Engenharia da Computação

Disciplina: DCExt Programação Imperativa

Aula 05: Estruturas Condicionais e de Repetição

Prof. Dr. Hemir da C. Santiago
hcs2@poli.br

Parte 1:

- Uso de {..} (delimitadores)
- Instrução *if..else* revisitada
- Instrução *switch..case*
- Operador condicional (?)
- Exercícios

Em programas usando *if* e *else*, delimita-se as instruções que devem ser executadas quando a condição *if* ou *else* é satisfeita. Considere o seguinte trecho de código e suponha o valor da variável ***a*** seja 4:

```
if(a > 0) {
    printf("%d\n" , a);
}
```

A instrução ***printf*** é executada, aparecendo o valor 4 na saída (tela). Agora, se as chaves são removidas, o comportamento e resultado permanecem inalterados, isto é:

```
if(a > 0)
    printf("%d\n" , a);
```

Uso de {..} (delimitadores)

Exemplo: Suponha que *x* e *y* sejam variáveis que representam as coordenadas dos eixos *x* e *y,* respectivamente. Assuma ainda que ambas as variáveis armazenam o valor 0 (zero), indicando as coordenadas ou posição (0, 0). O programa, inicialmente, deve exibir posição inicial e, em seguida, solicitar ao usuário que digite um caractere *n*, *s*, *l* ou *o* indicando direção norte, sul, leste ou oeste, respectivamente. Após o programa ler o caractere, digitado pelo usuário, ele atualiza o valor da coordenada x ou y e mostra a nova posição.

Instrução *if..else* revisitada

```
#include <stdio.h>

int main()
{
   int x = 0, y = 0;
   char c;

   //Mostra posição inicial
   printf("Posicao inicial = (%d, %d)\n", x, y);
   // solicita a direção de movimento
   printf("Digite n para norte\n");
   printf("Digite s para sul\n");
   printf("Digite l para leste\n");
   printf("Digite o para oeste\n");
   printf("Direcao = ");
   c = getchar(); // o caractere lido pela função é atribuído à variável c
```

Instrução *if..else* revisitada

```
 //Descobre direção e atualiza coordenadas
   if (c == 'n') //se norte
      y++; //atualiza y
   else if (c == 's') // se sul
      y--; //atualiza y
   else if (c == 'l') // se leste
      x++; //atualiza x
   else if (c == 'o') // se oeste
      x--; //atualiza x
   else
      printf("Caractere errado!\n");

   // Mostra novas coordenadas
   printf("Posicao Atual  = (%d, %d)\n", x, y);
}
```

Instrução *if..else* revisitada

A instrução ***switch..case*** é outra instrução de seleção, similarmente a *if..else*, onde a execução de uma ou mais instruções, como ocorre com o *if*, depende de ser verdadeira a condição avaliada. Sintaxe:

```
switch (opcao) {
   case valor_1: instrução_i;
                 break;
   case valor_2: instrução_ii;
                 break;
   case valor_3: instrução_iii;
                 break;
   ...
   default: instrução_final;
}
```

Instrução *switch..case*

```
#include <stdio.h>

int main()
{
   int x = 0, y = 0;
   char c;
   //Mostra posição inicial
   printf("Posicao inicial = (%d, %d)\n", x, y);
   // solicita a direção de movimento
   printf("Digite N(orte), S(ul), L(este) ou O(este):\n");
   printf("Direcao = ");
   c = getchar(); // o caractere lido pela função é atribuído à variável c

   //Descobre direção e atualiza coordenadas
   switch(c) { //c, contém valor a ser avaliado
      case 'N': //se norte
         y++; //atualiza y
         break;
```

Instrução *switch..case*

```
        case 'S': //se sul
            y--; //atualiza y
            break;
        case 'L': //se leste
            x++; //atualiza x
            break;
        case 'O': //se oeste
            x--; //atualiza x
            break;
        default: // default é executada apenas se todos case's falham
            printf("Caractere errado!\n");
    } // fim do switch ()

    // Mostra novas coordenadas
    printf("Posicao Atual  = (%d, %d)\n", x, y);
}
```

Instrução *switch..case*

Exemplo: Sejam *x* e *y* dois valores do tipo inteiro, digitados pelo usuário, deseja-se descobrir o maior dos dois números:

```
#include <stdio.h>

int main()
{
   int x, y, maior;
   printf("Digite x = ");
   scanf("%d", &x);
   printf("Digite y = ");
   scanf("%d", &y);
   if (x > y){
      maior = x;
   }
   else {
      maior = y;
   }
   printf("valor maior = %d", maior);
}
```

Operador condicional (?)

