Linguagem C:
Estruturas de Controle

Sumário

- Estrutura de Controle e de Fluxo
- Comandos de Seleção:
  - O comando if;
  - Ifs Aninhados;
  - A escada if-else-if;
  - A expressão condicional;
  - Switch;
  - Comandos switch aninhados;
- Comandos de Interação:
  - O laço for;
  - O laço while;
  - O laço do-while.

Estrutura de Controle e de Fluxo

- As estruturas de controle são fundamentais para qualquer linguagem de programação;
- Sem elas só haveria uma maneira do programa ser executado: de cima para baixo, comando por comando;
- O padrão ANSI divide os comandos de C nestes grupos:
  - Seleção;
  - Interação;
  - Desvio;
  - Rótulo;
  - Expressão;
  - Bloco.

Estrutura de Controle e de Fluxo

- Muitos comando em C contam com um teste condicional que determina o curso da ação;
- Uma expressão condicional chega a um valor verdadeiro ou falso.
- Em C, um valor verdadeiro é qualquer valor diferente de zero, incluindo números negativos;
- O valor falso é 0;

Comandos de Seleção

- C suporta dois tipos de comandos de seleção: **if** e **switch**. Além disso o operador ? É uma alternativa ao if em certas circunstâncias.

- A forma geral da sentença if é:

```
if(expressão)
        { comando; }
else
        { comando; }
```

  - Onde comando pode ser um único comando, um bloco de comandos ou nada (comandos vazios);
  - A cláusula else é opcional.

O comando if

- A expressão é avaliada. Se ela for verdadeira (diferente de 0), o comando ou bloco que forma o corpo do if é executado;
- Caso contrário, o comando ou bloco que é o corpo do else (se existir) é executado.
- Lembre-se: Apenas o código associado ao **if** ou o código associado ao **else** será executado, nunca ambos;

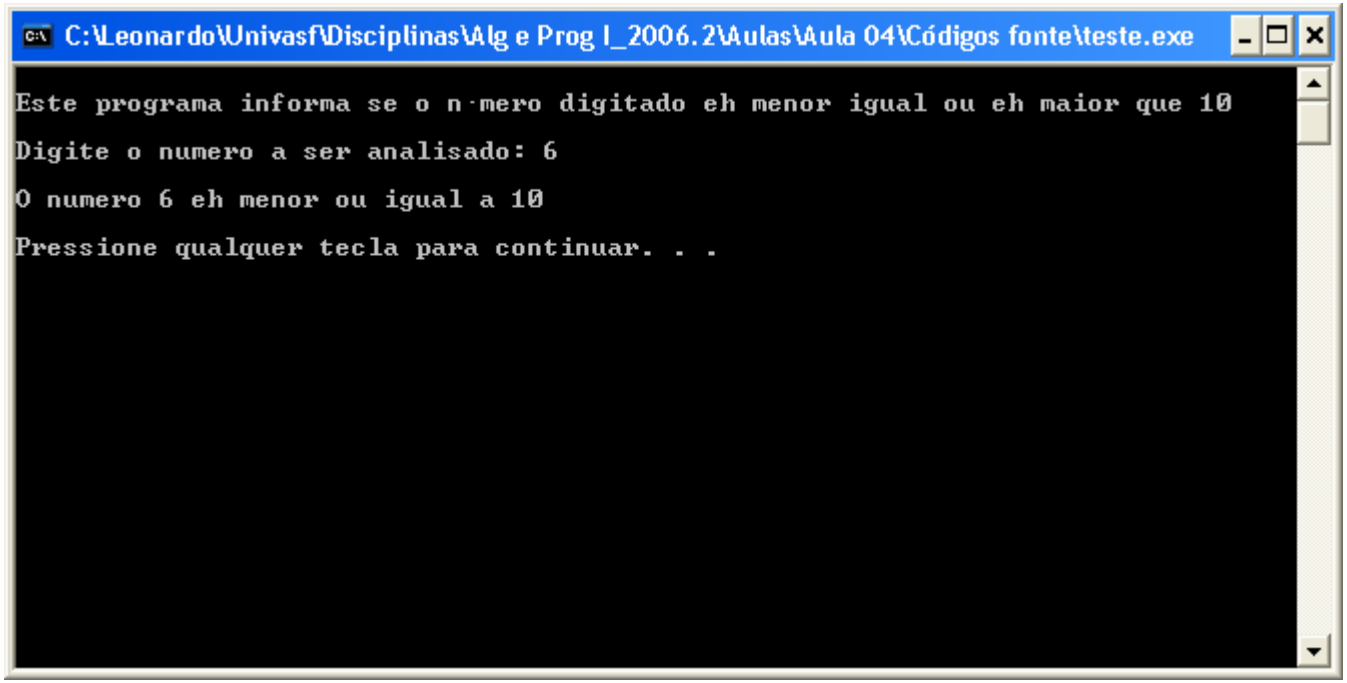
O comando if

- Aqui representamos um exemplo de uso do comando **if**, veja:

```
#include <stdio.h>
/* Este programa demonstra a utilização do comando de seleção IF */

int main()
{
      int num;
      printf("\nEste programa informa se o número digitado eh menor igual ou eh maior que 10"
      printf("\n\nDigite o numero a ser analisado: ");
      scanf("%d", &num);
      if(num>10)
                printf("\nO numero %d eh maior que 10", num);
      else
                printf("\nO numero %d eh menor ou igual a 10\n\n", num);
      system("pause");
      return 0;
}
```

O comando if

- A saída no console para o programa anterior será:



