
Faculdade Imaculada Conceição do Reci
UBEC
FICR
FICR

Estrutura de Dados

Sejam bem-vindos!

Bibliografias

Bibliografia Básica:

VETORAZZO, Adriana de Souza; SARAIVA, Maurício de Oliveira; BARRETO, Jeanine dos Santos; JUNIOR, Ramos **Estrutura de Dados.** Ed. Grupo A, 2018.

CURY, Thiago Espíndola; BARRETO, Jeanine dos Santos; SARAIVA, Maurício de Oliveira; VETTORAZZO, Adriana de Souza. **Estrutura de Dados.** Ed. Grupo A, 2018

PINTO, Rafael Albuquerque. Estrutura de Dados. Ed. Grupo A, 2020.

Bibliografia Complementar:

DROZDEK, Adam. **Estrutura de Dados e Algoritmos em C++** – Tradução da 4ª edição norte-americana. Ed. Cengage, 2018.

BIANCHI, Francisco. **Estrutura de Dados e Técnicas de Programação.** Ed. Grupo GEN, 2014.

KOFFMAN, Elliot B.; WOLFGANG, Paul A. T. **Objetos, Abstração, Estrutura de Dados e Projeto Usando C++.** Ed. Grupo GEN, 2008.

Tipos de Dados

Ponteiro

Tipos Abstratos de Dados

(Ponteiro)

O que é um Ponteiro ?

O Ponteiro ou Apontador, é uma variável que armazena endereços de memória.

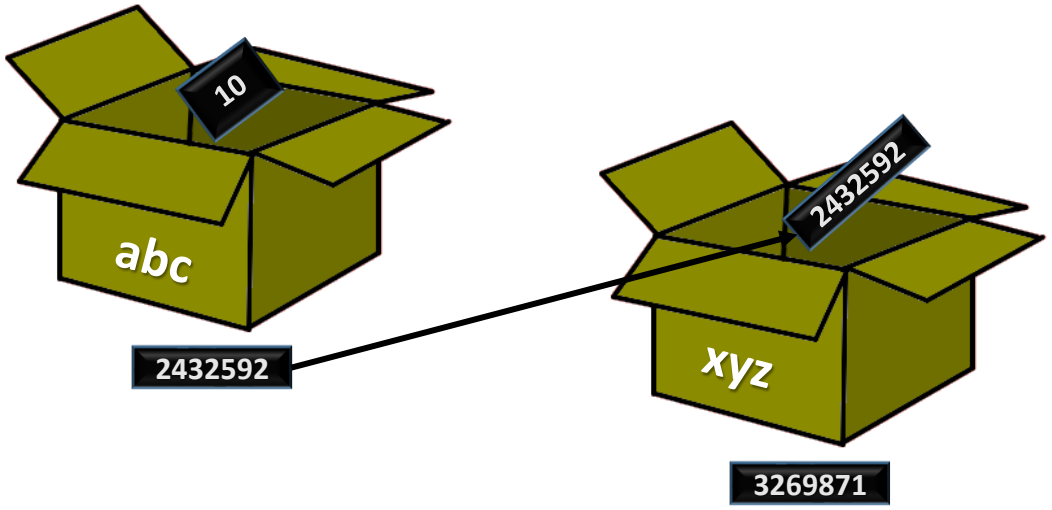
```
#include <stdio.h>

int main()
{
    int x = 10;
    int *ponteiro;
    ponteiro = &x;
    int y = 20;
    *ponteiro = y;
    printf("x: %i\ny: %i", x, y);
    getchar();
    return 0;
}
```

Tipos Abstratos de Dados

(Ponteiro)

Funcionamento do Ponteiro

```
int abc;
int *xyz;
xyz = &abc;
*xyz = 10;
```



Tipos Abstratos de Dados

(Ponteiro)

```
#include <stdio.h>
int main()
{
    int x = 10;
    int *ponteiro;
    ponteiro = &x;
    printf("valor de x: %i\n", x);
    printf("End.  de x: %i\n", &x);
    printf("&ponteiro : %i\n", &ponteiro);
    printf("ponteiro  : %i\n", ponteiro);
    printf("*ponteiro : %i\n", *ponteiro);
    getchar();
    return 0;
}
```

