Exame A e C
2007/2008

1ª Parte

(1)

1.

```
typedef struct {int p; int s;} Pair;
void minpairs (Pair a[], char v[], int m) {
    int i, j;
    for (i=0; i<m; i++) v[i]=1;            /* c1 c2 c3 c4 */
    for (i=m-1; i>=0; i--)                  /* c5 c6 c7 */
        for (j=i-1; j>=0; j--)              /* c8 c9 c10 */
            if (a[i].s <= a[j].s) v[j]=0;   /* c11 c12 */
```

$$T_{minpairs}(m) = c_1 + c_2 + \sum_{i=0}^{m-1}(c_2 + c_3 + c_4) + c_5 + c_6 + \sum_{i=0}^{m-1}\left(c_6 + c_7 + c_8 + c_9 + \sum_{j=0}^{i-1}(c_9 + c_{10} + c_{11} + c_{12})\right) =$$

$$= c_1 + c_2 + c_5 + c_6 + (c_2 + c_3 + c_4)\sum_{i=0}^{m-1} 1 + \sum_{i=0}^{m-1}\left(c_6 + c_7 + c_8 + c_9 + (c_9 + c_{10} + c_{11} + c_{12})\sum_{j=0}^{i-1} 1\right) =$$

$$= K_1 + K_2 \sum_{i=0}^{m-1} 1 + \sum_{i=0}^{m-1}\left(K_3 + K_4 \sum_{j=0}^{i-1} 1\right) =$$

$$= K_1 + K_2 m + \sum_{i=0}^{m-1}(K_3 + K_4 i) =$$

$$= K_1 + K_2 m + \sum_{i=0}^{m-1} K_3 + \sum_{i=0}^{m-1} K_4 i =$$

$$= K_1 + K_2 m + K_3 m + K_4 \frac{(m-1)m}{2} =$$

$$= K_1 + K_2 m + K_3 m + \frac{K_4}{2} m^2 - \frac{K_4}{2} m =$$

$$= K_1 + \left(K_2 + K_3 - \frac{K_4}{2}\right) m + \frac{K_4}{2} m^2 \in \Theta(m^2)$$

Exame 07/08

Alg C 1ª parte

2. <u>Correcção parcial</u>: significa que não temos de provar o variante, ou seja, que o ciclo termina.

$// \ a = a_0 > 0 \ \wedge \ b = b_0 > 0$

while $(a \, != b)$

$// \ mdc(a_0, b_0) = mdc(a, b)$

if $(a > b)$ $a = a - b$;

else $b = b - a$;

$// \ a = mdc(a_0, b_0)$

1) $P \Rightarrow I$

$a = a_0 > 0 \wedge b = b_0 > 0 \Rightarrow mdc(a_0, b_0) = mdc(a, b)$

2) $I \wedge \neg C \Rightarrow Q$

$mdc(a_0, b_0) = mdc(a, b) \wedge a == b \Rightarrow a = mdc(a_0, b_0)$

3) $\{I \wedge C\} \; S \; \{I\}$

$\{ mdc(a_0, b_0) = mdc(a, b) \wedge a \, != b \}$

if $(a > b)$ $a = a - b$

else $b = b - a$;

$\{ mdc(a_0, b_0) = mdc(a, b) \}$

temos então 2 casos em 3):

a) $\overset{a > b}{mdc}(a_0, b_0) = mdc(a, b) \wedge a \, != b \Rightarrow$
$\Rightarrow mdc(a_0, b_0) = mdc(a - b, b)$

b) $\overset{a <= b}{mdc}(a_0, b_0) = mdc(a, b) \wedge a \, != b \Rightarrow$
$\Rightarrow mdc(a, b - a)$

3.

