ALGORITMOS E ESTRUTURA DE DADOS II

Prof. Fábio Ferreira de Assis

REVISÃO
AULA 04

Array (Vetor e Matriz)

Prof. Msc. Fábio Ferreira de Assis
E-mail: fabio.assis@online.uscs.edu.br

Disciplina: ***ALGORITMOS E ESTRUTURA DE DADOS I***

Curiosidade

Condessa de Lovelace, conhecida como Ada Lovelace - a matemática que criou o primeiro algoritmo para ser processado por uma máquina, sendo a primeira programadora da história.

A relevância da criação de Lovelace é tamanha que há diversos anos em todo dia 15 de outubro é comemorado em todo o mundo o Ada Lovelace Day - data criada por Suw Charman-Anderson para celebrar e espalhar mundo afora as conquistas de mulheres na ciência, inspirando outras a seguirem carreira nessa área.

Fonte: https://canaltech.com.br/curiosidades/mulheres-historicas-ada-lovelace-a-primeira-programadora-de-todos-os-tempos-71395/
Acessado em 15/10/2021 – 18:00h

VETOR OU ARRAY

Vetor ou *Array*: são estruturas de dados simples que auxiliam quando trabalhamos com muitas variáveis do mesmo tipo.

Vetor ou também chamado de *Array* unidimensional: é uma variável que pode armazenar variáveis do mesmo tipo.

O **vetor** é uma estrutura de dados indexada, que armazena uma finita quantidade de valores do mesmo tipo.

VETOR OU ARRAY

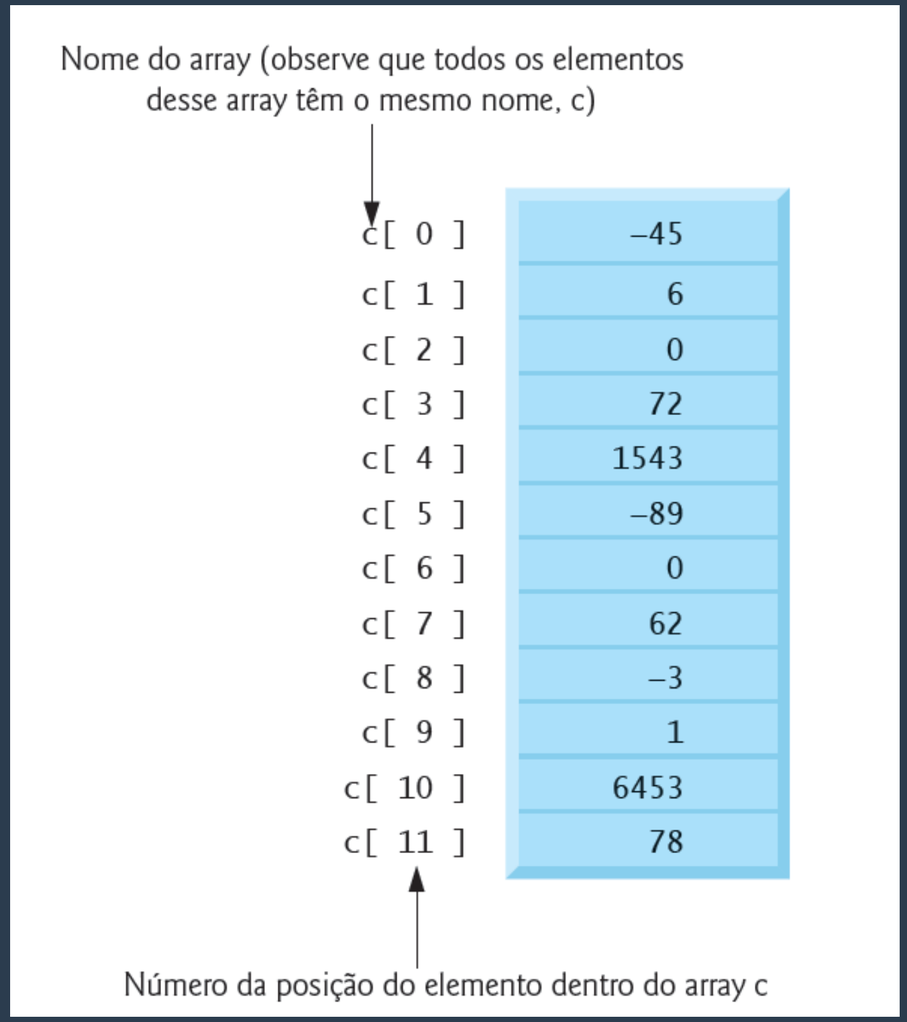
Arrays

Um array é um conjunto de espaços de memória que se relacionam pelo fato de que todos têm o mesmo nome e o mesmo tipo.

