



Estrutura de Dados

Prof. Amarildo Lucena

Recursividade

Tipos de Dados





Tipos Primitivos de Dados

São os tipos mais simples de dados, também chamados de básicos, que geralmente são incluídos diretamente na memória. Para armazenar valores, existem 5 tipos básicos:

- ✓ char;
- ✓ int;
- ✓ float;
- ✓ double;
- ✓ void.

Todos esses tipos podem ser modificados para representar apenas valores positivos precedendo o tipo com o modificador 'sem sinal' unsigned.



Tipos Primitivos de Dados

Tipo	Palavra Chave	Tamanho
Caractere	char	-128 a 127
Inteiro	int	-2.147.483.648 a 2.147.483.647
Real de Precisão Simples	float	$1,2 \times 10^{-38}$ a $3,4 \times 10^{38}$
Real de Precisão Dupla	double	$2,2 \times 10^{-308}$ a $1,8 \times 10^{308}$
Vazio (sem valor)	Void	



Tipos Primitivos de Dados

Com exceção de void, os outros tipos de dados primitivos podem receber modificadores. Os modificadores alteram o tamanho do tipo de dado ou sua forma de representação. Sua utilização faz com que seja possível adequar-se melhor às necessidades de armazenamento de dados em determinados casos.

Os modificadores são:

Tipo	Palavra Chave
Caractere	signed
Inteiro	unsigned
Longo	long
Curto	short



Tipos Primitivos de Dados

Tipo	Palavra Chave	Representatividade	Identificador
char	1 byte	-128 a 127	%c
unsigned char	1 byte	0 a 255	%c
short int	2 byte	-32.768 a 32.767	%d ou %i
unsigned short int	2 bytes	0 a 65.535	%d ou %i
int	4 bytes	-2.147.483.648 a 2.147.483.647	%d ou %i
unsigned int	4 bytes	0 a 4.294.967.295	%d ou %i
long int	4 bytes	-2.147.483.648 a 2.147.483.647	%d ou %i
unsigned long int	4 bytes	0 a 4.294.967.295	%d ou %i
float	4 bytes	$1,2 \times 10^{-38}$ a $3,4 \times 10^{38}$	%f ou %e
double	8 bytes	$2,2 \times 10^{-308}$ a $1,8 \times 10^{308}$	%f ou %e



Tipos Primitivos de Dados

Declarando e utilizando variáveis (int)

Exemplo 1:

```
#include <stdio.h>
int main()
{
    int ano = 2019;
    printf("Estamos no ano de %d\n\n", ano);
    return 0;
}
```




Tipos Primitivos de Dados

Declarando e utilizando variáveis (int)

Exemplo 2:

```
#include <stdio.h>
int main()
{
    int N1 = 10, N2 = 5;
    printf("%d + %d = %d\n\n", N1, N2, N1 + N2);
    return 0;
}
```



Tipos Primitivos de Dados

Declarando e utilizando variáveis (float)

Exemplo 3:

```
#include <stdio.h>
int main()
{
    int ano = 2019;
    float dias = 365.25;
    printf("Estamos no ano de %d.\n\n", ano);
    printf("Um ano tem %.2f dias.\n\n", dias);
    return 0;
}
```



Tipos Primitivos de Dados

Declarando e utilizando variáveis (**double**)

Exemplo 4:

```
#include <stdio.h>
int main()
{
    int ano = 2019;
    double dias = 365.25;
    printf("Estamos no ano de %d.\n\n", ano);
    printf("Um ano tem %.2f dias.\n\n", dias);
    return 0;
}
```



Tipos Primitivos de Dados

Declarando e utilizando variáveis (char)

Exemplo 5:

```
#include <stdio.h>
#include <locale.h>
int main()
{
    setlocale(LC_ALL, "portuguese");
    char letra = 'A';
    printf("O Código ASCII %i = %c.\n\n", letra, letra);
    return 0;
}
```



Tipos Primitivos de Dados

Declarando e utilizando variáveis booleanas (**bool**)

Na linguagem C, o tipo booleano não faz parte das variáveis primitivas, necessitando portanto, de um cabeçalho específico, o `stdbool.h`.