Tipos Abstratos de Dados

(Ponteiro)

```
#include <stdio.h>
int main()
{
    char x = 'F';
    char *ponteiro;
    ponteiro = &x;
    printf("valor de x: %c\n", *ponteiro);
    getchar();
    return 0;
}
```

Tipos Abstratos de Dados

(Ponteiro)

```
#include <stdio.h>
int verifica(int n1, int n2, int *soma, int *produto);
int main()
{
    int n1 = 10, n2 = 10, s = 0, p = 0;
    printf("Valor de Retorno: %i\n\n",verifica(n1,n2,&s,&p));
    printf("Soma = %i\n\n",s);
    printf("Produto = %i\n\n",p);
    getchar();
    return 0;
}

int verifica(int n1, int n2, int *soma, int *produto)
{
    printf("N1: %i\n\n",n1);
    printf("N2: %i\n\n",n2);
    *soma = (n1 + n2);
    *produto = (n1 * n2);
    if (n1==n2)
       return 0;
    else
       if (n1>n2)
          return 1;
       else
          return 2;
}
```

Tipos de Dados

Ponteiro com Vetor

Tipos Abstratos de Dados

(Ponteiro com Vetor)

Um Ponteiro pode se comportar como um vetor, permitindo o acesso a dados através da utilização de índices.

```
#include <stdio.h>
int main (void)
{
    int vetor[2],*v;
    v = vetor;
    v[0] = 123;
    v[1] = 456;
    printf ("vetor[0] = %d\n", vetor[0]);
    printf ("vetor[1] = %d\n", vetor[1]);
    getchar();
    return 0;
}
```

Tipos Abstratos de Dados

(Ponteiro com Vetor)

```
#include <stdio.h>
int main (void)
{
    int *v, vetor[5]={2,4,6,8,10};
    v=&vetor[0];
    printf ("*v \t= %d\n", *v);
    v++;
    printf ("v++\n");
    printf ("*v \t= %d\n", *v);
    v--;
    printf ("v--\n");
    printf ("*v \t= %d\n", *v);
    printf ("*(v+1) \t= %d\n", *(v+1));
    printf("v[2] \t= %d\n", v[2]);
    printf("v[3] \t= %d\n", v[3]);
    getchar();
    return 0;
}
```

Tipos de Dados

Ponteiro de Ponteiro

Tipos Abstratos de Dados

(Ponteiro de Ponteiro)

```
#include <stdio.h>
int main()
{
    int nr = 10, *p1, **p2, ***p3;
    p1 = &nr;
    p2 = &p1;
    p3 = &p2;
    printf("Valor de    nr: %i\n",    nr);
    printf("Valor de   *p1: %i\n",   *p1);
    printf("Valor de  **p2: %i\n",  **p2);
    printf("Valor de ***p3: %i\n", ***p3);
    getchar();
    return 0;
}
```

Tipos Abstratos de Dados

(Ponteiro de Ponteiro)

```
#include <stdio.h>
int main (void)
{
    int vetor[1],*v;
    v = vetor;
    v[0] = 123;
    printf ("vetor      = %d\n", vetor);
    printf ("&vetor     = %d\n", &vetor);
    printf ("&vetor[0] = %d\n", &vetor[0]);
    printf ("v          = %d\n", v);
    printf ("&v         = %d\n", &v);
    printf ("&v[0]      = %d\n", &v[0]);
    getchar();
    return 0;
}
```

Tipos Abstratos de Dados

Registro

Tipos Abstratos de Dados
(Registro)

O que é um Registro?