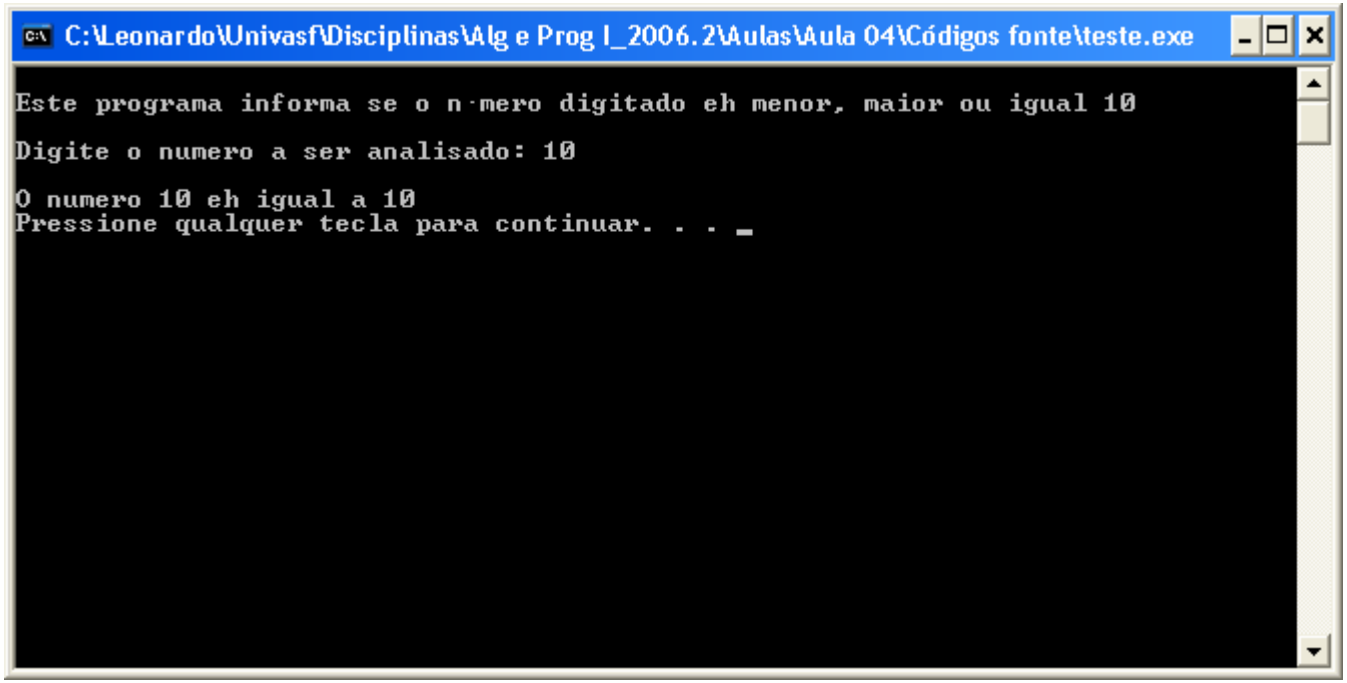
O comando if aninhado

- Pode-se também usar unicamento o comando if (sem else), veja:

```
#include <stdio.h>
/* Este programa demonstra a utilização do comando de seleção IF */

int main()
{
      int num;
      printf("\nEste programa informa se o número digitado eh menor, maior ou igual 10");
      printf("\n\nDigite o numero a ser analisado: ");
      scanf("%d", &num);
      if(num>10)
                printf("\nO numero %d eh maior que 10\n", num);
      if(num<10)
                printf("\nO numero %d eh menor que 10\n", num);
      if(num==10)
                printf("\nO numero %d eh igual a 10\n", num);
      system("pause");
      return 0;
}
```

O comando if aninhado

- A saída no console para o programa anterior será:



O comando if-else-if

- A estrutura **if-else-if** é apenas uma extensão da estrutura **if-else**;
- A sua forma geral é:

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

- As condições são avaliadas de cima para baixo.

O comando if-else-if

- Embora seja tecnicamente correta, o recuo da escada if-else-if anterior pode ser excessivamente profundo.
- Por essa razão, a escada if-else-if é geralmente recuada deste forma:

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

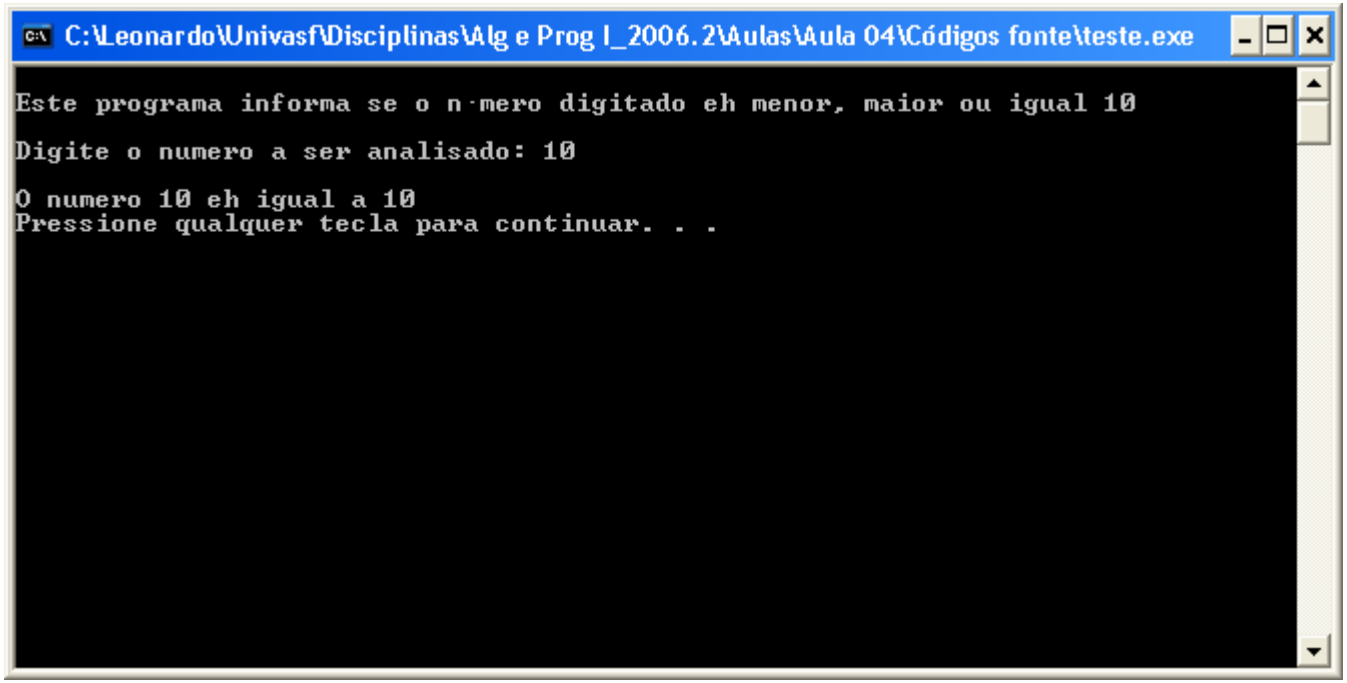
O comando if-else-if

- Vejamos um exemplo da escada if-else-if:

```
#include <stdio.h>
/* Este programa demonstra a utilização do comando de seleção IF */

int main()
{
      int num;
      printf("\nEste programa informa se o número digitado eh menor, maior ou igual 10");
      printf("\n\nDigite o numero a ser analisado: ");
      scanf("%d", &num);
      if(num>10)
                printf("\nO numero %d eh maior que 10\n", num);
      else
          if(num<10)
                printf("\nO numero %d eh menor que 10\n", num);
          else
                printf("\nO numero %d eh igual a 10\n", num);
      system("pause");
      return 0;
}
```

