Vetores e matrizes

Programação I
2020.2021

Teresa Gonçalves
tcg@uevora.pt

Departamento de Informática, ECT-UÉ

Reminder...

Como aprender?

Estudar, estudar, estudar...

Praticar, praticar, praticar...

Cometer erros, cometer erros, cometer erros...

Aprender com os erros, aprender com os erros, aprender com os erros ...

Sumário

Vetores

Matrizes

Instrução for

Outras construções da linguagem C

Vetores

Vetor

Conceito familiar em matemática

Conjunto de valores indexados por um índice

Valores

São todos do mesmo tipo (básico)

Índice

inteiro

Declaração e inicialização

```
tipo variavel[dimensao];
```

```
tipo variavel[dimensao] = {valor1, valor2, …, valorn};
```

Exemplos

```
int notas[10];
```

Vetor de inteiros de dimensão 10

```
float precos[5] = {1.5, 2.3, 4.0, 3.1, 5.5};
```

Vetor de reais de dimensão 5

Utilização

Acesso direto aos elementos através do índice

```
variavel[indice]
```

Indexação

Primeiro elemento: índice 0 (zero)

Último elemento: índice ?

Exemplos

```
nota = notas[5];
preco = precos[1];
```

Exemplo

```
float media( float v[], int n ){
    int i;
    float soma;
    i=0;
    soma=0;
    while(i<n){
      soma=soma+v[i];
      i=i+1;
    }
    return soma/n;
}
```

```
int main(){
  float notas[10], nota;
  int nnotas;
  nnotas=0;
  printf("Insira as notas terminando
com -1 (maximo 10)\n");
  scanf( "%d", &nota );
  while(nota!=-1 && nnotas<10){
    notas[nnotas]=nota;
    nnotas=nnotas+1;
    scanf( "%d", &nota );
  }
  printf( "A media e %f.\n",
media(notas, nnotas) );
}
```

Matrizes

Matriz

Generalização de vetor

Vetor com mais de uma dimensão

Pouco usual utilizar mais de 3 dimensões

Declaração

```
tipo variavel[dimensao1][dimensao2];
```

Utilização

```
variavel[indice1][indice2]
```

Exemplo

```
int somaMatriz( int mat[][10], int d1, int d2 ){
  /* d1, d2 devem ser menor ou igual a 10 */
  int i, j, soma;

  i=0;
  j=0;
  soma=0;
  while(i<d1){
    while(j<d2){
      soma=soma+mat[i][j];
      j=j+1;
    }
    i=i+1;
    j=0;
  }
  return soma;
}
```

Alternativas na indicação de parâmetros

Vetores

```
float media( float v[], int n );
float media( float v[10], int n );
```

Matrizes

```
int somaMatriz( int mat[][10], int d1, int d2 );
int somaMatriz( int mat[10][10], int d1, int d2 );
```

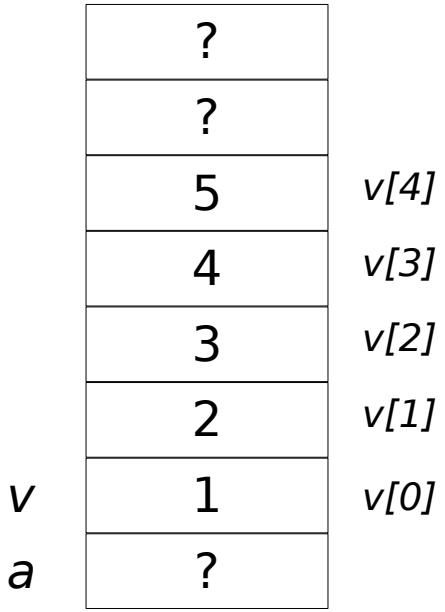
Nota

É necessário indicar todas as dimensões exceto a primeira!

