Universitat	Politècnica	${\rm de}$	Catalunya
Facultat d'l	nformàtica	de	Barcelona

							C	OΡ	111	) [ [	ıs,	N	(O)	m									L	,	Ν.,	l.	

Titulació: EI/ETIG Curs: Q2 2006-2007 (2ºn Parcial)
Assignatura: Anàlisi i Disseny d'Algorismes Data: 25 de juny de 2007

Duració: 2 hores

1. (3 punts) Es vol emmagatzemar en un dispositiu de capacitat L un conjunt de n arxius. La talla de l'arxiu i-èsim ve donada per  $\ell[i]$ , i se sap que  $\sum_{i=1}^{n} \ell[i] > L$ . Implementeu en C++ o pseudocodi un algorisme voraç que seleccioni un subconjunt dels arxius tot maximitzant el nombre d'arxius que es poden emmagatzemar al dispositiu. Justifiqueu la correctesa del vostre algorisme i analitzeu el seu cost.

							С	90	n	on	ıs,	N	n									Ι	) [	N.I		

(Continueu responent aquí a la Pregunta 1.)

2. (4 punts) Un concurs consisteix en el següent: hi ha dos jocs de boles numerades des de 1 fins a N, un de boles negres i un de blanques; en total 2N boles. D'aquestes boles se'n col·loquen N, una de cada número, dins una capsa. Es a dir, hi ha una bola (blanca o negra) amb el número 1, una bola (blanca o negra) amb el número 2, etc. Cada concursant fa una aposta sobre el color de quatre de les boles de la capsa, indicant número i color (e.g., la 1 negra, la 7 blanca, la 11 blanca i la 12 negra). En total hi ha M concursants. L'organitzador del joc, que és un trampós, vol calcular, un cop conegudes les M apostes, una combinació de les boles de la capsa tal que ningú no encerti les seves quatre prediccions. Dissenyar un algorisme que la calculi o bé que l'indiqui que no existeix.

Utilitza l'esquema que es dona tot seguit, completant els espais marcats i implementant totes les operacions auxiliars que et facin falta.

```
enum { undef, blanca, negra } color;
struct bola {
       int n;
       color c;
};
// una aposta és un vector que defineix quatre boles
typedef vector < bola > aposta;
class concurs {
public:
      ^{\prime\prime} la constructora rep el vector amb M apostes (correctes)
      concurs(int \mathbb{N}, vector<aposta>& apostes) {
          N = N;
          _apostes = apostes;
          _hi_ha_solucio = false;
          _solucio = vector < color > (_N, undef);
             aquest espai el podeu omplir si us convé
         backtrack(0);
          // aquest espai el podeu omplir si us convé
```

```
}
      hi_ha_solucio() { return _hi_ha_solucio; }
bool
vector<color> solucio() {
    if (_hi_ha_solucio) return _solucio;
    else return vector < color > (_N, undef);
}
private:
      int _N;
      vector < aposta > _ apostes;
      bool _hi_ha_solucio;
      vector < color > _ solucio;
      void backtracking(int k);
      // afegiu mètodes i atributs privats aquí, si ho necessiteu
};
void concurs::backtrack(int k) {
     if
        _hi_ha_solucio = true;
        return;
     _solucio[k] = blanca;
     if
        backtrack(k + 1);
     if
         _solucio[k] = negra;
         if
            backtrack(k + 1);
     }
}
```

							С	og	n	n	ıs.	N	Oı	m									Ι	).[	N.I		

(Continueu responent aquí a la Pregunta 2.)

							С	90	gno	on	ıs,	N	Vo:	m	
7												$\overline{}$			г

D.N.I.

3. (3 punts) S'anomena arbitratge a l'ús de les discrepàncies en les taxes de canvi de divises per a obtenir un benefici. Per exemple, durant un breu espai de temps pot succeïr que un euro (1 €) valgui 0.95 dòlars, un dòlar (1 \$) valgui 0.75 lliures esterlines i una lliura esterlina (1 £) valgui 1.45€. Llavors, començant amb 1 € podem acabar amb:

 $1 \in \times \frac{0.95\$}{1 \in \mathbb{R}} \times \frac{0.75 \pounds}{1\$} \times \frac{1.45 \in \mathbb{R}}{1 \pounds} = 1.033125 \in \mathbb{R}$ 

amb un benefici net d'una mica més del 3.3%. Suposeu que hi ha n tipus diferents de divises i que tenim una matriu C[1..n, 1..n] on C[i, j] és la taxa de canvi de (una unitat de) la divisa i a la divisa j. Per exemple si l'euro és la divisa 1 i la lliura esterlina la divisa 2 llavors C[2, 1] = 1.45. Es compleix que C[i, j] = 1/C[j, i] per a tota i i j, i que C[i, i] = 1.

Implementa en C++ o pseudocodi un algorisme de programació dinàmica que donada la matriu C i l'identificador i d'una divisa,  $1 \le i \le n$ , ens proporcioni el valor del millor arbitratge, on el millor arbitratge és una seqüència  $\langle i_1, i_2, \ldots, i_k \rangle$  d'identificadors diferents entre sí i diferents d'i tal que

$$C[i, i_1] \cdot C[i_1, i_2] \cdot \cdot \cdot C[i_{k-1}, i_k] \cdot C[i_k, i]$$

és màxim. Analitza el cost del teu algorisme.

							С	og	n	n	ıs.	N	Oı	m									Ι	).[	N.I		

(Continueu responent aquí a la Pregunta 3.)