es el cas pitjor per a aquest algorisme?

b) Analitzeu el cost temporal del cas pitjor de la crida misteri(v).

c) Digueu en una frase qu`e fa aquest algorisme.

d) Us recorda a algun altre algorisme?

2. (Dividir i V`encer) [2 punts]

A la secci´o de bricolatge d'un gran magatzem ha arribat la darrera remesa de cargols i femelles que ha d'abastir el centre per a tot l'estiu. El material ha arribat en dues grans caixes: en una hi ha n cargols, tots d'una amplada diferent, i a l'altre hi ha les n femelles corresponents.

Per oferir el material cal posar-lo als prestatges d'una forma que sigui f`acil de trobar pels clients. Per aix`o, s'ha d'aparellar cada cargol amb la seva femella abans de col·locar-lo al corresponent prestatge.

Malauradament, quan l'encarregat de la secci´o es disposava a aparellar cada cargol amb la seva femella corresponent, hi ha hagut un tall de corrent que ha deixat tot el centre a les fosques. A causa de la foscor no es poden llegir les mides exactes dels cargols ni tampoc de les femelles. L'´unica comparaci´o possible ´es entre cargol i femella, intentant cargolar un cargol a una femella per comprovar si ´es massa gran, si ´es massa petit o si encaixa perfectament.

Es demana:

a) Descriviu un algorisme de dividir i v`encer per aparellar cada cargol amb la seva femella en _(n log n) passos en el cas mitj`a.

b) Quin ´es el cas pitjor del vostre algorisme?

Pista 1: Penseu en un algorisme d'ordenaci´o fam´os.

Pista 2: Un cargol es pot fer servir m´es d'una vegada per comparar-lo amb diverses femelles.

3.

(1.5 punts) Tenim una taula t amb n elements i desitgem saber si k_n altres elements són dins de la taula t o no. Hi ha, com a mínim, dues maneres possibles de procedir:

Opció 1: Per a cadascun dels k_n elements, fem una cerca lineal a la taula t.

Opció 2: Ordenem primer la taula t amb un algorisme $\Theta(n \log n)$ i, després, per a cadascun dels k_n elements, fem una cerca dicotòmica a la taula t.

Digueu quin ha de ser el valor mínim de k_n (utilitzant notació asimptòtica) perquè la segona opció sigui més ràpida o igual que la primera.

4.

(3 punts) Dissenyeu un algorisme de cost $O(n \log n)$ que donada una taula amb n elements compti el nombre d'índexos i, $1 \le i \le n$, tals que la taula conté exactament $t[i]$ elements (diferents o iguals) més petits que $t[i]$. Dit d'altra manera l'algorisme ha de comptar el nombre d'elements del conjunt

$$\{\ i\ :\ 1 \le i \le n \wedge |\{j\ :\ t[j] < t[i]\}|\ =\ t[i]\ \}$$

Per exemple si $t = [8, 6, 1, 2, 3, 3, 5, 6, 3, 7]$ la resposta és 3. Els índexos per als quals se satisfà la condició són 2, 7 i 8. Justifiqueu la correctesa del vostre algorisme i que, efectivament, el seu cost és $O(n \log n)$.

5.

(2.5 punts) Tenim una taula amb n elements i volem els k elements més petits de la taula en ordre creixent. Tenim dues opcions:

(a) Fem un max-heap amb els primers k elements i després per a cadascun dels $n-k$ elements restants, el comparem amb el màxim del heap i si l'element en curs fos més petit, eliminem el màxim del heap i inserim l'element en curs. D'aquesta manera el heap contindrà els k elements més petits visitats fins al moment, en tot moment. En acabar de recórrer la taula, podem extreure, un per un, tots els elements del heap per tenir-los en ordre creixent.

(b) Fem un min-heap amb els n elements de la taula i després fem k vegades extreure el mínim, per obtenir els k elements més petits en ordre creixent.

Quin és el cost en cas pitjor de la primera opció en funció de k i n? I el de la segona opció? Per a quins valors de k seria millor la primera opció a la segona? Per a quins valors de k és millor la segona opció que la primera? Quan són els seus costos en cas pitjor asimptòticament equivalents? Justifiqueu la vostra resposta.

6.

(1 punt) Ordena de menor a major taxa de creixement asintòtic les següents quatre funcions: $\log_{10} n$, $2^{\sqrt{\log n}}$, $n^{2/3} \log\log n$, $n^{1/4}$. La base dels logarismes és 2, si no s'indica el contrari.

7.

(1 punt) Tenim un algorisme A que per a entrades de talles $n = 64, 128, 256$ i 512 triga en executar-se 4096, 16384, 65536 i 262144 milisegons, respectivament. Suposem que el cost de l'algorisme és de la forma $T_A(n) = a \cdot n^k$ per a certes constants a i k.

(a) Quant triga l'algorisme A amb una entrada de talla $n = 1024$ si l'executem sobre un computador que té el doble de velocitat que el computador sobre el qual es van fer les mesures originals?

(b) Quant triga un algorisme A' amb cost $T_{A'}(n) = an^k/\log_2 n$ amb una entrada de talla $n = 1024$ si l'executem sobre el computador original?

8.

(1 punt) Tenim un algorisme A amb cost en cas pitjor $\Theta(\sqrt{n})$ i un altre B amb cost en cas pitjor $\Theta(2^{\sqrt{\log_2 n}})$. Quin del dos algorismes és més eficient? O tenen la mateixa eficiència asimptòtica? Justifica la teva resposta.

N.B. Sense justificació, la vostra resposta no serà avaluada.

9.

(1 punt) El cost $T(n)$ d'un algorisme recursiu satisfà $T(n) = xT(n/4) + \Theta(\sqrt{n})$, amb $x > 0$. Quin és el cost $T(n)$?

Pista: Hi ha tres situacions possibles, segons l'interval en el qual cau la x.

10.

(3 punts) Es diu que una seqüència $A = (a_1, \ldots, a_n)$ de longitud $n \geq 3$ és *unimodal* si existeix un índex p, $1 \leq p \leq n$, tal que $a_1 < a_2 < \cdots < a_p$ i $a_p > a_{p+1} > \cdots > a_n$, és a dir, creix fins a a_p i després decreix. A l'element a_p se l'anomena *pic*. Escriu un algorisme de dividir per vèncer que donat un vector que conté una seqüència unimodal trobi el pic de la seqüència, amb cost $O(\log n)$. No es valorarà cap solució amb cost superior a logarísmic. Justifica que el cost de l'algorisme proposat és $O(\log n)$ i justifica la correctesa de l'algorisme.