Matrizes e memória

Vetor

```
int a;
int v[5] = {1,2,3,4,5};
```



Matriz

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

Ciclo for

Instrução for

```
for(<inicio>; <condicao>; <val_seguinte>)
    <instrução>

for(<inicio>; <condicao>; <val_seguinte>){
    <bloco de instruções>
}
```

<inicio>

Instrução executada primeiro e apenas 1 vez

Serve para **inicializar a variável de controlo** do ciclo

<condicao>

Determina se o bloco de instruções é executado

<val_seguinte>

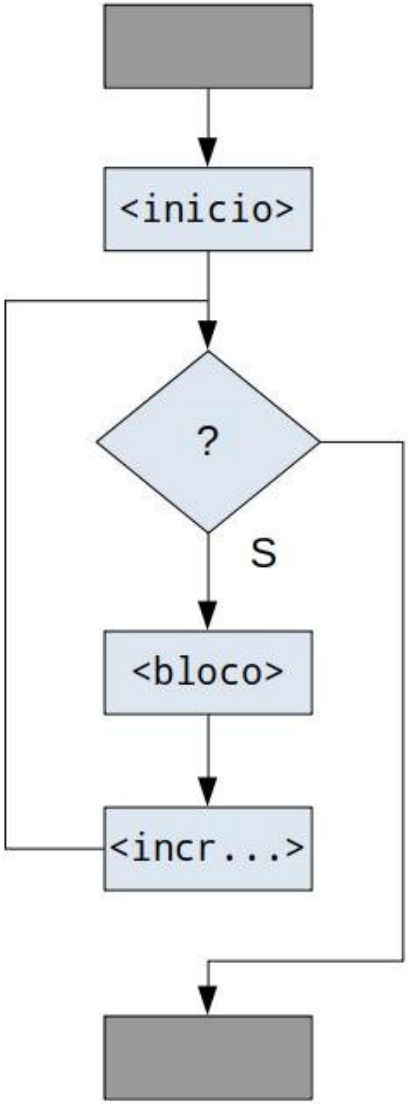
Depois de executar o bloco de instruções, a variável de controlo é atualizada de acordo com o <val_seguinte>

Fluxo de execução

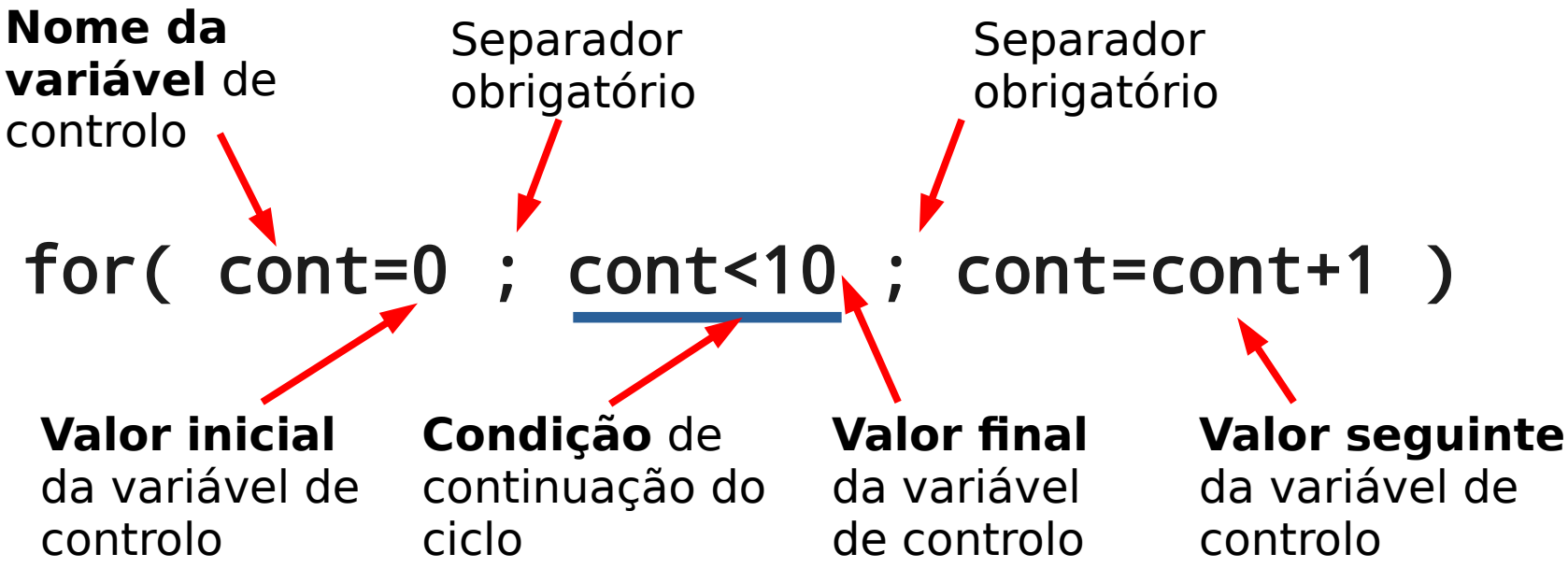
```
for(<inicio>; <condicao>; <val_seguinte>){
    <bloco de instruções>
}
```