Para se referir a um local ou elemento em particular no array, especificamos o nome do array e o **número da posição** do elemento em particular no array.

A Figura 1 mostra um array de inteiros chamado c. Esse array contém 12 **elementos**. Qualquer um desses elementos pode ser referenciado com o nome do array seguido pelo número de posição do elemento em particular entre colchetes ([ ]). O primeiro elemento em cada array é identificado pela posição **elemento zerésimo**. Assim, o primeiro elemento do array c é referenciado como c[0], o segundo elemento, como c[1], o sétimo elemento, como c[6], e, em geral, o *i*-ésimo elemento do array c é referenciado como c[i — 1]. Os nomes do array, como outros nomes de variável, só podem conter letras, dígitos e sublinhados. Nomes de array não podem começar por um dígito.

VETOR, MATRIZ OU ARRAY



Exemplo Vetor

Declaração do vetor

```
int Vetor[5];   // declara um vetor de 5 posições
```

Acesso aos elementos do vetor

Para acessar os elementos de um vetor usa-se índices. O índice define a posição da variável dentro do vetor.

Em todos os vetores tem o primeiro elemento na posição 0(zero). Assim, se tomarmos "K" como sendo o tamanho do vetor a última posição é a de índice "K-1"

```
Vetor[0] = 4; // Coloca 4 na primeira posição de "Vetor"
Vetor[4] = 8; // Coloca 8 na última posição de "Vetor"
```

Vetor

EXEMPLO DE VETOR

```
1  programa {
2      funcao inicio() {
3          inteiro numeros[] = {39, 45, 55}
4
5          escreva(numeros[0])
6      }
7  }
8
```

Vetor em Portugol e em C

EXEMPLO DE VETOR

Programa que imprime o valor da posição zero do vetor

```
programa {
    funcao inicio() {
        inteiro numeros[] = {39, 45, 55}

        escreva(numeros[0])
    }
}
```

</> source code

```c
#include <stdio.h>

int main()
{
    int numeros[] = {39, 45, 55};

    printf("%d", numeros[2]);

    return 0;
}
```

Vetor

```
programa
{
	funcao inicio()
	{
		inteiro i, numeros[]={39, 45, 4, 5}

		para(i = 0; i <= 3; i++){
			escreva(numeros[i], " ")
		}
	}
}
```

```
programa
{
	funcao inicio()
	{
	inteiro numeros[] = {39, 45, 4, 5}
		escreva(numeros[0])
		escreva("\n", numeros[1])
		escreva("\n", numeros[2])
		escreva("\n", numeros[3])
	}
}
```

```
programa
{
	funcao inicio()
	{
	inteiro numeros[] = {39, 45, 4, 5}
		escreva(numeros[0], "\n", numeros[1],
		"\n", numeros[2], "\n", numeros[3])
	}
}
```

EXEMPLO

Dado um vetor com 3 posições pré-definidas, imprime na tela o valor armazenado na posição 0 do vetor.

```
programa
{
	funcao inicio()
	{
		const inteiro tamanhoVetor = 3

		inteiro numeros[tamanhoVetor] = {39, 45, 55}
		escreva(numeros[0])

		numeros[0] = 9999

		escreva("\n\n\n")

		escreva(numeros[0])
	}
}
```

EXEMPLO

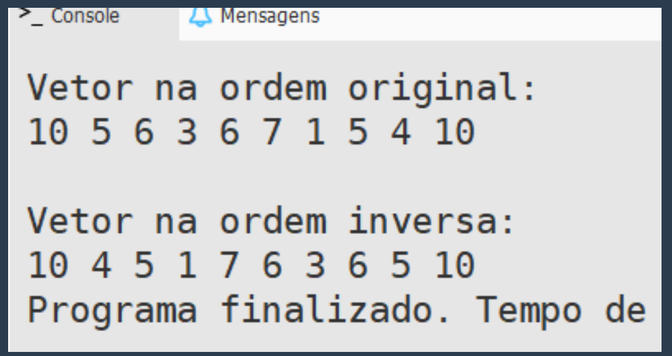
Dado um vetor com 3 posições pré-definidas, imprime na tela o valor armazenado na posição 0 do vetor.

```
programa
{
	funcao inicio()
	{
		const inteiro tamanhoVetor = 3

		inteiro numeros[tamanhoVetor] = {39, 45, 55}

		escreva("Vetor sem alteração")
		escreva("\n", numeros[0])
		escreva("\n", numeros[1])
		escreva("\n", numeros[2])

		numeros[0]= 9999

		escreva("\n\nVetor com alteração")
		escreva("\n", numeros[0])
		escreva("\n", numeros[1])
		escreva("\n", numeros[2])
	}
}
```

```
>_ Console   Mensagens
Vetor sem alteração
39
45
55

Vetor com alteração
9999
45
55
```