Exemplo 6:

```
#include <stdio.h>
#include <stdbool.h>
int main()
{
    bool flag = true;
    printf("Valor do flag: %i.\n\n", flag);
    return 0;
}
```



Tipos Abstratos de Dados

São dados que possuem dentro de si dados primitivos e procedimentos. Na programação orientada a objetos, seriam os próprios objetos. Alguns tipos abstratos de dados:

- ✓ vetores;
- ✓ ponteiros;
- ✓ registros;
- ✓ listas;
- ✓ listas encadeadas.



Tipos Abstrato de Dados

Declarando e utilizando variáveis (**vetor**)

O que é uma variável vetor?

É uma variável que possui a capacidade de armazenar mais de um dado do mesmo tipo. Os dados são acessados a partir de um índice que começa com o valor zero.

Os vetores podem ser dos seguintes tipos:

- ✓ Uni-Dimensional ou Array;
- ✓ Multi-Dimensional ou Matriz.



Tipos Abstrato de Dados

Declarando e utilizando variáveis (**vetor**)

Possui a capacidade de armazenar mais de um dado do mesmo tipo em uma única coluna de dados. Seu índice inicia em 0 conforme mostrado abaixo:

X [0] = 1	0	1
X [1] = 2	1	2
X [2] = 3	2	3
X [3] = 4	3	4
X [4] = 5	4	5



Tipos Abstrato de Dados

Declarando e utilizando variáveis (vetor)

Exemplo 7:

```
#include <stdio.h>
int main()
{
    int vetor[5] = {1, 2, 3, 4, 5};
    printf("%i\n", vetor[0]);
    printf("%i\n", vetor[1]);
    printf("%i\n", vetor[2]);
    printf("%i\n", vetor[3]);
    printf("%i\n", vetor[4]);
    printf("Tamanho do vetor: %i bytes.\n", sizeof(vetor));
    return 0;
}
```



Tipos Abstrato de Dados

Declarando e utilizando variáveis (vetor)

Exemplo 8:

```
#include <stdio.h>
int main()
{
    int vetor[5] = {1, 2, 3, 4, 5};
    for (int x = 0; x < 5; x++)
    {
        printf("%i\n", vetor[x]);
    }
    printf("Tamanho do vetor: %i elementos.\n", sizeof(vetor)/sizeof(int));
    getchar();
    return 0;
}
```



Tipos Abstrato de Dados

Exemplo 9:

```
#include <stdio.h>
int main()
{
    int vetor[5] = {1, 2, 3, 4, 5};
    for (int x = 0; x < 5; x++)
    {
        printf("%i\n", vetor[x]);
    }
    printf("Tamanho do vetor: %i bytes.\n", sizeof(vetor));
    printf("Tamanho do vetor: %i elementos.\n", sizeof(vetor)/sizeof(int));
    getchar();
    return 0;
}
```



Tipos Abstratos de Dados

Exemplo 10:

```
#include <stdio.h>
#include <locale.h>
int main()
{
    setlocale(LC_ALL, "portuguese");
    float notas[2];
    printf("Insira sua primeira nota: ");
    scanf("%f", &nota[0]);
    printf("Insira sua segunda nota: ");
    scanf("%f", &nota[1]);
    printf("\nSua media é: %.2 ", (notas[0] + notas[1])/2);
    return 0;
}
```



Tipos Abstratos de Dados

Exemplo 11:

```
#include <stdio.h>
int main()
{
    char nome[] = "Recife";
    printf("nome[0]=%c\n", nome[0]);
    printf("nome[1]=%c\n", nome[1]);
    printf("nome[2]=%c\n", nome[2]);
    printf("nome[3]=%c\n", nome[3]);
    printf("nome[4]=%c\n", nome[4]);
    printf("nome[5]=%c\n", nome[5]);
    printf("\n%s tem %i caracteres.\n", nome, sizeof(nome)-1);
    return 0;
}
```



Tipos Abstrato de Dados

Declarando e utilizando variáveis (**Matriz**)

