Paradigmas de Programação II
Departamento de Informática
Universidade do Minho

LESI 1º Ano MCC

2002/2003

Ficha Teórico-Prática Nº 3
Programação em C - Manipulação de arrays multi-dimensionais (matrizes)

12 de Março de 2003

1 Objectivos

Esta ficha prática aborda a manipulação de arrays multi-dimensionais em C.
São revisitados os mecanismos de controlo de fluxo (ciclos).

2 Análise de programas C: lint

> *Since human beings themselves are not fully debugged yet, there will be bugs in your code no matter what you do.* - Chris Mason

Para ajudar à criação de programas correctos em C, sugere-se a analise dos programas C, com:

1. `gcc -Wall`
2. O analisador `lint`

Uma implementação do lint está disponível em `http://www.splint.org/`

3 Arrays multi-dimensionais em C

Na linguagem C, os arrays podem ter quantas dimensões forem necessárias. Um array bi-dimensional de inteiros é declarado da seguinte forma:

```
int matriz[10][4];
```

Esta declação reserva $10 * 4 = 40$ células de memória, em que cada armazena um valor do tipo inteiro. As células são identificadas como sendo matriz [0][0] , matriz [0][1] até matriz [9][3] .

Inicialização estática de arrays

Os arrays podem ser inicializados quando são declarados. Por exemplo:

```
int medidas[10] = { 1,2,3,4,5,6,7,8,9,10 };
```

Os arrays podem ser inicializados sem especificar um tamanho concreto; nestes casos, o tamanho necessário é calculado pelo compilador. Exemplo:

```
int ponto[]= { 6, 9, 7 };
```

Os arrays de caracteres podem ser inicializados de uma forma mais espedita. Exemplo:

```
char str[] = "Viva o prof.";
```

que é o mesmo que:

```
char str[] = { 'V','i','v','a',' ','o',' ','p','r','o','f','.' };
```

Arrays multi-dimensionais podem ser inicializados da mesma forma. Exemplo:

```
int transposta[6][2] = { 1, 1, 2, 4, 1, 4, 7, 8, 1, 2, 3, 6 };
```

3.1 Exercícios

Exercício 1 (Multiplicação de matrizes) *Desenvolva um programa C que permite multiplicar duas matrizes.*

O tamanho das matrizes pode variar entre 2 e 10. Leia e valide, em primeiro lugar, as dimensões da primeira ($i \times j$) e da segunda ($j \times k$) matriz. Depois leia cada uma das matrizes.

No final, mostre a matriz ($i \times k$) resultado.

Exercício 2 (Batalha Naval) *Desenvolva as funções de suporte ao Jogo da Batalha Naval, nomeadamente:*

1. *Escreva a função que implementa a opção "Tiro certeiro", que pede uma linha (letra) e uma coluna (número), e devolve a informação sobre o resultado do tiro: água ou o tipo de barco alvejado.*
2. *Escreva a função que mostra o "Estado da frota": quantos navios foram atingidos, e quais os afundados. (Sugere alterações à estrutura de dados? Uma estrutura de dados auxiliar facilitará esta função?)*
3. *Escreva a função "Tiro aleatório", em que é o computador a calcular o próximo tiro, em vez de ser o utilizador (usando a função int rand(void);[1]).*
4. *Escreva a função booleana que verifica se a disposição da frota cumpre as seguintes restrições:*
   (a) *Os navios não se podem tocar, nem na diagonal.*
   (b) *Existe um porta-aviões com 5 canos dispostos em T.*
   (c) *Existe um destroyer com 4 canos dispostos em I.*
   (d) *Existem duas fragatas com 3 canos dispostos em I.*
   (e) *Existem três cruzadores com 2 canos dispostos em I.*
   (f) *Existem quatro submarinos com 1 cano.*

[1] Consulte o manual da função rand, e estude, como exemplo, o seguinte programa C, que calcula um número aleatório entre 1 e 10.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

// únmeros pseudo-óaleatrios entre 1 e 10
int main() {
        srand(time(NULL));
        int k=1+(int)(10.0*rand()/(RAND_MAX+1.0));
        printf("%d\n", k );
        return 0;
}
```

A função void srand(unsigned int seed); só precisa ser invocada uma única vez.

Utilize, como ponto de partida, o seguinte programa C.