EXEMPLO

```
programa {
	inclua biblioteca Util --> util

	funcao inicio() {
		inteiro vetor[10]

		// preenche o vetor
		para (inteiro posicao = 0; posicao < 10; posicao++) {
			// Sorteia um número e atribui à posição do vetor
			vetor[posicao] = util.sorteia(1, 100)
		}

		// Exibe o vetor na ordem original
		escreva ("Vetor na ordem original:\n")

		para(inteiro posicao = 0; posicao < 10; posicao++) {
			escreva (vetor[posicao], " ")
		}

		// Exibe o vetor na ordem inversa
		escreva ("\n\nVetor na ordem inversa:\n")

		para(inteiro posicao = 9; posicao >=0; posicao--) {
			escreva (vetor[posicao], " ")
		}
	}
}
```



EXEMPLO

```
programa {
    inclua biblioteca Util --> util

    funcao inicio() {
        inteiro vetor[100]

        // preenche o vetor
        para (inteiro posicao = 0; posicao < 100; posicao++) {
            // Sorteia um número e atribui à posição do vetor
            vetor[posicao] = util.sorteia(1, 10)
        }

        // Exibe o vetor na ordem original
        escreva ("Vetor na ordem original:\n")

        para(inteiro posicao = 0; posicao < 100; posicao++) {
            escreva (vetor[posicao], " ")
        }

        // Exibe o vetor na ordem inversa
        escreva ("\n\nVetor na ordem inversa:\n")

        para(inteiro posicao = 99; posicao >= 0; posicao--) {
            escreva (vetor[posicao], " ")
        }
    }
}
```

```
programa {
    inclua biblioteca Util --> util

    funcao inicio() {
        const inteiro tamanho = 100
        inteiro vetor[tamanho]

        // preenche o vetor
        para (inteiro posicao = 0; posicao < tamanho; posicao++) {
            // Sorteia um número e atribui à posição do vetor
            vetor[posicao] = util.sorteia(1, 10)
        }

        // Exibe o vetor na ordem original
        escreva ("Vetor na ordem original:\n")

        para(inteiro posicao = 0; posicao < tamanho; posicao++) {
            escreva (vetor[posicao], " ")
        }

        // Exibe o vetor na ordem inversa
        escreva ("\n\nVetor na ordem inversa:\n")

        para(inteiro posicao = tamanho-1; posicao >= 0; posicao--) {
            escreva (vetor[posicao], " ")
        }
    }
}
```

EXEMPLO

Programa que procura um valor digitado pelo usuário, dentro do vetor de cinco posições com valores pré-definidos. Se encontrar imprime “Valor encontrado na posição i”. Caso contrário imprime “Valor não encontrado”.

EXEMPLO – RESOLUÇÃO A – SEM LAÇO DE REPETIÇÃO

```
programa
{
    funcao inicio() {
        const inteiro tamanhoVetor = 5
        inteiro valor, numeros[tamanhoVetor] = {10, 11, 39, 45, 55}

        escreva("Digite um valor para ver se esta no vetor ")
        leia(valor)
        se(valor == numeros[0]) {
            escreva("Valor encontrado na posição: 0")
        }
        senao se(valor == numeros[1]) {
            escreva("Valor encontrado na posição: 1")
        }
        senao se(valor == numeros[2]) {
            escreva("Valor encontrado na posição: 2")
        }
        senao se(valor == numeros[3]) {
            escreva("Valor encontrado na posição: 3")
        }
        senao se(valor == numeros[4]) {
            escreva("Valor encontrado na posição: 4")
        }
        senao {
            escreva("Valor NÃO encontrado")
        }
    }
}
```

```
se(numeros[0] == valorDigitado)
        escreva("Valor encontrado na posição ", 1)

senao se(numeros[1] == valorDigitado)
        escreva("Valor encontrado na posição ", 2)

senao se(numeros[2] == valorDigitado)
        escreva("Valor encontrado na posição ", 3)

senao se(numeros[3] == valorDigitado)
        escreva("Valor encontrado na posição ", 4)

senao se(numeros[4] == valorDigitado)
        escreva("Valor encontrado na posição ", 5)

senao
        escreva("Valor não encontrado")
```

