Monitoria LPI - Módulo II

Comandos de Controle de Repetição

Os comandos de controle de repetição incluem *comandos de seleção*, (condicionais: **if** e **switch**) *comandos de interação*, (laços: **while**, **for** e **do-while**) comandos de *saltos* ou *desvios* (break, **continue** e **goto**) e *comandos de rótulos* (case e **default**). Discutiremos neste módulo os comandos condicionais **if, switch** e o **Operador Ternário** (**?**).

1. Comandos de seleção (condicional):

Verdadeiro e Falso em C

Antes de discutimos os comandos de seleção, devemos ter um entendimento dos testes condicionais que são feitos em C determinando seu curso. Uma expressão condicional tem valor verdadeiro ou falso, em C diferente de outras linguagens, verdadeiro é qualquer valor diferente de 0 incluindo valores negativos. Um valor falso é 0. Utiliza-se para isso os *operadores relacionais* e os *operadores lógicos.*

Operadores Relacionais

Os operadores relacionais em C são os seguintes:

Operador	Significado
>	Maior do que.
<	Menor do que.
>=	Maior do que ou igual a.
<=	Menor do que ou igual a .
==	Igual a.
!=	Diferente de.

Exemplo 1.0: Considere as seguintes variáveis:

```
...
int a=3;
float x=1.5;
…
```

Condição	Valor lógico
(a! = x)	Verdadeiro
(a/2.0 == x)	Verdadeiro
(a/2 == x)	Falso
(a != 2*x)	Falso
(a >= x)	Verdadeiro
(a/3 <= x)	Verdadeiro
(a/x < 2)	Falso
(a)	Verdadeiro
(a – 2 * x)	Falso

Operadores Lógicos

Os operadores lógicos permitem combinar condições em uma única expressão lógica.

Os operadores lógicos em C são:

Operador	Significado
&&	Conjunção Lógica (“and”)
||	Disjunção Lógica (“or”)
!	Negação Lógica (“not”)

Dica:

“... Como esse módulo foi escrito para uma iniciação em C, não detalharemos os operadores lógicos. Mas devem ser estudados a fundo porque vocês irão utilizar esses conceitos em toda sua vida de programador e devem estar bem entendidos...”.

Considere as variáveis o exemplo 1.0.

Expressão	Valor Lógico
((a/2 == x) && (a>2))	Falso
((a != x) || (a/x < 2))	Verdadeiro
((x <= a) && (a >=2*x))	Verdadeiro
(!(a/3 <= x))	Falso
((a/2 == x) || (a >= x) && ! (2*x != a))	Verdadeiro
(a && x)	Verdadeiro
((a – 2 * x) || (x < a/2))	Falso

I. O Operador Condicional *IF*

A forma geral da sentença é:

```
If(expressão)
{
        comando;
}
else
{
        Comando;
}
```

Onde *comando* pode ser um bloco de comandos, onde se faz necessário as chaves (“{ }“) ou ser for só um comendo a linguagem permite que seja colocada em uma linha. Vejamos um exemplo.

Exemplo 2.0:

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

main()
{
      int a = 3;
      float x = 1.5;

      if(a > b)
      {
            printf(“%d e’ maior que %f”,a,x);
      }
      else
            printf(“%f” e’ maior que %d”,x,a);
system(“pause”);
}
```

Se a expressão é verdadeira(algo diferente de 0), o comando é executado; caso contrário é executado o bloco do **else** (caso exista).

O comando condicional controlado por **if** deve produzir um resultado escalar. Um *escalar* é um inteiro, um caractere ou um tipo de ponto flutuante (que normalmente é evitado pela demora na sua execução).

II. If’s Aninhados

Um **if** aninhado é um comando **if** que é objeto de outro **if** ou **else.** Em C, um comando **else** se refere sempre ao **if** mais próximo, que está dentro do mesmo bloco e não estar associado a outro **if.** Vejamos um exemplo:

Exemplo 3.0:

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

main()
{
    int a=3;
    float x=1.5;

    if(a && x)
    {
        if(!(a / 3 <= x)) //esse if não tem um else
            comando1;
        if ((a/2 == x) && (a>2))
            comando2;
        else
            comando3;
    }
    else
        comando4;
}
```

** Essa chave indica a relação do **if** com o seu **else** **( )**

Exemplo usando if aninhados.