O comando if-else-if

- A saída no console para o programa anterior será:



A expressão condicional

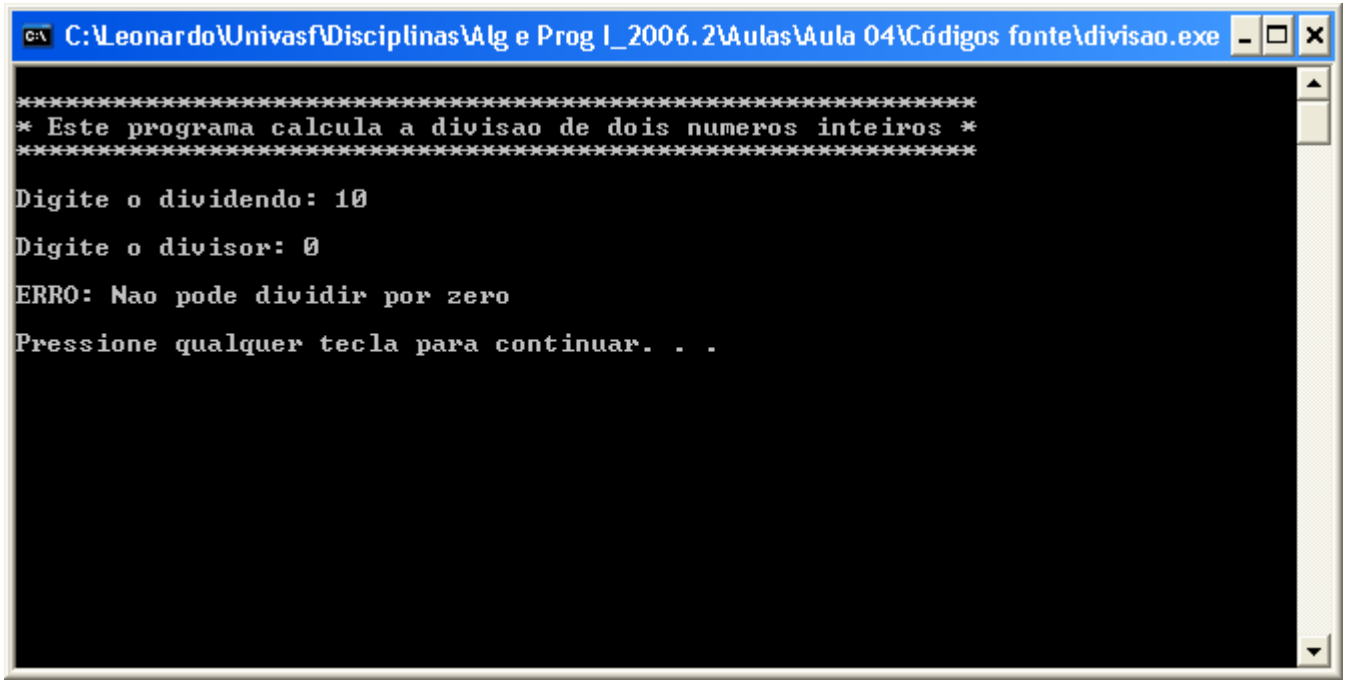
- Em C, o controlador do **if** precisa, apenas, ser zero ou não-zero, vejamos:

```
#include <stdio.h>
/* Este programa demostra uma expressão condicional no comando if */

int main()
{
    int num1, num2;
    printf("\n*************************************************************");
    printf("\n* Este programa calcula a divisao de dois numeros inteiros *");
    printf("\n*************************************************************");
    printf("\n\nDigite o dividendo: ");
    scanf("%d", &num1);
    printf("\nDigite o divisor: ");
    scanf("%d", &num2);
    if (num2)
        printf("\n%d dividido por %d eh: %d\n\n", num1, num2, num1/num2);
    else
        printf("\nERRO: Nao pode dividir por zero\n\n");
    system("pause");
    return 0;
}
```

if(num2!=0)

A expressão condicional

- A saída no console para o programa anterior será:



O comando switch

- A linguagem C tem um comando interno de seleção múltipla chamado switch;

- O switch testa sucessivamente o valor de uma expressão contra uma lista de constantes inteiras ou de caractere;

```
int num;
printf("Digite um numero: ");
scanf("%d", &num);
switch (num)
{
        case 9:
             printf("\nO numero e igual a 9\n");
             break;
        case 10:
             printf("\nO numero e igual a 10\n");
             break;
        default:
             printf("\nO numero nao eh nem 9 nem 10\n");
}
system("pause");
return 0;
```

O comando switch

- A forma geral do comando switch é:

```
switch(expressão){
case constante1:
    seqüência de comandos
    break;
case constante2:
    seqüência de comandos
    break;
case constante3:
    seqüência de comandos
    break;
...
default:
    seqüência de comandos
}
```

- O valor da expressão é testado, na ordem, contra os valores das constantes especificadas nos comandos case;
- Quando uma coincidência for encontrada, a seqüência de comando associada àquele case será executada até que o comando break ou o fim do comando switch seja alcançado;
- O comando default é executado se se nenhuma coincidência for detectada;
- O default é opcional.

O comando switch

- Exemplo de utilização do switch:

```
#include <stdio.h>
#include <conio.h>
/* Este programa exemplifica o uso do comando switch */
int main()
{
    float op1, op2;
    char operacao;
    printf("\n*************************************************************");
    printf("\n* Este programa executa as quatro operacoes basicas aritmeticas *");
    printf("\n*************************************************************");
    printf("\n\nDigite o primeiro operando: ");
    scanf("%f", &op1);
    printf("\nDigite o segundo operando: ");
    scanf("%f", &op2);
    printf("\nDigite o operador: ");
    operacao = getche();
```

- Continua no próximo slide

getche() função usada para leitura de caracteres, um por vez, da entrada padrão.