EXEMPLO – RESOLUÇÃO A – SEM LAÇO DE REPETIÇÃO

```
programa
{
    funcao inicio() {
        const inteiro tamanhoVetor = 5
        inteiro valor, numeros[tamanhoVetor] = {10, 11, 39, 45, 55}

        escreva("Digite um valor para ver se esta no vetor ")
        leia(valor)
        se(valor == numeros[0]) {
            escreva("Valor encontrado na posição: 0")
        }
        senao se(valor == numeros[1]) {
            escreva("Valor encontrado na posição: 1")
        }
        senao se(valor == numeros[2]) {
            escreva("Valor encontrado na posição: 2")
        }
        senao se(valor == numeros[3]) {
            escreva("Valor encontrado na posição: 3")
        }
        senao se(valor == numeros[4]) {
            escreva("Valor encontrado na posição: 4")
        }
        senao {
            escreva("Valor NÃO encontrado")
        }
    }
}
```

```
#include <stdio.h>
#include <locale.h>

int main(){
    setlocale(LC_ALL, "Portuguese");
    int valorDigitado=0;

    printf("Digite um valor: ");
    scanf("%d", &valorDigitado);

    int numeros[] = {10,11, 39, 45, 55};

    if(valorDigitado == numeros[0])
        printf("Valor encontrado na posicao 1");

    else if(valorDigitado == numeros[1])
        printf("Valor encontrado na posicao 2");

    else if(valorDigitado == numeros[2])
        printf("Valor encontrado na posicao 3");

    else if(valorDigitado == numeros[3])
        printf("Valor encontrado na posicao 4");

    else if(valorDigitado == numeros[4])
        printf("Valor encontrado na posicao 5");

    else
        printf("Valor nao encontrado!");

    return 0;
}
```

EXEMPLO – RESOLUÇÃO A – COM LAÇO DE REPETIÇÃO

```
1  programa {
2      funcao inicio() {
3          inteiro temp=0, valor=0, i=0
4          inteiro numeros[] = {39, 45, 39, 5, 100, 20, 21, 36, 5, 59}
5          escreva("Digite um número da sorte! ")
6          leia(valor)
7
8          para(i=0; i<10; i++){
9              se(valor == numeros[i]){
10                 escreva("Valor encontrado na posição ", i+1, "\n")
11                 temp++
12             }
13         }
14         se(temp==0)
15             escreva("\nValor não encontrado!")
16
17         escreva("\n\ni = ", i)
18     }
19 }
```

```
>_ Console   Mensagens

Digite um número da sorte! 39
Valor encontrado na posição 1
Valor encontrado na posição 3


i = 10
Programa finalizado. Tempo de
```

EXEMPLO – RESOLUÇÃO A – COM LAÇO DE REPETIÇÃO

```
programa {
    funcao inicio() {
        inteiro temp=0, valor=0, i=0
        inteiro numeros[] = {39, 45, 39, 5, 100, 20, 21, 36, 5, 59}
        escreva("Digite um número da sorte! ")
        leia(valor)

        para(i=0; i<10; i++){
            se(valor == numeros[i]){
                escreva("Parabéns você ganhou um prêmio!")
                escreva("Valor encontrado na posição ", i+1, "\n")
                temp++
                pare
            }
        }
        se(temp==0)
            escreva("\nValor não encontrado!")

        escreva("\ni = ", i)
    }
}
```

EXEMPLO

EXEMPLO DE VETOR

Preencher vetor com números aleatórios

```
programa{
  inclua biblioteca Util
  funcao inicio(){
    inteiro i, vet[20]

    para(i = 0; i < 20; i++)
      vet[i] = Util.sorteia(0, 20)

    para(i = 0; i < 20; i++)
      escreva(vet[i], ", ")
    escreva("\n")
  }
}
```

```
1  programa {
2      inclua biblioteca Util --> u
3
4      funcao inicio() {
5          const inteiro tamanho = 20
6          inteiro i=0, vet[tamanho]
7
8          para(i=0; i<tamanho; i++){
9              vet[i] = u.sorteia(0, 20)
10         }
11
12         para(i=0; i<tamanho; i++){
13             escreva(vet[i], ",\n")
14         }
15     }
16 }
```

EXEMPLO

EXEMPLO DE VETOR

Preencher vetor com números aleatórios

```
programa {
	inclua biblioteca Util --> u

	funcao inicio() {
		const inteiro tamanho = 20
		inteiro i=0, vet[tamanho]

		para(i=0; i<tamanho; i++){
			vet[i] = u.sorteia(0, 20)
		}

		para(i=0; i<tamanho; i++){
			se(i != tamanho-1)
				escreva(vet[i], ", ")
			senao
			escreva(vet[i], ".")
		}
	}
}
```

EXEMPLO

EXEMPLO DE VETOR

Preencher vetor com

números aleatórios

```
programa
{     inclua biblioteca Util --> ut
      funcao inicio()
      {
            inteiro temp =0, valor, i, vet[100]
            escreva("Digite o número Mágico: ")
            leia(valor)

            para(i=0; i<=99; i++){
                  vet[i] = ut.sorteia(0, 100)
                  escreva(vet[i], " ")
            }

            para(i = 0; i <= 99; i++){
                  se(valor == vet[i]){
                        escreva("\nValor encontrado na posição ", i)
                        temp++
                  }
            }
            se(temp > 0)
                  escreva("\nPARABÉNS VOCÊ GANHOU UM MILHÃO")
            senao
                  escreva("\nVOCÊ PERDEU")
      }
}
```