Exemplo3.1

```
/* Programa de números mágicos*/
#include <stdio.h>
#include <stdlib.h>

main()
{
int magic; // número mágico
int palpite; // palpite do usuário

magic=rand(); /* “rand();” -> gera um numero aleatório, estudar essa função!! */

printf(“Adivinhe o numero magico: ”);
scanf(“%d”,&palpite);

if(palpite==magic)
{
     printf(“*** Certo ***”);
     printf(“ %d e’ o numero  mágico\n”,magic);
}
else
{
printf(“Errado!!!!”);
  if(palpite > magic)
        printf(“Muito alto!! \n”);
  else
        printf(“Muito baixo!! \n”);
}
system(“pause”);
}
```

III. Escala de if-else-if

É uma construção comum em C, algumas vezes chamada de *escala if-else-if.* A sua forma geral é:

```
If(expressão) comando;
 else
   if(expressão) comando;
     else
```

if(expressão) *comando*;
.
.
.
else *comando;*

As condições são validas de cima para baixo. Sendo a expressão verdadeira o comando é executado e é desviado do resto da escala. Se a primeira for falsa, no último **else** é executado seu comando.

Exemplo 3.2

```
/* Programa de números mágicos*/
#include <stdio.h>
#include <stdlib.h>

main()
{
int magic; // numero mágico
int palpite; // palpite do usuário

magic=rand(); /* “rand();” -> gera um numero aleatório, estudar essa função!! */

printf(“Adivinhe o numero magico: ”);
scanf(“%d”,&palpite);

if(palpite==magic)
{
     printf(“*** Certo ***”);
     printf(“ %d e’ o numero  mágico\n”,magic);
}
else if(palpite > magic)
     printf(“Errado, muito alto!! \n”);
else
     printf(“Errado, muito baixo!! \n”);
}
system(“pause”);
}
```

O COMANDO SWITCH

O comando switch evita o uso excessivo de if's em certas aplicações, tornando o código mais limpo e mais fácil de entender. O valor da variável é testado de forma sucessiva, "caso" ele encontre uma coincidência o comando que corresponde a este caso é executado.

O switch é análogo ao if-else-if. Com uma diferença o switch não aceita expressões. Somente aceita constante.

Se nenhuma coincidência for encontrado é executado o "caso" default (ausência), vale lembrar que o uso do default é optativo, mas ele é fundamental para cobrir os demais casos.

O break ele pára a execução do switch, este comando não é obrigatório no switch, mas se não for declarado o switch continua sendo executado.

Sintaxe:

```
switch(variável){
        case constante_1:
                comandos;
                break;
        case constante_2:
                comandos;
                break;
        .
        .
        .
        case constante_n:
                comandos;
                break;
        default:
                comandos;
                break;
    }
```

Exemplo de implementação 1.

```
#include <stdio.h>
int main (){
    int num;
    printf ("Digite o numero: ");
    scanf ("%d",&num);
    switch (num)
```

```
 {
        case 1:
                printf ("\n\nO numero e igual a 1.\n");
                break;
        case 2:
                printf ("\n\nO numero e igual a 2.\n");
                break;
        case 3:
                printf ("\n\nO numero e igual a 3.\n");
                break;
        default:
                printf ("\n\nO numero nao eh 1 nem 2 nem 3.\n");
                break;
      }
        system("pause");
}
```

Exemplo de implementação 2.

```
#include <stdio.h>
int main (){
        int num;
        printf ("Digite o numero: ");
        scanf ("%d",&num);
        switch (num)
      {
        case 1:
         case 2:
         case 3:
                printf ("\n\nO numero eh 1 ou 2 ou 3.\n");
                break;
        default:
                printf ("\n\nO numero nao eh 1 nem 2 nem 3.\n");
                break;
      }
        system("pause");
}
```

Operador ternário

Sintaxe: Condição ? expressão1 : expressão2 ;

Esta operação pode ser interpretada da seguinte forma: se a Condição for verdadeira, executa a expressao1; caso contrario, executa a expressao2.

Comparação com o if – else:

```
If(Condição)
    expressão 1;
else
    expressão 2;
```

Ex (usando if-else):

```
//Programa pronto para ser compilado e executado no Dev C++
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
main()
{
int n;
printf("Numero: "); scanf("%d",&n);
if(n==10)
printf("numero eh = %d\n",n);
else
printf("\nO numero nao eh 10 e sim %d\n",n);
system("pause");
}
// -------------------------------------------------------------------
```

Usando o operador ternário teríamos:

```
//Programa pronto para ser compilado e executado no Dev C++
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
main()
{
int n;
printf("Numero: "); scanf("%d",&n);
(n==10)?printf("numero eh = %d\n",n):printf("\nO numero nao eh 10 e
sim %d\n",n);
system("pause"); }
```