



Estruturas de Dados | Programação II

Engenharia Informática | Sistemas e Tecnologias de Informação 1º ano, 2º semestre 2022-2023

estgoh
Delitéanies de Coimbre

Ponteiros

Ponteiro

Variável que guarda o endereço de memória de outra variável (para onde aponta).

ou

Apontador

<tipo> * <nome_ponteiro> declaração de <nome_ponteiro> como um ponteiro que aponta para uma variável do tipo

<tipo>

(não reserva espaço de memória, apenas espaço para um endereço de memória)

conteúdo da zona de memória guardada pelo *<nome_ponteiro>

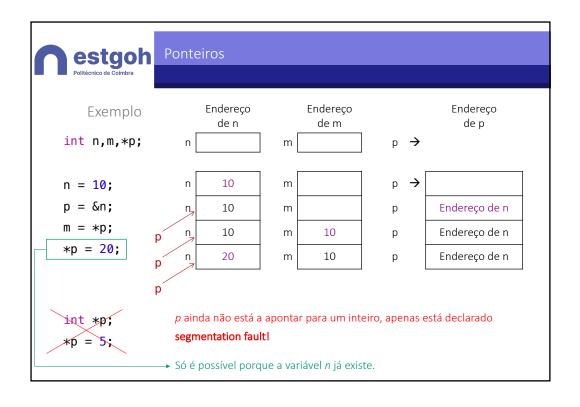
ponteiro <nome_ponteiro>

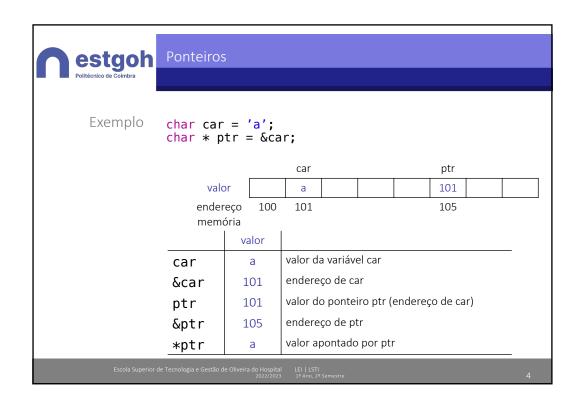
(valor para o qual < nome_ponteiro > aponta)

&<nome_ponteiro> endereço de memória de <nome_ponteiro>

*<nome_ponteiro> = NULL inicialização do ponteiro < nome_ponteiro >

como NULL, ou seja, a apontar para nada





```
Exemplos int n,*p;

p = &n;
*p = 5;

printf("n = %i",n);

ou

int *pont;

malloc pont = (int *)malloc(sizeof(int));
*pont = 5;

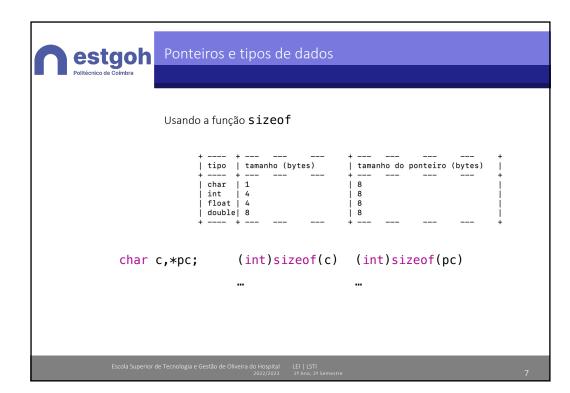
printf("valor = %i",*pont);

free free(pont);

Excola Superior de Tecnologia e Gestão de Oliveira de Hospital

LEI | LSTI
2022/2023 | 14 Ano. 24 Samestre
```

```
estgoh
                       Funções da biblioteca stdlib.h
           malloc
              free
                       void * malloc(size_t size)
                       The malloc() function allocates <a href="mailto:size">size</a> bytes of memory and returns a
                        pointer to the allocated memory.
                       If successful malloc() returns a pointer to allocated memory.
                       If there is an error, it returns a NULL pointer and set errno to
                        ENOMEM.
pont = (int *)malloc(sizeof(int));
                        void free(void *ptr)
                       The free() function deallocates the memory allocation pointed to by
                       ptr.
                       If ptr is a NULL pointer, no operation is performed.
                       The free() function does not return a value.
free(pont);
```





Aritmética de Ponteiros

Uma comparação (<, <=, >, >=, ==, !=) só pode ser realizada entre ponteiros do mesmo tipo.

scola Superior de Tecnologia e Gestão de Oliveira do Hospital LEI | LSTI



Ponteiros

Exemplo

```
char car = 'a';
char * ptr = &car;
```

variável	Identificador do formato	resultado
car	%C	a
	%i ou %d	97
&car ou ptr	%i	-534504853
	%s	a
	%p (pointer)	0x7ffee0241a6b
	%u (unsigned integer)	3760462443
	%li ou %ld	140732658883179
	%x ou %X (hexadecimal)	e0241a6b

Escola Superior de Tecnologia e Gestão de Oliveira do Hospita 2022/202: LEI | LSTI 1º Ano, 2º Semestre ^



Aritmética de Ponteiros e Tabelas

É possível realizar operações aritméticas com ponteiros (do mesmo tipo):

- Adicionar (ptr++; ptr = ptr + 2; ptr +=4; ptr1 + ptr2)
- Subtrair (ptr--; ptr = ptr 2; ptr -=4; ptr1 ptr2)
- Incrementar → avança no endereço de memória o número de bytes (sizeof(tipo de dados)) que o seu tipo ocupa
- Decrementar → recua no endereço de memória o número de bytes que o seu tipo ocupa

Estas operações podem ser úteis para manipular tabelas.

