



# PCI- Estructura Condicional

Profa. Mercedes Gonzales  
Márquez

---

# Estilo codificação e indentação

- Os compiladores C ignoram espaços em branco, caracteres de tabulação e caracteres de mudança de linha. Por exemplo, o programa acima poderia ser escrito de uma das seguintes maneiras:

```
int main(  
)  
{  
printf  
("Ola pessoal");  
}
```

```
int main(){printf  
("Ola pessoal");  
}
```

```
int main(){  
    printf ("Ola pessoal");  
}
```

```
int main()  
{  
    printf ("Ola pessoal");  
}
```

- Mas qual deles é mais fácil de ler? Os dois últimos, não é? Embora não exista um estilo obrigatório de se editar programas em C, o estilo específico da sua escolha deve procurar seguir os seguintes princípios:

# Estilo codificação e indentação

- Escreva uma instrução (um comando) por linha. Quando for necessário, divida a instrução em mais de uma linha para melhorar a legibilidade (você pode quebrar a linha em qualquer lugar onde poderia colocar um espaço). Em algumas situações também pode ser aceitável juntar duas instruções em uma linha só, mas não abuse disso.

# Estilo codificação e indentação

- Considere uma correta indentação: Sempre que tivermos um *bloco* de comandos (algo delimitado por chaves), o conteúdo do bloco (as linhas que pertencem ao mesmo bloco) deve estar mais afastado da margem que o exterior do bloco.
- Embora a indentação modique o código apenas do ponto de vista estético, ela torna muito mais fácil a visualização da estrutura do código e de onde os blocos se encaixam facilitando a leitura e interpretação do programa.

# Estilo codificação e indentação

## Sem indentação

```
1  #include <stdio.h>
2
3  int main()
4  {
5  int x;
6  printf ("Digite o valor de X..: ")
7  scanf ("%d", &x);
8  if (x%2 == 0)
9  printf ("X é Par\n");
10 else
11 printf ("X é Ímpar\n");
12 system ("pause");
13 }
```

## Com indentação

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int x;
6      printf ("Digite o valor de X..: ");
7      scanf ("%d", &x);
8      if (x%2 == 0)
9          printf ("X é Par\n");
10     else
11         printf ("X é Ímpar\n");
12     system ("pause");
13 }
```

# Expressão lógica

- **Uma Expressão Lógica** retorna verdadeiro ou falso.
- Ela é formada pela aplicação de operadores relacionais e/ou operadores lógicos tais como e (and), ou (or) e não (not).
- Exemplos:

Expressão Lógica	Resultado
$x \geq y$	falso
$(x < y)$	verdadeiro
$x \geq (y - 2)$	verdadeiro
$(x > 0) \ \&\& \ (x < y)$	verdadeiro
$(y > 0) \ \&\& \ (x > 0) \ \&\& \ (x > y)$	falso
$(x < 0) \    \ (y > x)$	verdadeiro
$(x < 0) \    \ (y > x) \ \&\& \ (y < 0)$	falso

para  $x=1$  e  $y=2$ .

# Estrutura Condicional

- A verificação de que uma condição (expressão lógica) é satisfeita e, a partir daí, um determinado bloco de comandos deve ser executado é chamada de *estrutura condicional*, *de seleção*, *estrutura de decisão*, *comando condicional* ou *comando de seleção*.

# Estrutura Condicional Simples

A estrutura condicional simples permite decidir se um determinado bloco de comandos deve ou não ser executado, a partir do resultado de uma expressão lógica.

## ***Estrutura if..***

Executa código somente se uma condição for verdadeira (resultado da expressão diferente de zero)

```
if (expressão lógica) {  
    bloco de comandos  
}
```

# Estrutura Condicional Simples

```
int main( ){
    int idade, diferenca;
    printf("Digite sua idade: ");
    scanf("%d", &idade);
    if (idade >= 18) {
        diferenca = idade -18;
        printf("Voce e´ maior de idade ha´ %d ano(s)",
            diferenca);
    }
}
```

# Estrutura Condicional Simples

Observação importante: Podemos omitir as chaves quando não houver um bloco de comandos, mas apenas um único comando.

```
if (expressão lógica)
    comando;
```

```
int main( ){
    int idade;
    printf("Digite sua idade: ");
    scanf("%d", &idade);
    if (idade >= 18)
        printf("Voce e´ maior de idade,,");
}
```

# Estrutura Condicional Simples

Experimente a tirar as chaves {} do bloco do if no primeiro exemplo, o que acontece quando a idade for <18 ?

# Estrutura Condicional Composta

- A estrutura condicional composta decide entre dois blocos comandos qual vai ser executado.

## ***Estrutura if.. else ...***

Condição verdadeira: executa o primeiro bloco

Caso contrário: executa o segundo bloco

```
if (expressão lógica) {  
    bloco de comandos1;  
} else{  
    bloco de comandos2;  
}
```

# Estrutura Condicional Composta

```
int main() {
    int idade, diferenca_tempo;
    printf("Digite sua idade: ");
    scanf("%d", &idade);
    if (idade >= 18) {
        diferenca_tempo = idade - 18;
        printf("Voce eh maior de idade ha %d ano(s)",
diferenca_tempo);
    } else {
        diferenca_tempo = 18 - idade;
        printf("Espere mais %d ano(s)!\n", diferenca_tempo);
    }
}
```

# Estrutura Condicional Composta

Observação importante: Podemos omitir as chaves quando não houver um bloco de comandos, mas apenas um único comando, já seja pertencendo ao if ou ao else.

