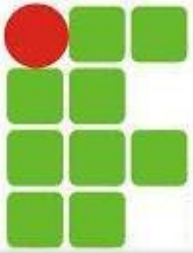
ALGORITMOS

Professor: Diego Oliveira

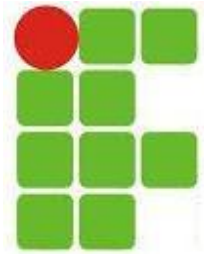


Aula 10 - Ponteiros

Ponteiros

- -Variáveis que armazenam endereços de memória.
- -Permitem manipular diretamente o conteúdo da memória.
- Sintaxe:

```
1  #include <stdio.h>
2
3  int main() {
4      int *p;
5
6      return 0;
7  }
```



Vantagens dos Ponteiros

- Eficiência no acesso e manipulação de dados.
- Passagem de parâmetros por referência.
- Manipulação de estruturas complexas (listas, árvores, etc.).
- Alocação dinâmica de memória.

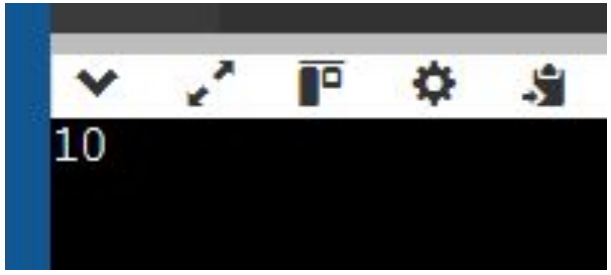
Exemplo de Ponteiro

- Código:

```c
int main() {
    int x = 10;
    int *p = &x; // p aponta para x
    printf("%d\n", *p); // imprime 10

    return 0;
}
```

- Saída:

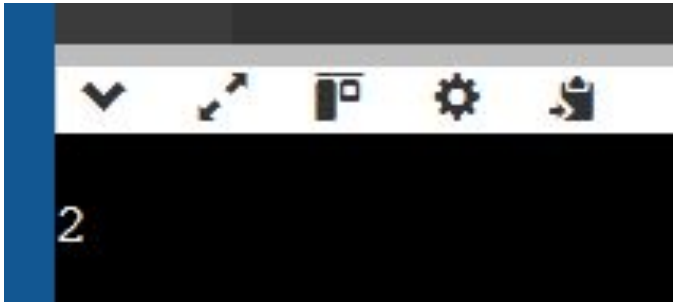


Exemplo de Ponteiro

- Vetores são tratados como ponteiros:

```
3  int main() {
4      int v[3] = {1,2,3};
5      int *p = v;
6      printf("%d\n", *(p+1)); // imprime 2
7
8      return 0;
9  }
```

- Saída:



Exemplo de Ponteiro

- Usamos **malloc** para criar vetores em tempo de execução:

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

int main() {
    int *v = (int*) malloc(5 * sizeof(int)
    v[0] = 10;
    printf("%d", v[0]);

    return 0;
}
```

- Saída:

Exemplo de Ponteiro

- Matrizes podem ser alocadas com ponteiros de ponteiros:

```c
int main() {
    int n = 3;  // número de linhas
    int m_col = 3; // número de colunas
    int **matriz;

    // aloca as linhas
    matriz = (int**) malloc(n * sizeof(int*));
    if (matriz == NULL) {
        printf("Erro de alocação!\n");
        return 1;
    }

    // aloca as colunas para cada linha
    for (int i = 0; i < n; i++) {
        matriz[i] = (int*) malloc(m_col * sizeof(int));
        if (matriz[i] == NULL) {
            printf("Erro de alocação!\n");
            return 1;
        }
    }
```

Exemplo de Ponteiro

- A liberação da memória acontece com **free**:

```
// inicializa a matriz com valor 5
for (int i = 0; i < n; i++) {
    for (int j = 0; j < m_col; j++) {
        matriz[i][j] = 5;
    }
}

// imprime um elemento da matriz
printf("Elemento [0][0] = %d\n", matriz[0][0]);

// libera memória
for (int i = 0; i < n; i++) {
    free(matriz[i]);
}
free(matriz);
```

Exercício

- Solicite ao usuário o tamanho de um vetor de inteiros
- Use alocação dinâmica (malloc) para criar esse vetor
- Preencha o vetor com valores aleatórios
- Utilize ponteiros para:
  - Imprimir todos os elementos do vetor
  - Calcular a soma dos elementos
  - Encontrar o maior e o menor valor
- Libere a memória alocada com free

Perguntas?