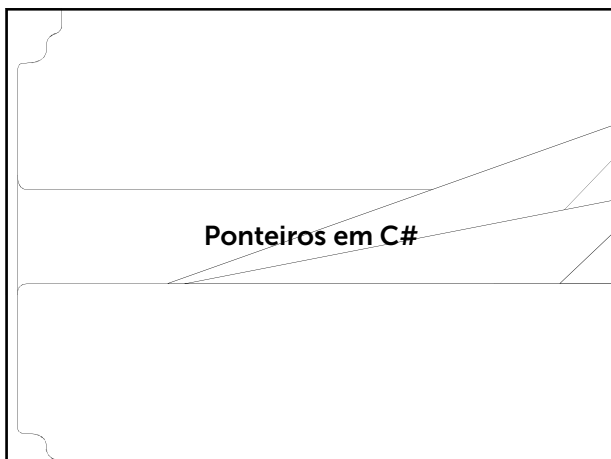


**Linguagem de Programação e
Estrutura de Dados**
Aula 3
Linguagem C#



Anderson Macedo
Especialista em Engenharia de Software com UML



Ponteiros em C#

Objetivo da aula

» Nesta aula vamos aprender como é feita a criação, a declaração e a manipulação de ponteiros na linguagem C#(Sharp), utilizando-se de exemplos criados em projetos “Console Application”.

Definição

- » Variáveis : endereçam uma posição de memória que contem um determinado valor dependendo do seu tipo (char, int, float, double, string...)

```
void main() {
    long a=5;
    char ch='x';
}
```

	endereço	valor	
a →	0x0100	0x00	}
	0x0101	0x00	
	0x0102	0x00	
	0x0103	0x05	
ch →	0x0104	0x78	'x'

Definição

- » Ponteiros: são variáveis cujo conteúdo é um endereço de memória.
- » Assim, um ponteiro endereça uma posição de memória que contém valores que são na verdade endereços para outras posições de memória.

```
void main() {
    long a=5;
    char ch='x';
    long *aPtr = &a;
}
```

	endereço	valor	
a →	0x0100	0x00	}
	0x0101	0x00	
	0x0102	0x00	
	0x0103	0x05	
ch →	0x0104	0x78	'c'
aPtr →	0x0105	0x00	}
	0x0106	0x00	
	0x0107	0x01	
	0x0108	0x00	

Criação de Ponteiros

- » Para declararmos um ponteiro, basta utilizar o operador *(asterisco) antes do nome da variável.

- » Exemplo:

```
int *p;
```

- » Ponteiros são tipados, ou seja, devem ser classificados em um tipo e somente podem apontar para variáveis deste mesmo tipo.

Operadores para Ponteiros

- » Para trabalharmos com ponteiros, C# disponibiliza os seguintes operadores:
 - » & - Fornece o endereço de memória onde está armazenado uma variável. Lê-se “o endereço de”.
 - » * - Valor armazenado na variável referenciada por um ponteiro. Lê-se “o valor apontado por”.

Operadores para Ponteiros

```
unsafe void macedo(
    long a=5;
    char ch='x';
    long *aPrt = &a;
    C.WL(*aPrt);
    C.WL(aPrt);
    C.WL(&aPrt);
}
```

	endereço	valor	
a →	0x0100	0x00	5
	0x0101	0x00	
	0x0102	0x00	
	0x0103	0x05	
ch →	0x0104	0x78	'c'
aPrt →	0x0105	0x00	
	0x0106	0x00	
	0x0107	0x01	
	0x0108	0x00	0x00000100

» O que será impresso na tela?

- » 5
- » 0x0100
- » 0x0105

Exemplo 1

```
void macedo ()
{
    int num,valor;
    int *p;
    num=55;
    p=&num; /* Pega o endereço de memória de num */
    valor=*p; /* Valor é igualado a num de uma maneira indireta */
    C.WL (valor);
    C.WL ("Endereco para onde o ponteiro aponta: " + p);
    C.WL ("Valor da variável apontada: " + *p);
}
```



Operadores para Ponteiros

- » Igualando ponteiros:
- » `int *p1, *p2;`
- » `p1=p2;`
 - » Repare que estamos fazendo com que p1 aponte para o mesmo lugar que p2.
- » Fazendo com que a variável apontada por p1 tenha o mesmo conteúdo da variável apontada por p2
- » `*p1=*p2;`

Exemplo 2

```
void macedo ()
{
    int num,*p1, *p2;
    num=55;
    p1=&num; /* Pega o endereço de num */
    p2=p1; /* p2 passa a apontar para o mesmo endereço apontado
    por p1 */
    C.WL("Conteúdo de p1: " + p1);
    C.WL("Valor apontado por p1: " + *p1);
    C.WL("Conteúdo de p2: " + p2);
    C.WL("Valor apontado por p2: " + *p2);
}
```

Operadores para Ponteiros

» Incremento/Decremento:

- » Apontar para o próximo valor do mesmo tipo para o qual o ponteiro aponta:

» long *aPtr, a=5;

» aPtr=&a;

» aPtr++;

	endereço	valor	
a →	0x0100	0x00	5
	0x0101	0x00	
	0x0102	0x00	
	0x0103	0x05	
ch →	0x0104	0x78	'c'
aPtr →	0x0105	0x00	
	0x0106	0x00	
	0x0107	0x01	
	0x0108	0x00	0x00000100

Exemplo 3

» Qual será o valor endereçado por aPtr++ ??

- » Se aPtr é long, como o long ocupa 4 bytes, aPtr irá apontar para o endereço 0x00000104
- » Este é o principal motivo que nos obriga a definir um tipo para um ponteiro!!!

	endereço	valor	
a →	0x0100	0x00	5
	0x0101	0x00	
	0x0102	0x00	
	0x0103	0x05	
ch →	0x0104	0x78	'c'
aPtr →	0x0105	0x00	
	0x0106	0x00	
	0x0107	0x01	
	0x0108	0x00	0x00000100

Operadores para Ponteiros

```
void macedo ()
{
    int num;
    int *p;
    num=55;
    p=&num;
    C.WL("Conteúdo de p: " + p);
    C.WL("Valor apontado por p: " + *p);
    C.WL("Conteúdo de p incrementado: " + (++p));
    C.WL("Valor apontado por p incrementado: " + *p);
}
```

