

**MAC 115 – Introdução à Computação**  
Instituto de Física – Primeiro Semestre de 2004 - Noturno

Prova 3 – 1/07/2004

**Instruções:**

1. A prova pode ser feita a lápis. Cuidado com a legibilidade.
2. Não é permitido o uso de folhas avulsas para rascunho.
3. Não é necessário apagar rascunhos no caderno de soluções.

**Questão 1** (valor: 4 pontos)

Simule a execução do programa abaixo *destacando a sua saída* (o que “vai sair na tela”). Não apague seus rascunhos.

```
/*  
Programa a ser simulado  
*/  
#include <stdio.h>  
  
void crie_mz(int a[][2], int x);  
void imp_mz(int a[][2]);  
void m(int x[][2], int y[][2], int z[][2]);  
  
int main()  
{  
    int a[2][2], b[2][2], c[2][2];  
  
    crie_mz(a, -1); imp_mz(a);  
    crie_mz(b, 0);  imp_mz(b);  
    m(a,b,c);      imp_mz(c);  
  
    crie_mz(a, -1); crie_mz(b, 0); m(a,b,a); imp_mz(a);  
    crie_mz(a, -1); crie_mz(b, 0); m(a,b,b); imp_mz(b);  
  
    return 0;  
}  
  
void m(int x[][2], int y[][2], int z[][2])  
{  
    z[0][0] = x[0][0] * y[0][0] + x[0][1] * y[1][0];  
    z[0][1] = x[0][0] * y[0][1] + x[0][1] * y[1][1];  
    z[1][0] = x[1][0] * y[0][0] + x[1][1] * y[1][0];  
    z[1][1] = x[1][0] * y[0][1] + x[1][1] * y[1][1];  
}  
  
void crie_mz(int a[][2], int x)  
{  
    a[0][0] = x;  a[0][1] = x+1;  a[1][0] = x-1;  a[1][1] = -x;  
}  
  
void imp_mz(int a[][2])  
{  
    printf("%d ", a[0][0]); printf("%d\n", a[0][1]);  
    printf("%d ", a[1][0]); printf("%d\n\n", a[1][1]);  
}
```

### Questão 2 (valor: 3 pontos)

Nesta questão, queremos implementar algumas funções para descobrir o número de linhas e o número de colunas nulas em uma matriz quadrada  $n \times n$ .

(i) Escreva uma função de protótipo

```
int eh_nula(int v[], int n);
```

que, ao ser chamado com `eh_nula(v, n)`, devolve “verdadeiro” se e somente se todas as entradas  $v[0], \dots, v[n-1]$  do vetor  $v$  são nulas.

(ii) Escreva uma função de protótipo

```
int no_linhas0(int a[][NMAX], int n);
```

que, ao ser chamado com `no_linhas0(a, n)`, devolve o número de linhas de  $a$  que são nulas (todas as suas entradas são nulas). Você deve usar a função `eh_nula()` de (i), mesmo que você não a tenha feito.

(iii) Escreva uma função de protótipo

```
int no_colunas0(int a[][NMAX], int n);
```

que, ao ser chamado com `no_colunas0(a, n)`, devolve o número de colunas de  $a$  que são nulas (todas as suas entradas são nulas). Você deve usar a função `eh_nula()` de (i), mesmo que você não a tenha feito.

### Questão 3 (valor: 3 pontos)

Nesta questão, você deve escrever três funções para manipular conjuntos finitos de inteiros. Para representar um conjunto  $A$ , você deve usar um vetor, digamos  $a$ , seguindo a convenção vista em sala: se  $A$  tem  $n$  elementos, então  $a[0] = n$  e  $A = \{a[1], \dots, a[n]\}$ .

(i) Escreva as funções de protótipos

```
void leia_conjunto(int a[]);
```

e

```
void imprima_conjunto(int a[]);
```

para ler e imprimir conjuntos. No caso da leitura, o usuário fornece primeiro a cardinalidade do conjunto e depois os elementos do conjunto.

(ii) Escreva uma função de protótipo

```
int pertence(int x, int a[]);
```

que, ao ser chamado com `pertence(x, a)`, devolve “verdadeiro” se e só se o inteiro  $x$  pertence ao conjunto representado pelo vetor  $a$ .

(iii) Escreva uma função de protótipo

```
void uniao(int u[], int a[], int b[]);
```

que, ao ser chamado com `uniao(u, a, b)`, devolve em  $u$  a união dos conjuntos representados pelo vetores  $a$  e  $b$ . Você deve usar a função `pertence()` de (ii) acima, mesmo que você não a tenha feito. *Observação importante:* conjuntos “não tem elementos repetidos”.