Comum apenas para DOS

O comando switch

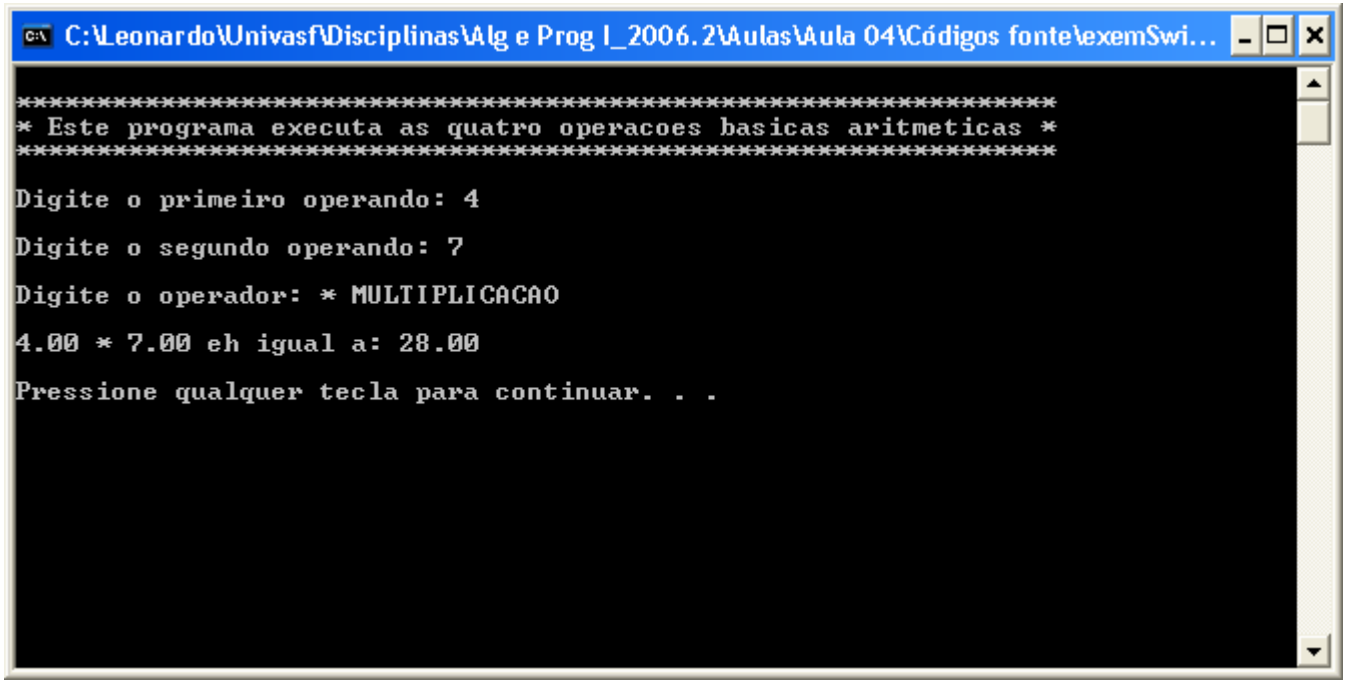
- Exemplo de utilização do switch(Continuação):

```
switch (operacao)
{
        case '+':
             printf(" SOMA\n\n%5.1f + %5.1f eh igual a: %5.1f\n\n", op1, op2, op1+op2);
             break;
        case '-':
             printf(" SUBTRACAO\n\n%6.1f - %6.1f eh igual a: %4.1f\n\n", op1, op2, op1-op2);
             break;
        case '*':
             printf(" MULTIPLICACAO\n\n%.2f * %.2f eh igual a: %.2f\n\n", op1, op2, op1*op2);
             break;
        case '/':
             printf(" DIVISAO\n\n%.5f / %.5f eh igual a: %.5f\n\n", op1, op2, op1/op2);
             break;
        default:
             printf(" %c\n\nOperacao desconhecida\n\n", operacao);
}
system("pause");
return 0;
}
```

Define quantas casas decimais serão impressas

O comando switch

- A saída no console para o programa anterior será:



O comando switch

- O padrão ANSI C especifica que um switch pode ter pelo menos 257 comandos case;
- Embora **case** seja um rótulo ele não pode existir sozinho, fora de um switch;
- O comando **break** é um dos comandos de desvio em C. Pode usá-lo em laços tal como no comando switch;
  - Quando um break é encontrado em um switch, a execução do programa "salta" para a linha de código seguinte ao comando switch;

O comando switch

- Se o comando **break** for omitido, a execução do programa continua pelos próximos comandos case até que um **break**, ou o fim do **switch** seja encontrado;

Ex:

```
int x=0, op; scanf("%d", op);
switch(op) {
        case 1: /*Nada*/
        case 2:
                x= 10;
                break;
        case 3:
                x++;
        case 4:
                x++;
        default:
                x--;
}
```

Opção	x
1	10
2	10
3	1
4	0
default	-1

O comando switch

- Há três observações importantes a saber sobre o comando switch:
  - switch só pode testar igualdade, enquanto que o if pode avaliar uma expressão lógica e/ou relacional;
  - Duas constantes case no mesmo switch não podem ter valores idênticos;
  - Se constantes de caractere são usadas em um comando switch, elas são automaticamente convertidas para seus valores inteiros;

O comando switch

- Os comandos associados a cada case não são blocos de códigos mas, sim, seqüência de comandos. Vejamos em que essa distinção técnica influencia:

```
/* Incorreto */
switch(op) {
   case 1:
        int i;
        ...
```

```
/* Correto */
switch(op) {
   case 1:
   {
        int i;
        ...
   }
```

Comandos switch aninhados

- Podemos ter um **switch** como parte de uma seqüência de comandos de outro **switch**, vejamos:

```
switch(x) {
        case 1:
                switch(y) {
                   case 0:      printf("\n10 em decimal eh 2");
                                break;
                   case 1: ...
                }
        case 2:
        ...
}
```

Não ocorre conflito com o case mais externo

Comandos de Interação

- Na linguagem C, comando de interação (também chamados laços) permitem que um conjunto de instruções seja executado até que ocorra uma certa condição;