EXEMPLO

Localiza um valor dentro do vetor

```
programa {
	inclua biblioteca Util --> u
	funcao inicio() {
		const inteiro tamanho = 100000
		inteiro temp=0, valor=0, i=0, vetor[tamanho]
		escreva("Digite um número da sorte! ")
		leia(valor)

		para (i = 0; i < tamanho; i++) {
			// Sorteia um número e atribui à posição do vetor
			vetor[i] = u.sorteia(1, 100)
		}

		para(i = 0; i < tamanho; i++){
			se(valor == vetor[i]){
				escreva("Parabéns você ganhou um prêmio!")
				escreva("Valor encontrado na posição ", i+1, "\n")
				temp++
				pare
			}
		}
		se(temp==0)
			escreva("\nValor não encontrado!")

		escreva("\ni = ", i)
	}
}
```

EXEMPLO

Programa que recebe diversos valores digitados pelo usuário até preencher um vetor com 5 elementos. Solicite ao usuário para digitar valores entre 0 e 100.

Depois imprime o maior valor dentro do vetor.

EXEMPLO – RESOLUÇÃO

```
programa
{
    funcao inicio()
    {
        inteiro vet[5], i=0, maior=0

        para(i=0; i<=4; i++){
            escreva("Digite um valor entre 0 e 100: ")
            leia(vet[i])

            se(i==0)
                maior = vet[i]

            se(maior < vet[i])
                maior = vet[i]
        }

        escreva("Maior valor = ", maior)
    }
}
```

</> source code

```c
#include <stdio.h>

int main(void) {
    int tamanho = 5;
    int numeros[tamanho];
    int i, maior = 0;

    for(i=0; i<tamanho; i++)
    {
        printf("Digite o número %d\n", i+1);
        scanf("%d", &numeros[i]);

        if(numeros[i]>maior)
            maior = numeros[i];
    }
    printf("O maior é %d", maior);
    return 0;
}
```

EXEMPLO – RESOLUÇÃO 2

```
programa
{
	funcao inicio()
	{
		const inteiro tamanho = 10
		inteiro numeros[tamanho]
		inteiro i, maior=0

		para(i=0; i<tamanho; i++)
		{
			escreva("Digite um valor entre 0 e 100: ")
			leia(numeros[i])
			se(numeros[i] > maior)
				maior = numeros[i]
		}
		escreva("\nO maior valor é = ", maior)
	}
}
```

EXEMPLO – RESOLUÇÃO 3 – MOSTRANDO O MAIOR VALOR E POSIÇÃO NO VETOR

```
programa
{
	funcao inicio()
	{
		inteiro vet[5], posicao=0, i=0, maior=0

		para(i=0; i<=4; i++){
			escreva("Digite um valor entre 0 e 100: ")
			leia(vet[i])

			se(i==0)
				maior = vet[i]

			se(maior < vet[i]){
				maior = vet[i]
				posicao = i
			}
		}

		escreva("Maior valor = ", maior)
		escreva("\nPosicao = ", posicao)
	}
}
```

EXEMPLO em C

</> source code

```c
#include <stdio.h>

int main(void) {
    int tamanho = 5;
    int numeros[tamanho];
    int i, maior = 0;

    for(i=0; i<tamanho; i++)
    {
        printf("Digite o número %d\n", i+1);
        scanf("%d", &numeros[i]);

        if(numeros[i]>maior)
            maior = numeros[i];
    }
    printf("O maior é %d", maior);
    return 0;
}
```

EXEMPLO

```
1    Inicializando um array com uma lista de inicializadores */
2  #include <stdio.h>
3
4  /* função main inicia a execução do programa */
5  int main( void )
6  {
7     /* usa lista de inicializadores para inicializar array n */
8     int n[ 10 ] = { 32, 27, 64, 18, 95, 14, 90, 70, 60, 37 };
9     int i; /* contador */
10
11    printf( "%s%13s\n", "Elemento", "Valor" );
12
13    /* lista conteúdo do array em formato tabular */
14    for ( i = 0; i < 10; i++ ) {
15       printf( "%7d%13d\n", i, n[ i ] );
16    } /* fim do for */
17
18    return 0; /* indica conclusão bem-sucedida */
19 } /* fim do main */
20
```

Elemento	Valor
0	32
1	27
2	64
3	18
4	95
5	14
6	90
7	70
8	60
9	37

Exemplo Vetor/Matriz

Declaração de Matrizes

```
int Matriz[5][3]; // declara uma matriz de 5 linhas e 3 colunas
```