Possui a capacidade de armazenar mais de um dado do mesmo tipo em uma ou mais linhas e colunas de dados. Seu índice inicia em 0 conforme mostrado abaixo:

$X[0][0] = 1$

$X[0][1] = 4$

$X[1][0] = 2$

$X[1][1] = 5$

$X[2][0] = 3$

$X[2][1] = 6$

	0	1
0	1	4
1	2	5
2	3	6



Tipos Abstratos de Dados

Exemplo 12:

```
#include <stdio.h>
int main()
{
    int matriz[3][2] = {1, 4, 2, 5, 3, 6};
    printf("%i\n", matriz[0][0]);
    printf("%i\n", matriz[1][0]);
    printf("%i\n", matriz[2][0]);
    printf("%i\n", matriz[1][1]);
    printf("%i\n", matriz[2][1]);
    printf("%i\n", matriz[3][1]);
    printf("Tamanho do matriz: %i bytes.\n", sizeof(matriz));
    getchar();
    return 0;
}
```



Tipos Abstratos de Dados

Exemplo 13:

```
#include <stdio.h>
int main()
{
    int matriz[3][2] = {{1, 4}, {2, 5}, {3, 6}};
    printf("%i\n", matriz[0][0]);
    printf("%i\n", matriz[1][0]);
    printf("%i\n", matriz[2][0]);
    printf("%i\n", matriz[1][1]);
    printf("%i\n", matriz[2][1]);
    printf("%i\n", matriz[3][1]);
    printf("Tamanho do matriz: %i bytes.\n", sizeof(matriz));
    getchar();
    return 0;
}
```




Tipos Abstratos de Dados

Exemplo 14:

```
#include <stdio.h>
int main()
{
    int matriz[3][2];
    matriz[0][0] = 1;
    matriz[1][0] = 2;
    matriz[2][0] = 3;
    matriz[1][1] = 4;
    matriz[2][1] = 5;
    matriz[3][1] = 6;

    ...

    return 0;
}
```



Tipos Abstratos de Dados

Busca Sequencial

	0	1	2	3	4	5	6
i=0	23	4	67	-8	54	90	21
i=1	23	4	67	-8	54	90	21
i=2	23	4	67	-8	54	90	21
i=3	23	4	67	-8	54	90	21
i=4	23	4	67	-8	54	90	21



Tipos Abstratos de Dados

```
#include <stdio.h>
#include <locale.h>
int main()
{
    int vetor[] = {23, 4, 67, -8, 54, 90, 1};
    setlocale(LC_ALL, "portuguese");
    int tamanho = sizeof(vetor)/sizeof(int);
    int i = 0, valor;
    printf("Informe o valor a ser procurado: ");
    scanf("%d", &valor);
    for (i = 0; i < tamanho; i++) {
        if (vetor[i] == valor) {
            printf("A posição do valor %d é %d", valor, i);
            return 0;
        }
    }
    printf("O valor %d não foi encontrado!", valor);
    getchar();
    return 0;
}
```



Tipos Abstratos de Dados

Busca Binária

	0	1	2	3	4	5	6	7	8	9
meio=4	-8	-5	1	4	14	21	23	54	67	90
	0	1	2	3	4	5	6	7	8	9
meio=1	-8	-5	1	4	14	21	23	54	67	90
	0	1	2	3	4	5	6	7	8	9
meio=2	-8	-5	1	4	14	21	23	54	67	90
	0	1	2	3	4	5	6	7	8	9
meio=3	-8	-5	1	4	14	21	23	54	67	90



Tipos Abstratos de Dados

```
...
int tam = sizeof(vetor)/sizeof(int);
int inf = 0, sup = (tam - 1), meio, valor;
printf("Informe o valor a ser procurado: ");
scanf("%d", &valor);
while (inf <= sup) {
    meio = (inf + sup)/2;
    if (valor == vetor[meio]) {
        printf("A posição do valor %d é %d", valor, meio);
        return 0;
    }
    if (valor < vetor[meio])
        sup = meio - 1;
    else
        inf = meio + 1;
}
printf("O valor %d não foi encontrado!", valor);
getchar();
return 0;
}
```