1. Executa a instrução `<inicio>`
2. Avalia a `<condição>`, obtendo verdade ou falso
3. Se falso, sai da instrução `for` e continua com a próxima instrução
4. Se verdade, executa o `<bloco de instruções>` e atualiza a variável de controlo com `<val_seguinte>`
5. Volta ao ponto 2

Fluxograma



Utilização típica



While vs. For

While

```
int i;
float r;

r=1;
i=0;
while(i<10){
   r=r*1.5;
   i=i+1;
}
```

For

```
int i;
float r;

r=1;
for(i=0; i<10; i=i+1)
   r=r*1.5;
```

Exemplo

Calcular a soma de todos os números pares até ao número n

```
int somaPares( int n ){
  int soma, num;
  soma=0;
  for( num=2; num<=n; num=num+2 )
    soma=soma+num;
  return soma;
}
```

Outras construções da linguagem C

Atribuição

var = var op expr

```
i = i+1
j = j-10
k = k*(n+1)
```

var op= expr

```
i += 1
j -= 10
k *= n+1
```

Incremento e decremento

```
a++;
++aa;
```

```
a += 1;
a += 1;
```

```
k = 2 * ++i; (prefixo)
k = 2 * i++; (sufixo)
```

```
i += 1; k = 2*i;
k = 2*i; i += 1;
```

Mais operadores

Relacionais

>	maior
>=	maior ou igual
<	menor
<=	menor ou igual
==	igual
!=	diferente

Bit a Bit

Operadores aplicados a cada um dos bits que representam o inteiro

~	negação
&	e
|	ou
^	ou exclusivo
<<	deslocamento para a esq

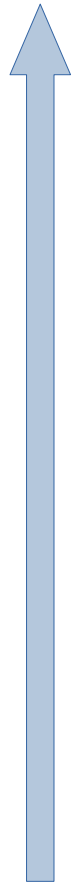
(corresponde a multiplicar o nº por 2)

>>	deslocamento para a dir

(corresponde a dividir o nº por 2)

Precedência a associatividade

maior



menor

Precedência	Associatividade
() []	esquerda
! ~ ++ --	direita
* / %	esquerda
+ -	esquerda
<< >>	esquerda
< <= > >=	esquerda
== !=	esquerda
&	esquerda
&&	esquerda
||	esquerda
^	esquerda
|	esquerda
= += -= *= /=	direita

Iteração

```
while( condição )
    instrução
```

```
do
    instrução
while( condição );
```

Estrutura de um programa C

```
diretivas de pré-processamento

declaração de funções

int main()
{
    declarações
    instruções
}
```

Exemplo

```
#include <stdio.h>

float calcula_area_circulo(int raio);

int main()
{
    int raio = 1;
    int n = 3;
    printf("A. Circulo: %.3f\n", calcula_area_circulo(raio));
    return(0);
}

float calcula_area_circulo(int raio)
{
    float area;
    area = 3.1415 * raio * raio;
    return area;
}
```

Protótipo

Corresponde ao cabeçalho da função

Permite que as funções sejam declaradas em qualquer local de um ficheiro

- Desde que sejam colocados os protótipos no início

printf e scanf

```
printf( “string controlo”, arg1, arg2, …, argn )
scanf( “string controlo”, &arg1, &arg2, … &argn )
```

Expressão controlo

%d inteiro (em decimal)
%o inteiro (em octal)
%x inteiro (em hexadecimal)
%c caracter
%s string (cadeia de caracteres)
%f loat
%e (notação exponencial)
%% caracter %

Verificação da leitura

A função scanf devolve o nº de valores corretamente lidos

Exemplo

```
int resultado;
float x, y;
resultado = scanf("%f %f", &x, &y);
if(resultado != 2)
    printf("Valores invalidos“);
```