```
int main() {  
    int idade;  
    printf("Digite sua idade: ");  
    scanf("%d", &idade);  
    if (idade >= 18)  
        printf("Voce eh maior de idade");  
    else  
        printf("Voce ainda e´ menor de idade");  
}
```

# Estrutura Condicional Aninhada

Um if aninhado é um comando if que é o objeto de outro if ou else e assim por diante. Exemplo:

```
if (i){  
    if (j)  
        comando 1;  
    if (k)  
        comando 2;  
    else  
        comando 3; /* associado a if(k)*/  
}else  
    comando 4; /* associado a if (i)*/
```

O último else está associado ao if (i). O else interno está associado ao if(k) que é o if mais próximo

# Estrutura Condicional Aninhada

```
if (cond1)
    if (cond2)
        comando1;
else
    comando2;
```

Quando o comando2 é executado?

# Estrutura Condicional Aninhada

```
if (cond1)
    if (cond2)
        comando1;
    else
        comando2;
```

Quando o comando2 é executado?

# Estrutura Condicional Aninhada

```
if (cond1){  
    if (cond2)  
        comando1;  
}else  
    comando2;
```

Quando o comando2 é executado?

# Estrutura Condicional Aninhada

```
if (cond1){  
    if (cond2)  
        comando1;  
}else  
    comando2;
```

Quando o comando2 é executado?

Exercício: Faça um programa que determine o menor entre três números inteiros.

# Estrutura Condicional Aninhada

```
#include <stdio.h>
int main() {
    int a, b, c;
    printf("Digite tres numeros: ");
    scanf("%d %d %d", &a, &b, &c);
    if ((a <= b) && (a <= c))
        printf("Menor: %d\n", a);
    else
        if (b <= c)
            printf("Menor: %d\n", b);
        else
            printf("Menor: %d\n", c);
}
```

# Estrutura Condicional Aninhada

Dados três inteiros, faça um algoritmo que os coloque em ordem crescente.

Solução 1.

```
int main() {  
    int a, b, c;  
    printf("Digite tres numeros: ");  
    scanf("%d %d %d", &a, &b, &c);  
    if ((a <= b) && (a <= c)) { /* Menor numero: a */  
        if (b <= c)  
            printf("Ordem: %d, %d, %d\n", a, b, c);  
        else  
            printf("Ordem: %d, %d, %d\n", a, c, b);  
    } ...  
}
```

# Estrutura Condicional Aninhada

```
... else if (b <= c) { /* Menor numero: b */
    if (a <= c)
        printf("Ordem: %d, %d, %d\n", b, a, c);
    else
        printf("Ordem: %d, %d, %d\n", b, c, a);
} else { /* Menor numero: c */
    if (a <= b)
        printf("Ordem: %d, %d, %d\n", c, a, b);
    else
        printf("Ordem: %d, %d, %d\n", c, b, a);
}
}
```

Vejam os a solução 2

# Estrutura Condicional Aninhada

```
int main() { /* Solucao 2 */
    int a, b, c, aux;
    printf("Digite tres numeros: ");
    scanf("%d %d %d", &a, &b, &c);
    if (a>b || a>c) /* Condição para a não ser o menor */
        if (b<=c){ /* Troca a por b */
            aux =a;
            a =b;
            b =aux;
        }else{ /* Troca a por c */
            aux =a;
            a =c;
            c =aux;
        }
}
```

# Estrutura Condicional Aninhada

```
if (b>c) { /* Troca b por c */
    aux =b;
    b =c;
    c =aux;
}
printf("Ordem: %d, %d, %d\n", a, b, c);
}
```

# Estrutura Condicional

## *Estrutura if...else if...else...*

Múltiplas decisões mutuamente exclusivas

```
if (expressão) {  
    sentença;  
    ...  
} else if (expressão) {  
    sentença;  
    ...  
} else {  
    sentença;  
    ...  
}
```

# Estrutura Condicional

```
int main() {
    int idade;
    printf("Digite sua idade:");
    scanf("%d", &idade);
    if ( (idade >= 0) && (idade < 18) ) {
        printf("Nao possui habilitacao.\n");
    } else if ( idade < 65 ) {
        printf("Renove exames a cada 5 anos.\n");
    } else {
        printf("Renove exames a cada 3 anos.\n");
    }
}
```

# Estrutura Condicional

## *Estrutura switch*

O switch é uma construção de múltiplas possibilidades de decisão. Ele compara o resultado de uma expressão com uma série de valores constantes.

# Estrutura Condicional

## *Estrutura switch*

```
switch (expressão) {  
  case valor1:  
    sentenças;  
    ...  
  case valor2:  
    sentenças;  
    ...  
  case valor3:  
    sentenças;  
    ...  
  default:  
    sentenças;  
    ...  
}
```

# Estrutura Condicional

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

```
int main() {
```

```
float preco;
```

```
char categoria;
```

```
float preco_final;
```

```
printf("Digite o preco do ingresso: ");
```

```
scanf("%f", &preco);
```

```
printf("Selecione:\n");
```

```
printf("E - estudante,\nA - Aposentado,\nN - normal\n");
```

```
printf("Digite a categoria do cliente (E/A/N): ");
```

```
scanf(" %c", &categoria);
```

# Estrutura Condicional

```
switch (categoria) {
```

```
    case 'e': case 'E':
```

```
        preco_final = preco * 0.5;
```

```
        printf("Com desconto estudante: %f\n", preco_final);
```

```
        break;
```

```
    case 'a': case 'A':
```

```
        preco_final = preco * 0.7;
```

```
        printf("Com desconto aposentado: %f\n", preco_final);
```

```
        break;
```

```
    case 'n': case 'N':
```

```
        printf("Preço sem desconto: %f\n", preco);
```

```
        break;
```

```
    default:
```

```
        printf("Categoria invalida!\n");
```

```
        break;
```