Sintaxe do operador condicional:

(expressao1) ? expressao2: expressao3;

O operador condicional (***?***) é um ***operador ternário***, ele possui três operandos, i.e., *expressao1*, *expressao2* e *expressao3*.

expressao1 é uma condição que se avaliada verdadeira, então a *expressao2* é executada. Caso contrário, a *expressao3* é executada.

Operador condicional (?)

Exemplo: Sejam *x* e *y* dois valores do tipo inteiro, digitados pelo usuário, deseja-se descobrir o maior dos dois números:

```
#include <stdio.h>

int main()
{
   int x, y, maior;
   printf("Digite x = ");
   scanf("%d", &x);
   printf("Digite y = ");
   scanf("%d", &y);

   maior = (x > y) ? x : y;

   printf("valor maior = %d", maior);
}
```

Operador condicional (?)

1) Elabore um programa que lê dois valores inteiros *x* e *y*. O programa deverá descobrir e exibir a diferença (valor absoluto) entre o maior e o menor.

2) Elabore um programa que lê as notas de 1º e 2º exercício escolar de um estudante. O programa deve descobrir se o estudante foi aprovado, está na final ou foi reprovado, considerando que:

Se média >= 7.0, estudante aprovado

Se 7.0 > média >= 3.0, exame final

Se média < 3.0, aluno reprovado

Exercícios

3) Elabore um programa que solicita ao usuário digitar um número inteiro, verificando se o número digitado é múltiplo de 3. Se o número é múltiplo de 3, o programa deve exibi-lo. Caso contrário, o programa deve determinar o próximo número múltiplo de 3 e o exibir.

4) Elabore um programa que solicita ao usuário digitar quatro números do tipo *float* e descobre e exibe o maior deles.

Exercícios

Parte 2:

- Controle por repetição
- Instrução *while*
- Instrução *do..while*
- Instrução *for*
- Exercícios

No ***controle por repetição*** uma expressão (condição) define quantas vezes a execução de uma instrução (ou bloco contendo instruções) será repetido.

Quando um trecho de um código é executado repetidamente, diz-se que o ***programa ou código está executando um laço*** ou em *loop*.

Se a condição (de controle) do fluxo de execução por repetição permanece verdadeira sempre, o programa fica em "*loop* infinito".

Controle por repetição

A instrução ***while*** permite que o programador especifique situações em que uma instrução (ou bloco de instruções) deve ser ***executado repetidamente enquanto uma condição é verdadeira*:**

```
while(condição) {
    instrução_1;
    instrução_2;
    ...
    instrução_n;
}
```

Instrução *while*

Exemplo: Elaborar um programa que inicializa uma variável *n* do tipo inteiro com o valor 0 (zero). Em seguida, o programa deve exibir o valor inicial da variável *n* e incrementá-la ***enquanto*** seu valor é menor ou igual a 10.

```
#include <stdio.h>

int main()
{
   int n = 0;
   while (n <= 10){ // início do while
      printf("%d ", n);
      n++;
   } //fim do while
}
```

Instrução *while*

Exemplo: Modificar o programa anterior de modo que a variável *n* do tipo inteiro seja inicializada com o valor 10 (dez). Em seguida, o programa exibe o valor inicial da variável *n* e decrementá-la ***enquanto*** seu valor é maior ou igual a 0.

```
#include <stdio.h>

int main()
{
   int n = 10;
   while (n >= 0){
      printf("%d ", n);
      n--;
   } //fim do while
}
```

Instrução *while*

Exemplo: Solicite ao usuário que digite um valor inteiro (variável *n*, pode ser positivo, negativo ou zero). Em seguida, o programa deve exibir o valor inicial de *n* e mostrar os 10 valores sucessivos.

```
#include <stdio.h>

int main()
{
   int n, c = 1;
   printf("Digite o valor de n: ");
   scanf("%d", &n);
   printf("\n%d ", n); //exibe 1º valor de n

   while (c <= 10){
      n++; //obtém o sucessor
      printf("%d ", n);
      c++; // incrementa contador
   } //fim do while
}
```

Instrução *while*

Exercício: Elaborar programas para obter os seguintes somatórios (considere *n* = *10)*:

(a) $\sum_{i=1}^{n} i$

(b) $\sum_{i=1}^{n} 2i$

(c) $\sum_{i=1}^{n} i^2$

(d) $\sum_{i=1}^{n} i^3$

Instrução *while*

A instrução ***do..while*** permite a execução inicial do corpo de instruções e, após isso apenas, a condição é testada. Se ainda verdadeira, o corpo de instruções é executado novamente. Esse processo continua até que a condição se torne falsa.

