Estrutura de Dados I
Aula de Laboratório 01

1 Ponteiros

1.1 Sintaxe

A declaração de uma variável do tipo ponteiro segue a seguinte sintaxe:

```
<tipo do dado> * <nome da variável>;
```

Onde:

`<tipo do dado>`: pode ser substituído por um tipo primitivo da linguagem C (int, float, double, char, etc.) ou por um tipo *registro*, definido pelo próprio programador.

`<nome da variável>`: deve seguir as regras de nomes de variáveis e funções da linguagem C.

1.2 Operadores de ponteiro

- Operador *: acessa o valor armazenado no endereço de memória.
- Operador &: devolve o endereço de memória da variável.
- Formato %p: indica que o formato do valor a ser impresso é um endereço de memória.

1.3 Exemplos

```
1  #include <stdio.h>
2
3  int main () {
4      int x = 10;
5      int *p;
6      printf("Valor de x: %d\n", x);
7      printf("Endereco de x: %p\n", &x);
8      printf("Endereco armazenado em p: %p\n", p);
9      p = &x;
10     printf("Novo endereco armazenado em p: %p\n", p);
11     printf("Valor armazenado no endereco guardado em p: %d\n", *p);
12     *p = 15;
13     printf("Novo valor armazenado guardado em x: %d\n", x);
```

```
14      return 0;
15  }
```

```
1  #include <stdio.h>
2
3  int main () {
4      int V[5] = {1, 3, 6, 8, 19};
5      int i;
6      for (i=0;i<5;i++)
7          printf("%d\t",V[i]);
8      printf("\n");
9      printf("V = %p\n", V);
10     printf("V[0] = %d\n", V[0]);
11     printf("*V = %d\n", *V);
12     printf("&V[0] = %p\n", &(V[0]));
13     printf("V+1 = %p\n", V+1);
14     printf("*(V+1) = %d\n", *(V+1));
15     return 0;
16 }
```

1.4 Acessando campos da estrutura por meio de apontadores

```
1  #include <stdio.h>
2  #include <string.h>
3
4  struct aluno {
5      int id;
6      char nome[50];
7      float media;
8  };
9
10 int main () {
11     struct aluno A;
12     struct aluno *B;
13     A.id = 1001;
14     strcpy(A.nome,"Juliana");
15     A.media = 9.5;
16
17     printf(" Id: %d\n Nome: %s\n Media: %.2f\n",A.id,A.nome,A.media);
18
19     B = &A;
20
21     printf("Nome de B: %s\n", B->nome);
22     printf("Id de B: %d\n", (*B).id);
23
```

```
24      return 0;
25  }
```

1.5 Passagem de parâmetros por valor e por referência

```
1  #include <stdio.h>
2
3  void incremento (int a) {
4      a = a + 1;
5  }
6
7  void incrementoV2 (int *a) {
8      *a = *a + 1;
9  }
10
11 int main () {
12     int a = 0;
13     incremento(a);
14     printf("incremento(a) = %d\n", a);
15     incrementoV2(&a);
16     printf("incrementoV2(&a) = %d\n", a);
17     return 0;
18 }
```

2 Alocação Dinâmica

2.1 Funções

- malloc
- calloc
- realloc
- free

2.2 Alocação de estruturas homogêneas

- Alocação de vetor unidimensional
- Alocação de matriz: utilizando vetor de vetores
- Alocação de matriz: utilizando um único vetor