- As estruturas de repetição em C apresentam-se em 3 formas distintas:
  - for
  - while
  - do-while

O laço for

- O laço for é a instrução mais poderosa na criação de estruturas de repetição;
- Vejamos sua forma geral mais comum:

```
for(inicialização; condição; incremento)
{
        comandos;
}
```

  - *Inicialização* é, geralmente, um comando de atribuição que é usado para colocar um valor na variável de controle do laço;
  - A *condição* é uma expressão relacional que determina quando o laço acaba;
  - O *incremento* define como a variável de controle do laço varia cada vez que o laço é repetido;

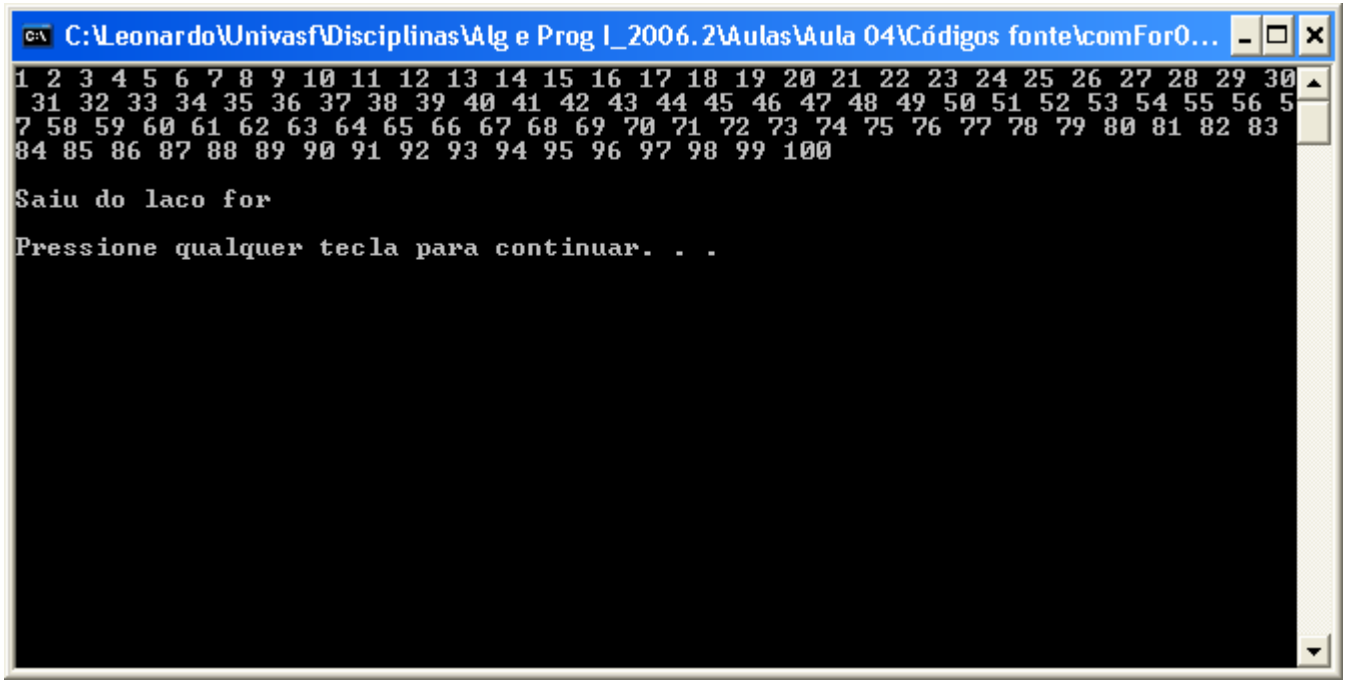
O laço for

- As seções no comando for são separadas por pontos-e-vírgulas;
- Uma vez que a condição se torne falsa, a execução do programa continua no comando seguinte ao for, vejamos:

```c
#include <stdio.h>
/* Este programa exemplifica a utilização do comando for */

int main()
{
    int x;
    for(x=1; x<=100; x++)
    {
            printf("%d ", x);
    }
    printf("\n\nSaiu do laco for\n\n");
    system("pause");
    return 0;
}
```

O laço for

- A saída no console para o programa anterior será:



O laço for

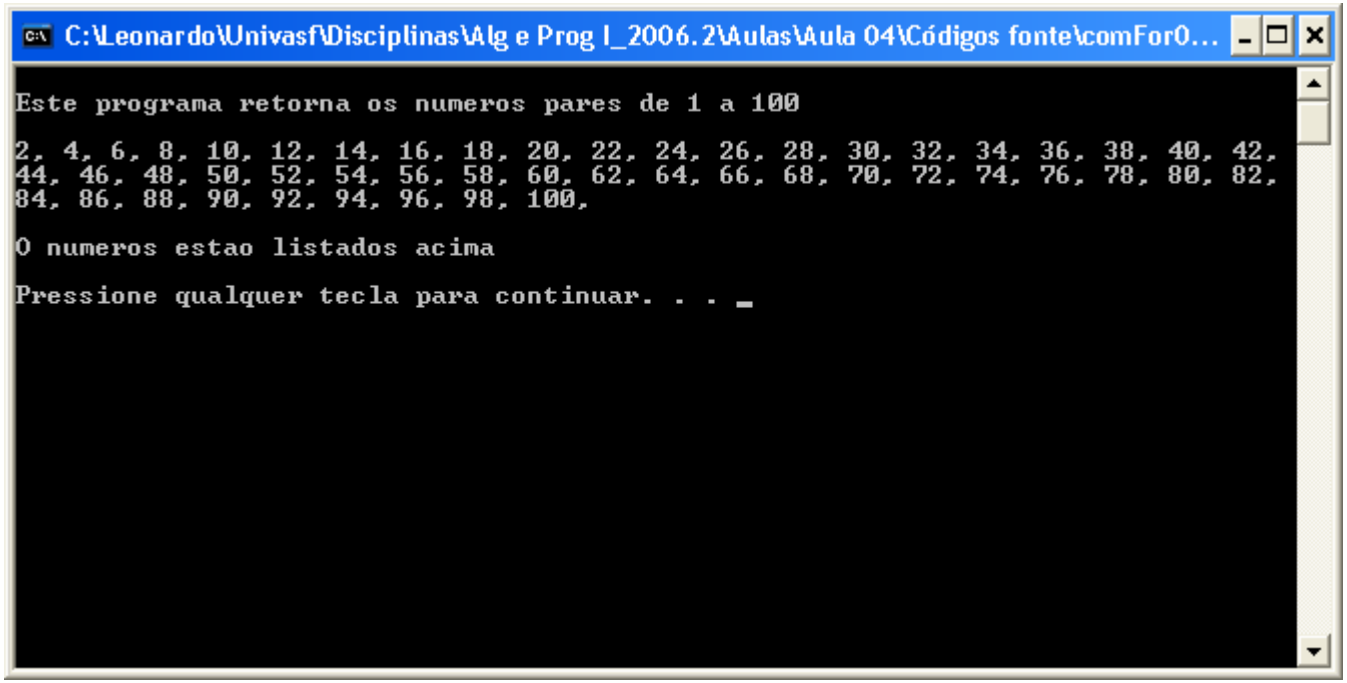
- Podemos utilizar outras estruturas de controle dentro do laço for, vejamos:

```
#include <stdio.h>
/* Este programa exemplifica a utilização do comando for com if */

int main()
{
    int i;
    printf("\nEste programa retorna os numeros pares de 1 a 100\n\n");
    for(i=1; i<=100; i++)
    {
            if(i%2==0)
            {
                    printf("%d, ", i);
            }
    }
    printf("\n\nO numeros estao listados acima\n\n");
    system("pause");
    return 0;
}
```

