Algoritmos e Programação de Computadores

Linguagem e Técnica de Programação - C

* *Todos os direitos reservados nos termos da Lei Nº 9.610/98.*

Programa da Disciplina

- **Organização básica de computadores**
- **Algoritmos**
- **Introdução a linguagem de programação estruturada**
- **Tipos de dados, identificadores, constantes e variáveis**
- **Comandos de entrada/saída e operadores**
- **Estruturas de decisão (*if*, *switch*)**
- **Estruturas de repetição (*for*, *while*, *do-while*)**
- **Avaliação P1**
- **Vetores e matrizes**
- ***Strings* e estruturas**
- **Modularização de programas**
- **Ponteiros e alocação dinâmica de memória**
- **Avaliação P2**

Sumário (Estruturas de Decisão)

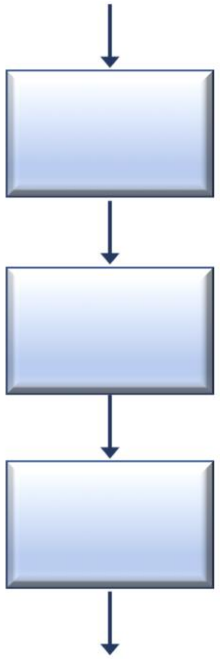
O comando *if*

- **Instrução única**
- **Bloco de instruções**
- **Encadeado**
- ***if-else***
- **Operador condicional ternário ?:**

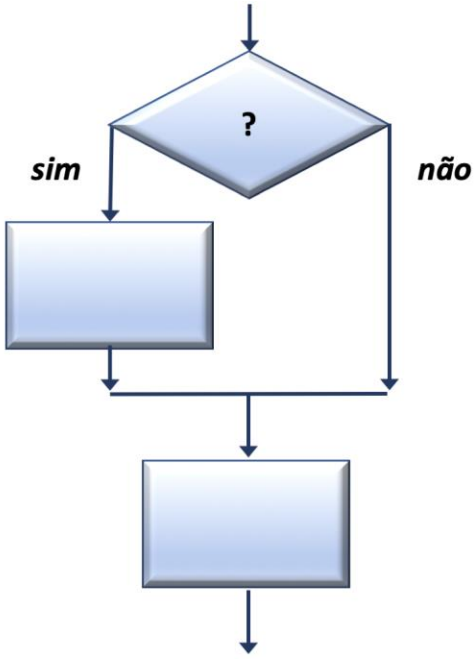
O comando *switch-case*

- **Com *break***
- **Sem *break***

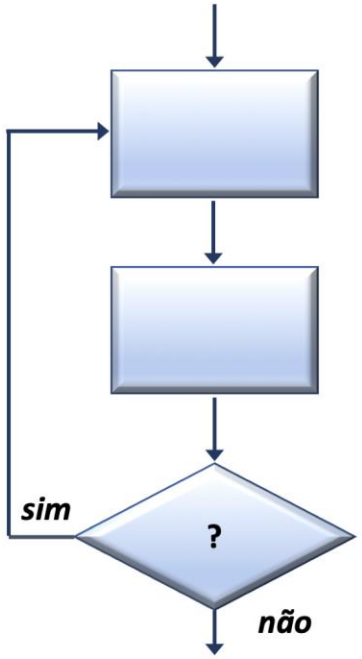
Programação Estruturada



Estrutura de Controle Sequencial



Estrutura de Controle Condicional



Estrutura de Controle Repetitiva

A estrutura de decisão é utilizada quando a execução de uma instrução (ou de um bloco de instruções) depende de um teste anterior (uma ou mais comparações).

O comando *if* – instrução única

Usado para execução condicional de uma instrução.

Sintaxe:

```
if (condição)
    instrução;
```

- **condição = expressão lógica. Se for verdadeira a instrução é executada.**
- **instrução = código a ser executado caso a condição seja verdadeira.**

Observação: recomenda-se o uso da indentação (espaçamento extra para instruções internas a estruturas de decisão, repetição, funções, etc).