```
#include <stdio.h>
#include <ctype.h>
#include <string.h>
#define TAM 10
//  ' ' vazio
//  'A'                     1x porta-ões avies
//  'B'                     1x destroyer (4 canos)
//  'C', 'D'                2x fragata   (3 canos)
//  'E', 'F', 'G'           3x cruzador  (2 canos)
//  'H', 'I', 'J', 'K'      4x submarino (1 cano)
char tabuleiro[TAM][TAM]=
{
        ' ', ' ', 'E', ' ', ' ', ' ', ' ', ' ', 'B', ' ',
        ' ', ' ', 'E', ' ', 'C', 'C', 'C', ' ', 'B', ' ',
        ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', 'B', ' ',
        'F', 'F', ' ', ' ', ' ', ' ', 'H', ' ', 'B', ' ',
        ' ', ' ', ' ', 'A', ' ', ' ', ' ', ' ', ' ', ' ',
        'G', 'G', ' ', 'A', 'A', 'A', ' ', ' ', ' ', 'D',
        ' ', ' ', ' ', 'A', ' ', ' ', ' ', ' ', ' ', 'D',
        ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', 'D',
        ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ',
        ' ', 'I', ' ', ' ', ' ', 'J', ' ', ' ', ' ', 'K'
};

int mostra(char tab[TAM][TAM]) {
        int linha, coluna;
        for(linha=0; linha < TAM; linha++) {
                for(coluna=0; coluna < TAM; coluna++)
                        printf("%c", tab[linha][coluna]);
                printf("\n");
        }
        return 0;
}

int main() {
        char opcao;
        do {
                printf("(T)iro certeiro\n");
                printf("(A)iro óaleatrio\n");
                printf("(E)stado da frota\n");
                printf("(M)ostrar\n");
                printf("(V)alidar çãdistribuio da esquadra\n");
                printf("\n(S)air\n");
                scanf("%c", &opcao);
                switch(toupper(opcao)) {
                        case 'M': mostra(tabuleiro);
                                  break;
                        default: if (toupper(opcao)!='S')
                                        printf("çãOpo áinvlida\n");
                }
        } while (toupper(opcao)!='S');
        return 0;
```

}

Exercício 3 (Jogo das minas) *Desenvolva um jogo baseado no metáfora de percorrer um campo minado sem pisar uma mina. O campo minado é representado por uma matriz de $n \times n$ (o jogador pode escolher o valor de $5 \leq n \leq 50$).*

Cada elemento dessa matriz pode conter ou não uma mina. O número de minas nas células é calculado de acordo com um critério de dificuldade (que pode variar entre 1 e 5, em que 1 indica que apenas 10% dos elementos são minas, enquanto que ao valor 5 corresponde um campo com 50% de minas).

A distribuição das minas deve ser o mais aleatória possível, usando para isso, a função int rand(void).

Ao utilizador é pedido que vá indicando pares (linha e coluna) de localizações; o jogo termina quando todas as células sem minas forem visitadas (o jogador vence), ou quando o utilizador escolher uma célula com mina (o jogador é derrotado).

Opcionalmente, permita que, a cada momento o jogador pode ver o campo minado, com a indicação das células já pisadas; nestas aparece o número de minas adjacentes.

Exercício 4 (4 cavalos e 4 rainhas) *O objectivo do jogo é conseguir colocar 4 cavalos e quatro rainhas num tabuleiro de xadrez.*

Faça um programa que lê até oito jogadas (peça, linha, coluna), e indique quantas peças estão em condições de serem efectivamente colocadas, sem serem comidas e sem ficarem em posição de comer peças já colocadas.

Exercício 5 (8 rainhas) *O objectivo do jogo é conseguir colocar 8 rainhas num tabuleiro de xadrez, sem que se ameacem umas às outras.*

Faça um programa que verifique se é possível colocar as restantes 7 rainhas, depois de colocar uma numa posição escolhida pelo utilizador.