O comando if está selecionando apenas os números ímpares

O laço for

- A saída no console para o programa anterior será:



O laço for

- O mesmo programa anterior poderia ser escrito da seguinte forma:

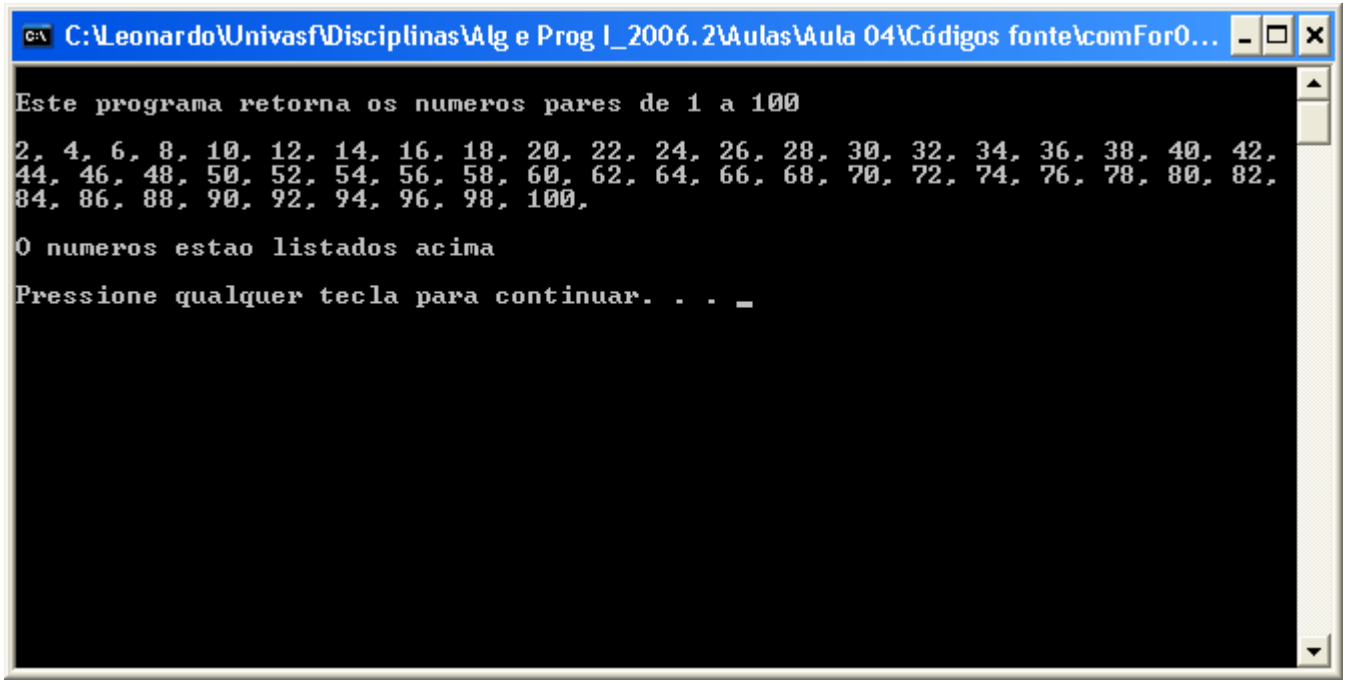
```
#include <stdio.h>
/* Este programa exemplifica a utilização do comando for com if */

int main()
{
    int i;
    printf("\nEste programa retorna os numeros pares de 1 a 100\n\n");
    for(i=2; i<=100; i+=2)
    {
        printf("%d, ", i);
    }
    printf("\n\nO numeros estao listados acima\n\n");
    system("pause");
    return 0;
}
```

i = i +2;

A lógica também mudou um pouco

O laço for

- A saída no console para o programa anterior será:



Variações do laço for

- Uma das variações mais comuns do laço for usa o operador vírgula (,) para permitir que duas ou mais variáveis controlem o laço, veja:

```
int x, y;
for (x=0, y=0; x+y<10, x++)
{
        y = getche();
        x = y - '0';/* Subtrai o código ASCII do caracter 0 de Y */
}
```

