

Cognoms

Nom

DNI

Examen Final AP3

Duració: 3 hores

9/1/2019

-
- *L'enunciat té 6 fulls, 12 cares i 4 problemes.*
 - *Poseu el vostre nom complet i número de DNI a cada full.*
 - *Contesteu tots els problemes en el propi full de l'enunciat i a l'espai reservat.*
 - *A no ser que es digui el contrari, cal justificar les respostes.*
-

Problema 1

(1.5 punts)

Volem generar totes les permutacions de $\{1, 2, \dots, n\}$ tals que cap subseqüència de k nombres consecutius sumi més de s . Pots assumir que $n \geq k \geq 2$, $s \geq 0$. Per exemple, si $n = 4$, $k = 3$, $s = 7$, les úniques permutacions possibles són

$(3, 1, 2, 4), (3, 2, 1, 4), (4, 1, 2, 3), (4, 2, 1, 3)$

La permutació $(1, 3, 2, 4)$ no és vàlida perquè la subseqüència $(3, 2, 4)$ suma més de 7. Completa el següent codi:

```
// Escriu el vector sol a la sortida estàndard
void write_solution (vector<int>& sol);

// Si n = v.size() i p = min(k,n), retorna v[n-1] + v[n-2] + ... + v[n-p]
int sum_last_k(const vector<int>&v, int k);

void write_perm_max(int n, int k, int s, vector<int>& sol, vector<bool>& used) {
    if (sol.size() == n) write_solution (sol);
    else {
        for (int i = 1; i <= n; ++i) {

        }
    }
}
```

```

void write_perm_max(int n, int k, int s) {
    vector<bool> used(n+1, false);
    vector<int> sol; // Inicialment buit
    write_perm_max(n, k, s, sol, used);
}

int main(){
    int n, k, s;
    cin >> n >> k >> s;
    write_perm_max(n, k, s);
}

```

Cognoms**Nom****DNI****Problema 2****(2.5 punts)**

En Joan és fill d'una família que ha embogit davant la voràgine de consumisme en què s'han convertit les festes nadalenques. Pels $n \geq 1$ dies que duren aquestes festes, els seus pares han comprat n regals. Cada dia, li mostraran el regal corresponent a aquell dia, i en Joan el podrà acceptar o rebutjar. L'única restricció és que no pot acceptar el regal dos dies consecutius. Davant l'ansietat de no escollir prou bé, en Joan ha regirat tota la casa i ha trobat una llista (v_1, v_2, \dots, v_n) , on v_i és el valor del regal que li oferiran el dia i -èssim. En Joan, desesperat, ens demana ajuda: vol saber quina és la màxima suma de valors que pot aconseguir.

(a) (1 pt.) Completa el codi (recursiu) següent per solucionar aquest problema:

```
int max_valor (const vector<int>& v, int k) {  
    if (k < 0) return ;  
    else if (k == 0) return ;  
    else return max( ,  );  
}  
  
int main() {  
    int n;  
    cin >> n;  
    vector<int> v(n);  
    for (int i = 0; i < n; ++i) cin >> v[i];  
    cout << "Valor: " << max_valor(v, n-1) << endl;  
}
```

(b) (0.5 pts.) Troba la recurrència que descriu el nombre de vegades que s'executa algun dels returns de les dues primeres línies de *max_valor*, en funció de k . Usa la solució d'aquesta recurrència per calcular una fita del cost en temps, en funció de k , de *max_valor*. Finalment calcula una fita inferior del cost del programa principal.

Ajuda: et pot ser útil recordar que per $n \geq 0$ es té que F_n , el n -èssim nombre de la successió de Fibonacci $1, 1, 2, 3, 5, \dots$ compleix $F_n \in \Theta(\phi^n)$ on $\phi = \frac{1+\sqrt{5}}{2}$.

- (c) (1 pt.) Implementa una funció iterativa que solucioni el problema anterior de la manera més eficient possible, tant en temps com en espai. Analitza el cost en temps i espai de la teva solució.

```
int max_valor (const vector<int>& v) {
```

```
}
```

Anàlisi del temps i espai:

(3 punts)

Un conegut investigador visitarà la UPC el proper 30 de gener per tal d'impartir-hi una xerrada. Davant l'allau de peticions per assistir-hi, s'ha decidit que l'investigador repetirà la xerrada diverses vegades. Per tal de determinar quantes xerrades es faran i l'hora d'inici de cadascuna d'elles, hem preguntat a les n persones interessades l'interval de temps en el que els aniria bé que la xerrada comencés.

Disposem, així doncs, d'un conjunt d'interval·ls $I = \{ [s_i, t_i] \mid 1 \leq i \leq n \}$ i volem esbrinar el menor nombre de xerrades necessàries, i la seva hora d'inici, per tal que tothom pugui assistir a almenys una d'elles.