Escola Superior de Tecnologia e Gestão de Oliveira do Hospital 2022/2023 LEI | LSTI



Aritmética de Ponteiros e Tabelas

```
int v[] = {10,20,30,40,50};
int * pv;

pv = &v[0]; //atribui a pv o endereco do primeiro
elemento da tabela
printf("%i\n",*pv);

++pv; //ponteiro aponta para segundo elemento da tabela
printf("%i\n",*pv);

Resultado da execução:
10
20
```



Aritmética de Ponteiros e Tabelas

```
int v[] = \{10, 20, 30, 40, 50\};
           Exemplo
                            int * pv;
  Os elementos de uma
tabela ocupam posições
                            pv = v; //alternativa a pv = &v[0];
       consecutivas de
memória, sendo o nome
                            for(int j = 0; j < 5; j++)
    printf("v[%i] = %i\n",j,*(v+j));</pre>
      da tabela igual ao
  endereço do primeiro
     elemento, isto é, o
       menor endereço
                            Resultado da execução:
v[0] == *v
v[1] == *(v + 1)
v[2] == *(v + 2)
                            v[0] = 10
                            v[1] = 20
v[2] = 30
v[3] = 40
v[4] = 50
v[n] == *(v + n)
```

Escola Superior de Tecnologia e Gestão de Oliveira do Hospital LI



Aritmética de Ponteiros e Tabelas



Aritmética de Ponteiros e Tabelas

scola Superior de Tecnologia e Gestão de Oliveira do Hospital LEI | LSTI

```
Exemplo

float * soma(float *p1, float *p2)

float *p = NULL, *p0 = NULL;

p0 = (float *)malloc(sizeof(float)*DIM*DIM);

*p0 = 0;

p = p0;

for(int i = 1; i <= DIM*DIM; i++)

{
    *p = *p1 + *p2;
    p++;
    p1++;
    p2++;
}

return p0;
}
```

estgoh

Aritmética de Ponteiros e Tabelas

scola Superior de Tecnologia e Gestão de Oliveira do Hospital LEI | LSTI

```
Tabelas de ponteiros

Exemplo int v[] = {10,20,30,40,50};
int * pv;

pv = &v[0];
printf("%i\n",*pv);
++pv;
printf("%i\n",*pv);

int * ptab[10]; //tabela com 10 ponteiros para inteiros ptab[0] = pv;
printf("%i\n",*ptab[0]);

Escola Superior de Tecnologia e Gestão de Oliveira do Houpital 14 Ano. 24 Samestre 17
```

```
estgoh
                                             /*...*/
struct dados_aluno
                                                                                                                             (*nome_ponteiro).campo
                                                     char curso[5];
int numero;
char nome[50];
float media;
                Exemplo
                                                                                                                             nome_ponteiro->campo
                                             typedef struct dados_aluno aluno;
int main ()
{
        aluno a,*p;
        p = &a;
       //insercao dados aluno
strcpy((*p).nome,"ze");
strcpy((*p).curso,"grsi");
(*p).numero = 123;
(*p).media = 12.0f;
                                                                                                         strcpy(p->nome,"ze");
Strcpy(p->curso,"grsi");
p->numero = 123;
p->media = 12.0f;
                                                                                             ou
        //apresentacao dados aluno
printf("nome: %s\n",(*p).nome);
printf("curso: %s\n",(*p).curso);
printf("numero: %i\n",(*p).numero);
printf("media: %f\n",(*p).media);
                                                                                                         printf("nome: %s\n",p->nome);
printf("curso: %s\n",p->curso);
printf("numero: %i\n",p->numero);
printf("media: %f\n",p->media);
         return 0;
```

```
estgoh
                                                            aluno * le_aluno2()
{
       Exemplo
                                                               aluno *pa;
void le_aluno1(aluno *pa)
                                                               pa = (aluno*)malloc(sizeof(aluno));
   printf("curso: ");
                                                               printf("curso: ");
   gets(pa->curso);
printf("nome: ");
                                                               gets(pa->curso);
printf("nome: ");
   gets(pa->nome);
printf("numero: ");
scanf("%d",&pa->numero);
printf("media: ");
scanf("%f",&pa->media);
                                                               gets(pa=>nome);
printf("numero: ");
scanf("%d",&pa=>numero);
printf("media: ");
scanf("%f",&pa=>media);
return pa;
                                               No main:
aluno a, *p;
p = &a;
le_aluno1(p);
// ou apenas
                                                            aluno *p;
p = le_aluno2();
aluno a;
le_aluno1(&a);
```

```
estgoh
                      /*...*/
                      # define TAM 15
      Exemplo
                      /*...*/
(anterior com
        tabela)
                      int main()
                           // usando uma tabela de alunos
                           aluno t_alunos[TAM];
                           int i;
                           for(i=0;i<TAM;i++)</pre>
                                le_aluno1(&t_alunos[i]);
                                                                       // void le_aluno1(aluno *pa)
                           for(i=0;i<TAM;i++)</pre>
                                mostra_aluno(&t_alunos[i]); //void mostra_aluno(aluno *pa)
                           return 0;
                      }
                                                         void mostra_aluno(aluno *pa)
                                                              printf("nome: %s\n",pa->nome);
printf("curso: %s\n",pa->curso);
printf("media: %f\n",pa->media);
```