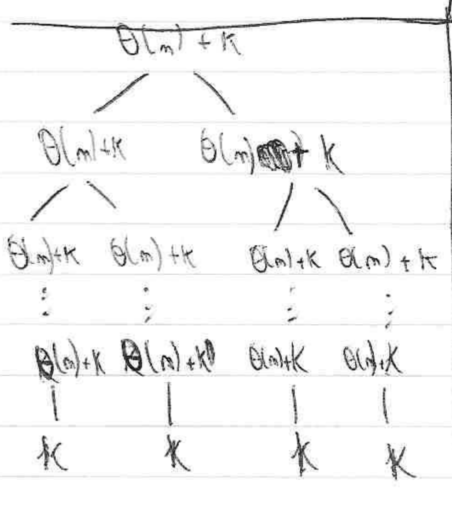
```
void exemplo (int a[], int m) {
    int x = m/2;  c1
    if (m > 0) {  c2
        exemplo (a, x);
        processa (a, m); // Θ(m)
        exemplo (a + x, m - x);
    }
}
```

$x = m/2$

Recorrência:

$$T_{exemplo}(m) = \begin{cases} c_1 + c_2 & \text{se } m \leq 0 \\ \Theta(m) + c_1 + c_2 + T\left(\frac{m}{2}\right) + T\left(\overbrace{m - \frac{m}{2}}^{m/2}\right) & \text{se } m > 0 \end{cases} \iff$$

$$\iff T_{exemplo}(m) = \begin{cases} K & \text{se } m \leq 0 \\ \Theta(m) + K + 2T\left(\frac{m}{2}\right) & \text{se } m > 0 \end{cases}$$



Tamanho	nível	# Rec	Custo
m	0	1	$\Theta(m) + k$
$m/2$	1	2	$\Theta(m) + k$
$m/4$	2	4	$\Theta(m) + k$
$m/2^i$	i	2^i	$\Theta(m) + k$
1	$\log_2 m$	m	$\Theta(m) + k$
0	$\log_2 m + 1$	$2^{\log_2 m + 1} = 2^{\log_2 m} \cdot 2 = 2m$	K

$m/2^i = 1 \iff m = 2^i \iff i = \log_2 m$

$$T_{exemplo}(m) = 2m \cdot K + \sum_{i=0}^{\log_2 m} (K + \Theta(m)) \cdot 2^i =$$

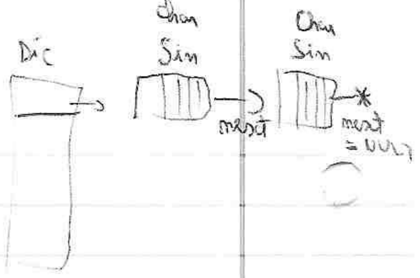
$$= 2m \cdot K + (K + \Theta(m)) \cdot \sum_{i=0}^{\log_2 m} 2^i = 2mK + (K + \Theta(m)) \cdot \frac{2^{\log_2 m + 1} - 1}{2 - 1} =$$

$$= 2nK + (K + \Theta(n)) \cdot (2n - 1) = 2nK + 2nK - K + 2\Theta(n) \cdot n - \Theta(n) =$$

$$= \overbrace{4nK - \underbrace{K}_{\Theta(1)} + \underbrace{2\Theta(n) \cdot n}_{\Theta(n^2)} - \Theta(n)}^{\Theta(n)} =$$

$$= \Theta(n^2) + \Theta(n) + \Theta(1) \in \Theta(n^2)$$

A solução desta recorrência é quadrática.




```
4.  typedef struct s {
        char * sin;
        struct s * next;
    } Sin;

    typedef struct p {
        char * pal;
        Sin * sins;
        struct p * next;
    } Pal;

    # define TAM ...
    typedef Pal * Dic [TAM];

    int hash (char * pal);

    void sinonimos (Dic d, char * pal) {
        int x = hash (pal);
        Pal percorre = d [x];
        while (percorre && strcmp (percorre -> pal, pal))
            percorre = percorre -> next;

        if (percorre) {
            Sin aux = percorre -> sins;
            while (aux) {
                printf ("%s\n", aux -> sin);
                aux = aux -> next;
            }
        }
        else printf ("A palavra n̄ existe no dic.\n");
    }
```

$2n+1 \rightarrow$ esq

$2n+2 \rightarrow$ dir

6.

```
#define TAM ...

typedef int Heap [TAM]

typedef struct nodo {
        int val;
        struct nodo *esq, *dir;
    } Nodo, *Tree;

int heapToTree (Tree *t, Heap h[], int i) {
        if (i < TAM) {
                ~~HeapToTree (t->esq, h, 2i+1)~~
                *t = malloc (sizeof (Nodo))
                t -> val = h [i];
                t -> esq = NULL;
                t -> dir = NULL;
                heapToTree (t->esq, h, 2i+1);
                heapToTree (t->dir, h, 2i+2);
        else t = NULL
```