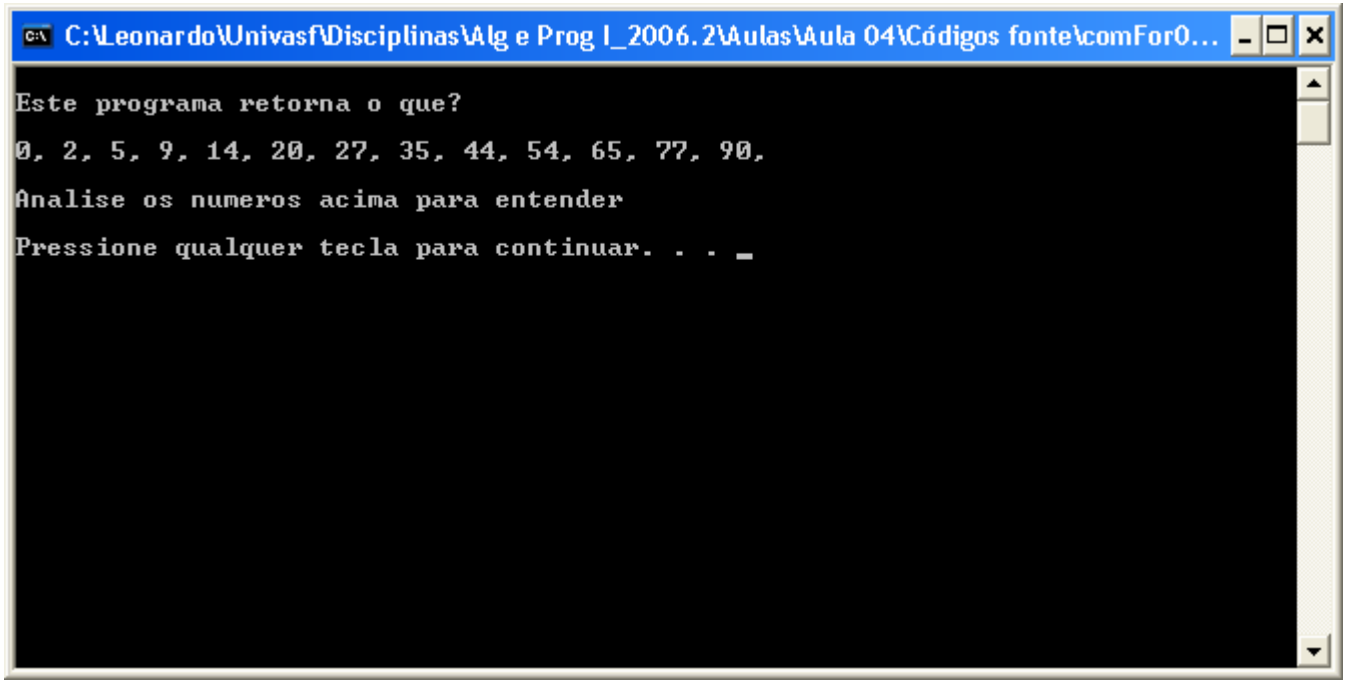
Variações do laço for

- Além da sintaxe vista anteriormente, o laço for permite escrita de expressões mais elaboradas, vejamos:

```
#include <stdio.h>
/* Este programa exemplifica a utilização do comando for mais elaborado */

int main()
{
    int x, y;
    printf("\nEste programa retorna o que?\n\n");
    for(x=0, y=0; x+y<100; ++x, y=y+x)
    {
            printf("%d, ", x+y);
    }
    printf("\n\nAnalise os numeros acima para entender\n\n");
    system("pause");
    return 0;
}
```

Variações do laço for

- A saída no console para o programa anterior será:



Laços for aninhados

- Quando um laço for faz parte de outro laço for, dizemos que o laço interno está aninhado. Vejamos:

```
int i,j;
for(i=0; i<3; i++)
{
        for(j=0; j<3; j++)
        printf("%d, ", i+j);
}
```

saída
0
1
2
1
2
3
2
3
4

O laço while

- O segundo laço disponível em C é o laço while. A sua forma geral é:

```
while(condição)
{
        comando;
}
```

  - *comando* é um comando vazio, um comando simples ou um bloco de comandos;
  - A *condição* pode ser qualquer expressão, e verdadeiro é qualquer valor não-zero;
  - O laço se repete quando a condição for verdadeira. Quando a condição é falsa, o controle do programa passa para a linha após o ódigo do laço