INTEIRO num[2][5];

num	1	2	3	4	5
1	num[1][1]				
2			num[2][3]		

num[2][5] = {1, 2, 3, 4, 5,
6, 7, 8, 9, 10}

num[ ][ ] = {1, 2, 3, 4, 5,
6, 7, 8, 9, 10}

num	0	1	2	3	4
0	1	2	3	4	5
1	6	7	8	9	10

Exemplo Vetor/Matriz

Exemplos com Vetores

```
int Vetor[5];  // declara um vetor de 5 posições

int Matriz[5][3]; // declara uma matriz de 5 linhas e 3 colunas

    Vetor[0] = 9; // coloca 9 na primeira posição do vetor
    Vetor[4] = 30 // coloca 30 na última posição do vetor

    Matriz[0][1] = 15; // coloca 15 na célula que está na primeira linha
                       // e na segunda coluna da matriz
```

Preenchimento de um vetor com um dado

```
for(i=0; i<5; i++)
   Vetor[i] = 30;
```

```
for(i=0; i<=4; i++)
   Vetor[i] = 30;
```

Colocar os números de 1 a 5 em Vetor

```
for(i=0; i<5; i++)
  Vetor[i] = i+1;
```

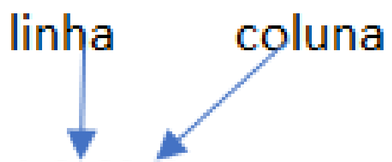
Colocar os números de 5 a 1 em Vetor

```
for(i=0; i<5; i++)
  Vetor[i] = 5-i;
```

EXEMPLO MATRIZ

matriz[2][2]

	coluna0	coluna1
linha0	matriz[0][0]	matriz[0][1]
linha1	matriz[1][0]	matriz[1][1]



matriz[2][4]

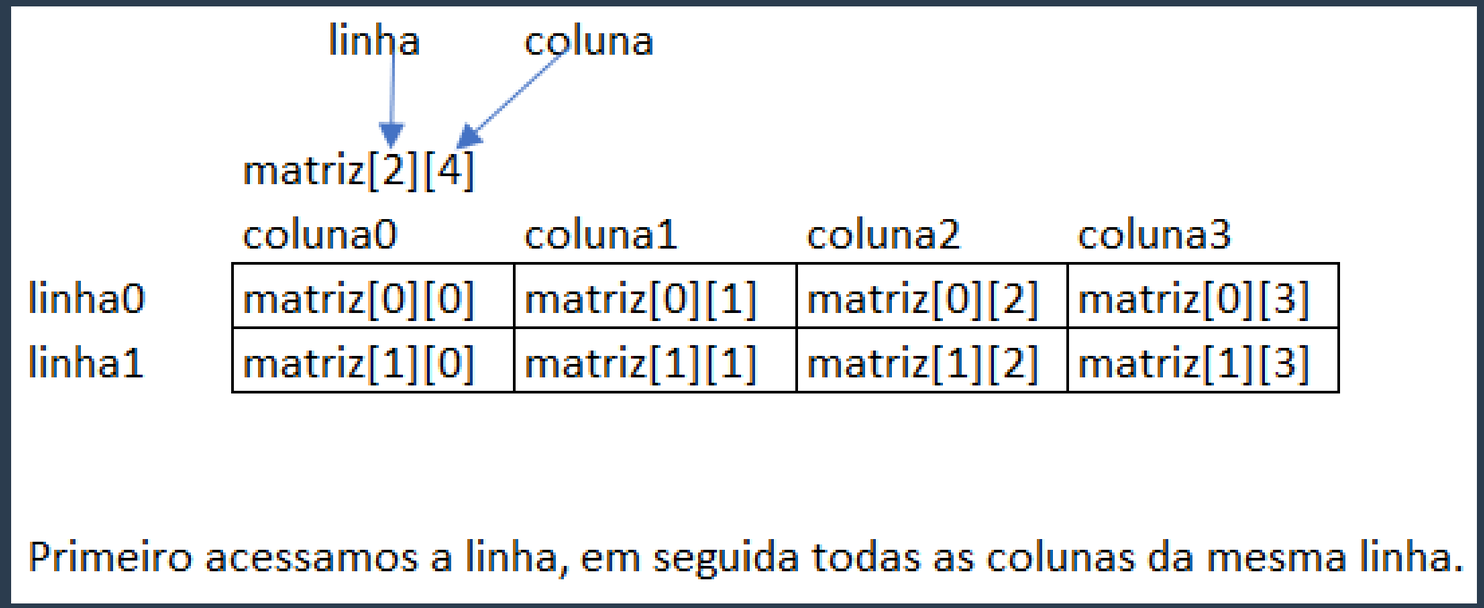
	coluna0	coluna1	coluna2	coluna3
linha0	matriz[0][0]	matriz[0][1]	matriz[0][2]	matriz[0][3]
linha1	matriz[1][0]	matriz[1][1]	matriz[1][2]	matriz[1][3]

EXEMPLO

Programa que recebe diversos valores digitados pelo usuário até preencher uma matriz 2x4.