(a) (0.5 pts.) Considera el següent algorisme golafre:

$$P := I$$

mentre $P \neq \emptyset$

escull el nombre t inclòs en el major nombre d'interval·ls de P

(en cas d'empat escull el menor t)

planifica una xerrada a les t hores

elimina de P tots els intervals que inclouen t

fmentre

Demostra que aquest algorisme no sempre calcula la solució òptima, tot donant un contraexemple.

[illegible]

(b) (1.5 pts.) Considera el següent algorisme golafre:

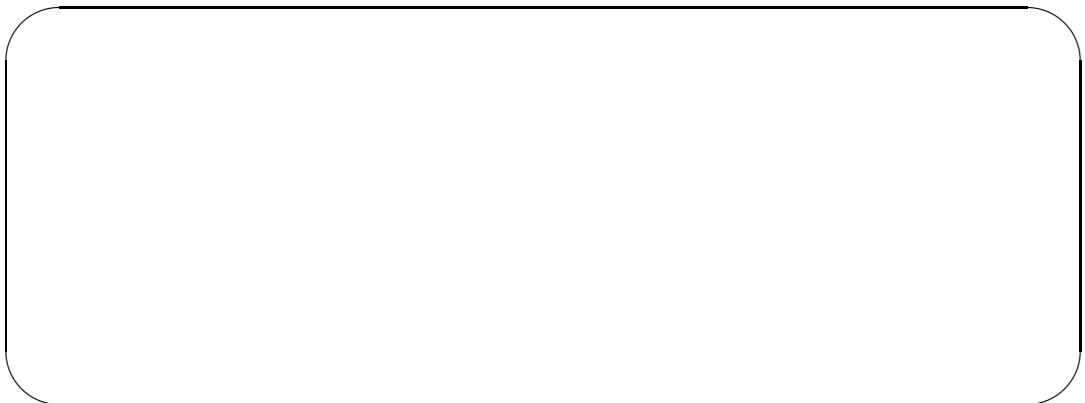
```
 $P := I$   
 $i := 1$   
mentre  $P \neq \emptyset$   
    escull  $[s, t] \in P$  amb mínim  $t$   
    planifica una xerrada a les  $t$  hores;  $t_i := t$   
    elimina de  $P$  tots els intervals que inclouen  $t$   
     $i := i + 1$   
fmentre
```

Siguin $\{t_1, t_2, \dots, t_g\}$ les hores en les que l'algorisme golafre anterior planifica les xerrades i considerem $t_1^* < t_2^* < \dots < t_\theta^*$ les hores, ordenades, d'una solució òptima.

- Demostra que $t_1 < t_2 < \dots < t_g$. Pots fer-ho per reducció a l'absurd.



- Demostra que $\theta \leq g$.



Cognoms

Nom

DNI

- Demuestra per inducció que per a tota $1 \leq i \leq \theta$ es compleix $t_i \geq t_i^*$.

- (c) (1 pt.) Demuestra que l'algorisme golafre de l'apartat anterior sempre produeix una solució amb el nombre mínim de xerrades.

Cognoms

Nom

DNI

Problema 4

(3 punts)

Com que s'apropen les eleccions municipals, l'ajuntament ha decidit obrir un centre cívic on les diferents associacions del municipi podran celebrar les seves reunions. Cadascuna de les n associacions existents pot fer una única petició de reserva per una certa franja, expressada com un interval de temps. Les peticions recollides formen el conjunt $I = \{[s_i, f_i] \mid 1 \leq i \leq n\}$. L'objectiu de l'ajuntament és reservar la sala al major nombre d'associacions, però tenint en compte que mai dues reserves acceptades poden solapar-se.

(a) (0.5 pts.) Considera el següent algorisme golafre:

$P := I$

mentre $P \neq \emptyset$

escull l'interval més curt $[s_k, t_k] \in P$ (en cas d'empat, el de menor índex k)

accepta la reserva k -èsima

elimina de P tots els intervals que interseccionen amb $[s_k, t_k]$

fmentre

Demostra que aquest algorisme no sempre calcula la solució òptima, tot donant un contraexemple.

- (b) (1.5 pts.) Demuestra que si s^* és el major nombre de reserves que es poden acceptar, i s és el nombre de reserves que accepta l'algorisme golafre anterior, aleshores $s \geq \frac{1}{2}s^*$.

Ajuda: sigui $S = (J_1, J_2, \dots, J_s)$ la seqüència d'interval·ls acceptats per l'algorisme anterior. És a dir, en la iteració i -èsima s'accepta l'interval J_i . Per $1 \leq i \leq s$, considerem $B_i = J_i \cup \{J \in I \mid J \text{ és eliminat de } P \text{ en la iteració } i\text{-èsima}\}$. Esbrina quants interval·ls dins de cada B_i poden aparèixer com a molt en una solució òptima.

Cognoms

Nom

DNI

- (c) (1 pt.) Es coneix algun algorisme polinòmic per solucionar de manera òptima aquest problema? Si es així, explica'l de manera informal sense demostrar la seva optimalitat ni analitzar el seu cost. Si no se'n coneix cap, raona per què.

Aquesta cara estaria en blanc intencionadament si no fos per aquesta nota.