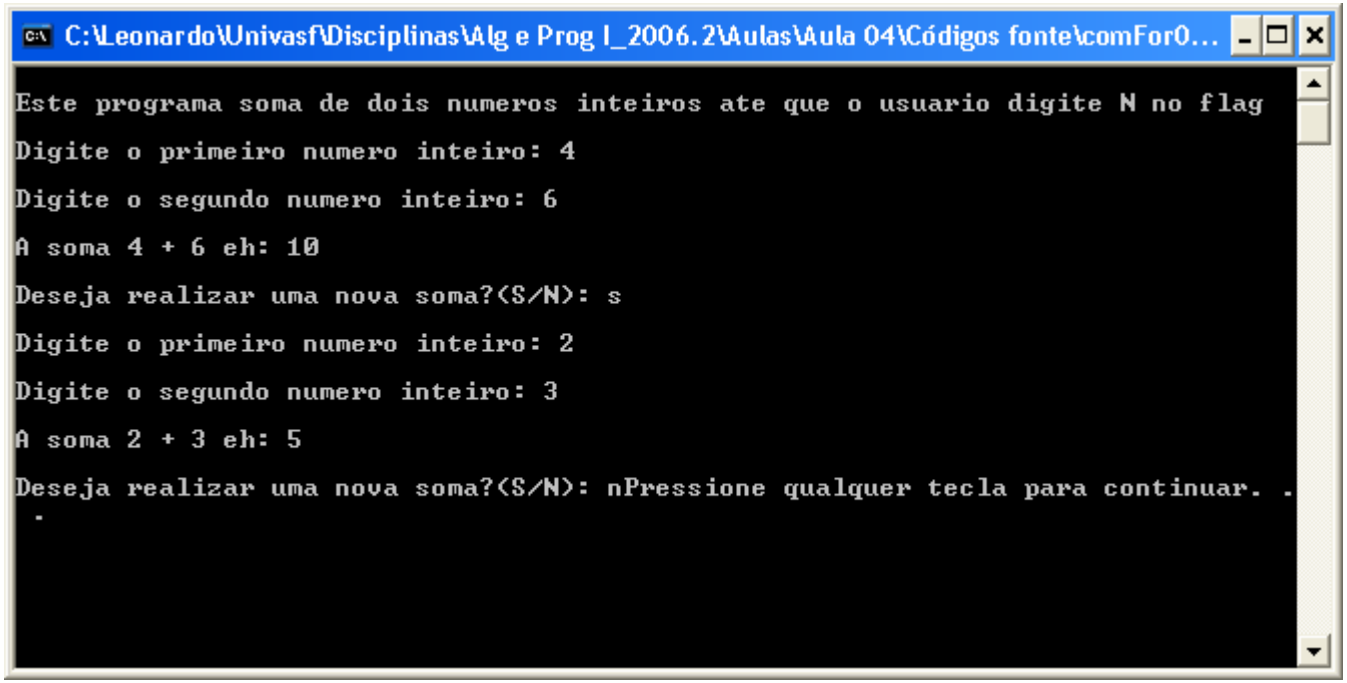
O laço while

- O exemplo a seguir mostra uma rotina de entrada pelo teclado, que simplesmente se repete até que o usuário digite n:

```
#include<stdio.h>
/* Este programa calcula a soma de dois números inteiros até que o usuário digite N no flag */
int main()
{
    int num1, num2;
    char ch='\0';
    printf("\nEste programa soma de dois numeros inteiros ate que o usuario digite N no flag");
    while(ch!= 'N' && ch!= 'n')
    {
            printf("\n\nDigite o primeiro numero inteiro: ");
            scanf("%d", &num1);
            printf("\nDigite o segundo numero inteiro: ");
            scanf("%d", &num2);
            printf("\nA soma %d + %d eh: %d ",num1, num2, num1+num2);
            printf("\n\nDeseja realizar uma nova soma?(S/N): ");
            ch = getche();
    }
    system("pause");
    return 0;
}
```

O laço while

- A saída no console para o programa anterior será:



O laço do-while

- Ao contrário dos laços for e while, que testam a condição do laço no começo, o laço do-while verifica a condição ao final do laço;
- Portanto, o laço do-while será executado ao menos uma vez;
- A forma geral do lado do-while é:

```
do{
        comando;
} while(condição);
```

  - O laço do-while repete até que a *condição* se torne falsa.

O laço do-while

- Vejamos a principal diferença entre o laço do-while e o laço while:

```
int num = 101;
do{
   scanf("%d", &num);
} while(num<100);
```

```
int num = 101;
while(num<100)
{
   scanf("%d", &num);
}
```

  - do-while executa pelo menos uma vez.

O laço do-while

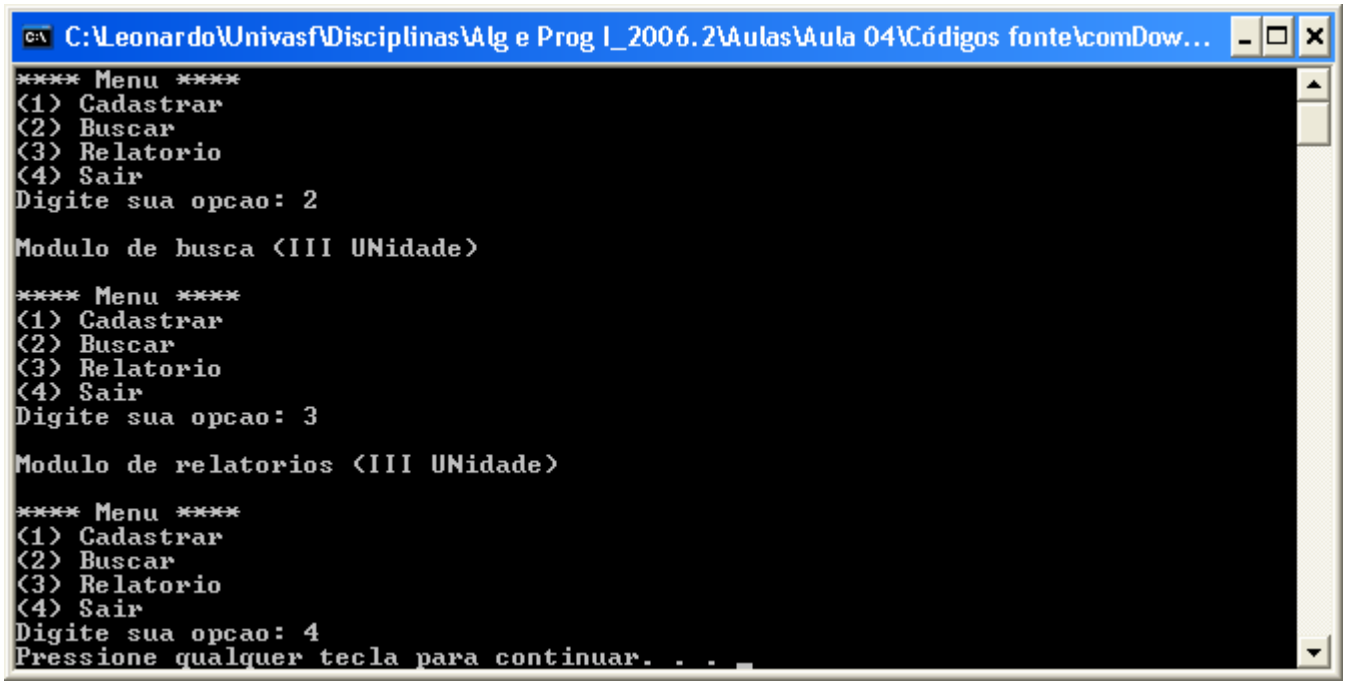
- Talvez o uso mais comum do laço do-while seja em uma rotina de seleção por menu, vejamos:

O laço do-while

```
int main()
{
    int opt;
    do
    {
        do
        {
            printf("**** Menu ****");
            printf("\n(1) Cadastrar ");
            printf("\n(2) Buscar ");
            printf("\n(3) Relatorio ");
            printf("\n(4) Sair ");
            printf("\nDigite sua opcao: ");
            scanf("%d", &opt);
        } while ((opt<1) || (opt>4));
        switch(opt)
        {
                    case 1:
                        printf("\nModulo de cadastramento (III UNidade)\n\n");
                        break;
                    case 2:
                        printf("\nModulo de busca (III UNidade)\n\n");
                        break;
                    case 3:
                        printf("\nModulo de relatorios (III UNidade)\n\n");
                        break;
        }
    } while(opt != 4);
    system("pause");
    return 0;
}
```

O laço do-while

- A saída no console para o programa anterior será:



Bibliografia

- SCHILDT H. "*C Completo e Total*", Makron Books. SP, 1997.
- MIZRAHI, V. V. "*Treinamento em Linguagem C++ Módulo 1*", Makron Books, SP, 1995.
- FORBELLONE, A. L. V. "*Lógica de Programação: A construção de algoritmos e estruturas de dados*", Prentice Hall, SP, 2005.