Solicite ao usuário para digitar valores entre 0 e 100.

Depois imprime o maior valor dentro da matriz.



EXEMPLO – RESOLUÇÃO

```
programa {
	funcao inicio()
	{	// Define as dimensões (linhas e colunas) da matriz
		const inteiro numLinha = 2, numColuna = 4

		inteiro linha, coluna, maior = 0
		inteiro matriz[numLinha][numColuna] // Cria a matriz

		para (linha = 0; linha < numLinha; linha++)
		{
			para (coluna = 0; coluna < numColuna; coluna++)
			{
				escreva("Digite um número entre 0 e 100: ")
				// Atribui o valor digitado à posição da matriz
				leia (matriz[linha][coluna])

				se(matriz[linha][coluna] > maior)
					maior = matriz[linha][coluna]
			}
		}

		escreva("O maior valor da matriz = ", maior)
	}
}
```

```
#include <stdio.h>
int main(){

    int linha, coluna, matriz[2][4];

    for(linha = 0; linha < 2; linha++) {
        for(coluna = 0; coluna < 4; coluna++){
            printf("Digite valores de 0 a 100: ");
            scanf("%d", &matriz[linha][coluna]);
        }
    }

    for(linha = 0; linha < 2; linha++) {
        for(coluna = 0; coluna <4; coluna++){
            printf("%d ", matriz[linha][coluna]);
        }
        printf("\n");
    }
}
```

EXEMPLO – RESOLUÇÃO 2 – foi adicionado a impressor do valor final de i e j

```
programa
{
	funcao inicio()
	{
		inteiro vet[2][4], posicao=0, i=0, j=0, maior=0

		para(i=0; i<=1; i++){
			para(j=0; j<=3; j++){
				escreva("Digite um valor entre 0 e 100: ")
				leia(vet[i][j])

				se(i==0 e j==0)
					maior = vet[i][j]

				se(maior < vet[i][j])
					maior = vet[i][j]
			}
		}

		escreva("Maior valor = ", maior)
		escreva("\nValor de I = ", i)
		escreva("\nValor de J = ", j)
	}
}
```

EXEMPLO – RESOLUÇÃO

Programa que preenche uma matriz 2x4 e depois imprime os valores da matriz.

```c
#include <stdio.h>
int main(){

    int linha, coluna, matriz[2][4];

    for(linha = 0; linha < 2; linha++) {
        for(coluna = 0; coluna < 4; coluna++){
            printf("Digite valores de 0 a 100: ");
            scanf("%d", &matriz[linha][coluna]);
        }
    }

    for(linha = 0; linha < 2; linha++) {
        for(coluna = 0; coluna <4; coluna++){
            printf("%d ", matriz[linha][coluna]);
        }
        printf("\n");
    }
}
```

EXEMPLO

Programa que procura um valor digitado pelo usuário, dentro do vetor de cinco posições com valores pré-definidos. Se encontrar imprime “Valor encontrado na posição i”. Caso contrário imprime “Valor não encontrado”.

EXEMPLO – RESOLUÇÃO A – SEM LAÇO DE REPETIÇÃO

```
programa
{
    funcao inicio() {
        const inteiro tamanhoVetor = 5
        inteiro valor, numeros[tamanhoVetor] = {10, 11, 39, 45, 55}

        escreva("Digite um valor para ver se esta no vetor ")
        leia(valor)
        se(valor == numeros[0]) {
            escreva("Valor encontrado na posição: 0")
        }
        senao se(valor == numeros[1]) {
            escreva("Valor encontrado na posição: 1")
        }
        senao se(valor == numeros[2]) {
            escreva("Valor encontrado na posição: 2")
        }
        senao se(valor == numeros[3]) {
            escreva("Valor encontrado na posição: 3")
        }
        senao se(valor == numeros[4]) {
            escreva("Valor encontrado na posição: 4")
        }
        senao {
            escreva("Valor NÃO encontrado")
        }
    }
}
```

Exercícios

FAÇA OS EXERCÍCIOS EM LINGUAGEM DE PROGRAMAÇÃO C

EXERCÍCIO 1

Faça um programa que recebe do usuário os valores de uma matriz 2x2. Logo após imprime esta matriz.