Sintaxe:

```
do {
   instrução_1;
   instrução_2;
   ...
   instrução_n;
} while(condição);
```

Instrução *do-while*

Exemplo: Elaborar um programa que inicializa uma variável *n* do tipo inteiro com o valor 0 (zero). Em seguida, o programa exibe o valor inicial da variável *n,* a incrementa e mostra na saída (tela) os dez valores seguintes.

```
#include <stdio.h>

int main()
{
   int n = 0;
   do { // início do do..while
      printf("%d ", n);
      n++;
   } while (n <= 10); // fim do do..while
}
```

Instrução *do-while*

A instrução ***for*** requer a especificação de 3 elementos:

- Definição da ***variável de controle*** do laço (ou *loop*) e respectivo valor inicial.
- Descrição da ***condição*** de controle do *loop*.
- Definição da expressão de ***incremento*** (ou decremento), usado no *loop*.

Sintaxe:

```
for(variávelDeControle; condição; expressãoDeIncremento) {
   instrução_1;
   instrução_2;
   ...
   instrução_n;
}
```

Instrução *for*

Exemplo: Elaborar um programa que inicializa uma variável *n* do tipo inteiro com o valor 0 (zero). Em seguida, o programa exibe o valor inicial da variável *n,* a incrementa e mostra na saída (tela) os dez valores seguintes.

```
#include <stdio.h>

int main()
{
   int n;

   for (n = 0; n <= 10; n++) { // início do for
      printf("%d ", n);
   } // fim do for
}
```

Instrução *for*

1) Elabore um programa que solicite ao usuário digitar um número inteiro positivo *x* e, em seguida, seu programa gera a tabuada matemática de multiplicação para o número *x* digitado. Assim, por exemplo, se o usuário digitar o número 7, a saída a ser exibida deve ser como mostrado a seguir:

7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
7 x 4 = 28
7 x 5 = 35
7 x 6 = 42
7 x 7 = 49
7 x 8 = 56
7 x 9 = 63
7 x 10 = 70

Exercícios

2) Elabore um programa que calcule e possa exibir na saída (tela) o valor de *S*, onde *S = 1/1 + 3/2 + 5/3 + 7/4 + ... + 99/50*

3) Elaborar um programa para calcular o somatório abaixo. Note que este é o somatório de todos valores x_i no intervalo de *m* até *n*, sendo os valores x_i todos os valores de *m* a *n*. Para fins de exercício, suponha m = 10 e n = 10*m.

$$\sum_{i=m}^{n} x_i$$

Exercícios

4) Elaborar um programa que solicita ao usuário digitar dois números inteiros *x* e *y*, e exiba como saída todos os números inteiros entre *x* e *y* e os respectivos valores elevados ao quadrado, exemplo para *x* = 1 e *y* = 6:

1 1

2 4

3 9

4 16

5 25

6 36

Exercícios

5) Elaborar um programa que solicita ao usuário digitar um número inteiro *n*. Contudo, *0 ≤ n ≤ 9*. Desse modo, se o usuário digitar 4, então o programa deve exibir:

```
4 4 4 4
4 4 4 4
4 4 4 4
4 4 4 4
```

Ou, se o usuário digitar 2, então o programa deve exibir:

```
2 2
2 2
```

Exercícios

	DATA	AULA
1	22/08/2024	Apresentação da disciplina | Introdução à Programação Imperativa
2	29/08/2024	Introdução à Linguagem de Programação C
3	05/09/2024	Conceitos Fundamentais
4	12/09/2024	Tipos de Dados Especiais em C
5	19/09/2024	Estruturas Condicionais e de Repetição
6	26/09/2024	Pré-processamento
7	03/10/2024	Registros/Estruturas de Dados
8	10/10/2024	Ponteiros
9	17/10/2024	1º Exercício Escolar

Plano de Aulas

POLI ESCOLA POLITÉCNICA DE PERNAMBUCO
UPE UNIVERSIDADE DE PERNAMBUCO

	DATA	AULA
10	24/10/2024	Arquivos
11	31/10/2024	Acompanhamento de projetos
12	07/11/2024	Acompanhamento de projetos
13	14/11/2024	Acompanhamento de projetos
14	21/11/2024	Acompanhamento de projetos
15	28/11/2024	Apresentação parcial
16	05/12/2024	Apresentação de projetos
17	12/12/2024	Avaliação Final

Plano de Aulas