É uma coleção de variáveis que podem ser de tipos diversos e referenciada por apenas um nome.

```
#include <stdio.h>

struct <Nome>
{
        <tipo> <nome_var_1>;
        <tipo> <nome_var_2>;
        <tipo> <nome_var_3>;
        <tipo> <nome_var_n>;
};
```

Registro

Exemplo 1

Tipos Abstratos de Dados

(Registro)

```
#include <stdio.h>
int main()
{
    typedef struct
    {
        int hora;
        int minuto;
        int segundo;
    }horario;
    horario inicio;
    inicio.hora = 8;
    inicio.minuto = 10;
    inicio.segundo = 20;
    printf("Inicio: %i: %i: %i\n",inicio.hora,inicio.minuto,inicio.segundo);
    getchar();
    return 0;
}
```

Registro

Exemplo 2

```c
#include <stdio.h>
#include <stdlib.h>
struct s_dados
{
    char nome[20];
};
int main()
{
    int i, qtd;
    printf("Quantidade registros: ");
    scanf("%d",&qtd); fflush(stdin);
    s_dados dados[qtd];
    for(i=0;i<qtd;++i)
    {
        printf("Nome: ");
        scanf(" %20[^\n]",&dados[i].nome); fflush(stdin);
    };
    system("CLS");
    for(i=0;i<qtd;++i)
        printf("Nome: %s\n",dados[i].nome);
    system("PAUSE");
    return 0;
}
```

Registro

Exemplo 3

Tipos Abstratos de Dados

(Registro)

```
#include <stdio.h>
#include <stdlib.h>
#define MAX 3
////////////////////////////////////////////////////////
typedef struct
{
    char nome[20];
}s_conjuge;
////////////////////////////////////////////////////////
typedef struct
{
    char nome[20];
    int  fone[10];
    s_conjuge conjuge;
}s_cliente;
////////////////////////////////////////////////////////
```

```
int main(void)
{
    s_cliente cliente[MAX];
    int i;
    for(i=0;i<MAX;i++)
    {
        printf("Ficha  : %d\n",i+1);
        printf("Nome   : ");
        scanf(" %20[^\n]",&cliente[i].nome);
        printf("Fone   : ");
        scanf(" %10[^\n]",&cliente[i].fone);
        printf("conjuge: ");
        scanf(" %20[^\n]",&cliente[i].conjuge.nome);
        system("cls");
    };
    for(i=0;i<MAX;i++)
    {
        printf("Ficha  : %d\n",i+1);
        printf("---------------------------------------\n");
        printf("Nome   : %s\n",cliente[i].nome);
        printf("Fone   : %s\n",cliente[i].fone);
        printf("Conjuge: %s\n",cliente[i].conjuge.nome);
        printf("---------------------------------------\n");
    };
    system("pause");
    return 0;
}
```

Registro

Exemplo 4

Tipos Abstratos de Dados

(Registro)

```
#include <stdio.h>
#include <stdlib.h>
#define MAX 3
typedef struct
{
    char nome[2][20];
}s_filiacao;
typedef struct
{
    char nome[20];
    int  fone[10];
    s_filiacao filiacao;
}s_cliente;
```

```
int main(void)
{
    s_cliente cliente[MAX];
    int i;
    for(i=0;i<MAX;i++)
    {
        printf("Ficha: %d\n",i+1);
        printf("Nome : ");
        scanf(" %20[^\n]",&cliente[i].nome);
        printf("Fone : ");
        scanf(" %10[^\n]",&cliente[i].fone);
        printf("Pai  : ");
        scanf(" %20[^\n]",&cliente[i].filiacao.nome[0]);
        printf("Mae  : ");
        scanf(" %20[^\n]",&cliente[i].filiacao.nome[1]);
        system("cls");
    };
    for(i=0;i<MAX;i++)
    {
        printf("Ficha: %d\n",i+1);
        printf("----------------------------------------\n");
        printf("Nome : %s\n",cliente[i].nome);
        printf("Fone : %s\n",cliente[i].fone);
        printf("Pai  : %s\n",cliente[i].filiacao.nome[0]);
        printf("Mae  : %s\n",cliente[i].filiacao.nome[1]);
        printf("----------------------------------------\n");
    };
    system("pause");
    return 0;
}
```


Faculdade Imaculada Conceição do Reci
FICR
FICR