EXERCÍCIO 1 – RESOLUÇÃO

```
programa {
    funcao inicio()
    {    // Define as dimensões (linhas e colunas) da matriz
        const inteiro TAMANHO = 2

        inteiro matriz[TAMANHO][TAMANHO] // Cria a matriz

        para (inteiro linha = 0; linha < TAMANHO; linha++)
        {
            para (inteiro coluna = 0; coluna < TAMANHO; coluna++)
            {
                escreva("Digite um número: ")
                // Atribui o valor digitado à posição da matriz
                leia (matriz[linha][coluna])
            }
        }

        escreva(" ---- Matriz digitada foi ----")
        para (inteiro linha = 0; linha < TAMANHO; linha++)
        {
            para (inteiro coluna = 0; coluna < TAMANHO; coluna++)
            {    // Exibe o valor contido em cada posição da matriz
                escreva("[", matriz[linha][coluna], "]")
            }
        }
    }
}
```

EXERCÍCIO 2

Faça um programa que solicita ao usuário para digitar números até preencher uma matriz 2x3, avise ao usuário que pode ser digitado valores entre 1 e 100. Após imprima o maior e o menor valor da matriz e sua posição dentro da matriz.

Solução

EXERCÍCIO 2 – RESOLUÇÃO

```c
#include <stdio.h>
#include <locale.h>
int main(void) {
    setlocale(LC_ALL, "Portuguese");
    int li=0, col=0, maior=0, menor=100, pM1=0, pM2=0, pMenor1=0, pMenor2=0, M[2][3];

    for(li=0; li<2; li++)   {
        for(col=0; col<3; col++) {
            printf("Digite um número: %d%d da matriz = ", li, col);
            scanf("%d", &M[li][col]);
        }
    }
    //imprimindo o maior e menor valor e suas posições na matriz
    for(li=0; li<2; li++) {
        for(col=0; col<3; col++) {
            if(maior < M[li][col])  {
                maior = M[li][col];
                pM1 = li;
                pM2 = col;
            }

            if(menor > M[li][col])  {
                menor = M[li][col];
                pMenor1 = li;
                pMenor2 = col;
            }
        }
    }
    printf("\nO MAIOR número é %d e esta na posição %d%d ", maior, pM1, pM2);
    printf("\n\nO menor número é %d e esta na posição %d%d ", menor, pMenor1, pMenor2);

    return 0;
}
```

Fim

Figura 2 – Computação x robótica (Adaptado)
Fonte: https://tecnico.ulisboa.pt/pt/tag/robotica/

Bibliografia

Básica
1 - Entendendo Algoritmos: Um Guia Ilustrado Para Programadores e Outros Curiosos
Autor: Aditya Bhargava
Editora: Novatec Editora; 1ª edição (24 abril 2017), Idioma: Português, ISBN-10: 8575225634, ISBN-13: 978-8575225639

2 - C - Como Programar [6ª Ed][Deitel & Deitel][2011]
Autor: Paul Deitel, Harvey Deitel
ASIN: B013H5WJR6, Editora: Pearson; 6ª edição (6 agosto 2015), Idioma: Português

3 - Raciocínio Lógico Passo a Passo
Autor: Mauro César Nunes e Luiz Claúdio Cabral. Editora Elsevier, 2013.
ISBN 978-85-352-7005-1, ISBN (versão digital): 978-85-352-7006-8

Complementar
1 - C - Algoritmos Teoria e Prática 2ª edição
Autor: CORMEN, T H LEISERSON, C E RIVEST, R L STEIN, Rio de Janeiro, RJ Campus, 2002

2 - Lógica de programação com Portugol
Autor: Joice Barbosa Mendes, Rafael da Silva Muniz, ISBN:978-85-5519-291-3, Data publicação:02/2022

3 - Algoritmos e Lógica de Programação
Autor: Marco A. Furlan de Souza (Autor), Marcelo Marques Gomes (Autor), Marcio Vieira Soares (Autor), Ricardo Concilio (Autor)
Editora: Cengage Learning; 3ª edição (10 janeiro 2019), Idioma: Português, ISBN-10: 8522128146, ISBN-13: 978-8522128143

4 - Introdução à Programação com Python: Algoritmos e Lógica de Programação Para Iniciantes
Autor: Nilo Ney Coutinho Menezes. Editora: Novatec Editora; 3ª edição (8 janeiro 2019)
Idioma: Português, ISBN-10: 8575227181, ISBN-13: 978-8575227183

5 - Aprenda Lógica de Programação e Algoritmos com Implementações em Portugol, Scratch, C, Java, C# e Python
Autor: Cláudio Luís Vieira Oliveira
Editora: Ciência Moderna (1 janeiro 2016), Idioma: Português, ISBN-10: 8539907798, ISBN-13: 978-8539907793