O comando *if* – instrução única

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.      char c;
5.      printf ("Você quer ver uma mensagem na tela <s/n>?");
6.      scanf ("%c", &c);
7.      if (c == ‘s’ || c==‘S’)
8.          printf ("Você escolheu sim!");
9.      return 0;
10. }
```

```
Você quer ver uma mensagem na tela <s/n>?s
Você escolheu sim!
```

O comando *if* – bloco de instruções

Usado para execução condicional de um bloco de instruções.

Sintaxe:

```
if (condição)
{
    instrução-1;
    instrução-2;
    ...
    instrução-n;
}
```

Um bloco de instruções (*instrução-1; instrução-2; ... instrução-n;*) sempre deve ser delimitado por chaves {}. Sem elas o programa compila, mas apresenta erro lógico.

O comando *if* – bloco de instruções

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.      float nota1, nota2, media;
5.      printf ("Informe as duas notas: \n");
6.      scanf ("%f %f", &nota1, &nota2);
7.      media = (nota1 + nota2) / 2;
8.      if (media > 5)
9.      {
10.         printf ("Média: %5.2f\n", media);
11.         printf ("Aluno aprovado!");
12.     }
13.     return 0;
14. }
```

```
Informe as duas notas:
6
8
Média:  7.00
Aluno aprovado!
```

O comando *if* – bloco de instruções (sem chaves)

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.      float nota1, nota2, media;
5.      printf ("Informe as duas notas: \n");
6.      scanf ("%f %f", &nota1, &nota2);
7.      media = (nota1 + nota2) / 2;
8.      if (media > 5)
9.
10.         printf ("Média: %5.2f\n", media);
11.         printf ("Aluno aprovado!");
12.
13.     return 0;
14. }
```

```
Informe as duas notas:
4
5
Aluno aprovado!
```

Erro lógico!

O programa compila, mas funciona incorretamente.

O comando *if* – encadeado

Um comando *if* pode estar dentro do corpo de outro comando *if*.

Sintaxe:

```
if (condição)
{
    instruções;
    if (condição)
    {
        instruções;
    }
}
```

Novamente cada comando *if* pode ter uma ou várias instruções. Para várias instruções as chaves são necessárias.

O comando *if* – encadeado

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.      int faltas;
5.      float nota1, nota2, media;
6.      printf ("Informe o número de faltas: \n");
7.      scanf ("%d", &faltas);
8.      if (faltas < 3)
9.      {
10.         printf ("Informe as duas nota: \n");
11.         scanf ("%f %f", &nota1, &nota2);
12.         media = (nota1 + nota2) / 2;
13.         if (media > 5)
14.             printf("Aluno aprovado com média %5.2f.", media);
15.     }
16.     return 0;
17. }
```

```
Informe o número de faltas:
1
Informe as duas nota:
6
8
Aluno aprovado com média  7.00.
```

O comando *if – else*

Para o comando *if – else* podem ser definidas instruções a serem executadas caso a condição seja verdadeira e instruções a serem executadas caso a condição seja falsa.

Sintaxe:

```
if (condição)
{
    instruções-A;
}
else
{
    instruções-B;
}
```

- **instruções-A;**
  executadas se a condição for verdadeira.

- **instruções-B;**
  executadas se a condição for falsa.

O comando *if – else*

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.      int faltas;
5.      float nota1, nota2, media;
6.      printf ("Digite faltas, nota1 e nota2:\n");
7.      scanf ("%d %f %f", &faltas, &nota1, &nota2);
8.      media = (nota1 + nota2) / 2;
9.      if (faltas < 3 && media > 5)
10.         printf ("Aluno aprovado com %d faltas e média %4.2f. \n", faltas, media);
11.     else
12.         printf ("Aluno reprovado!");
13.     return 0;
14. }
```

```
Digite faltas, nota1 e nota2:
2
6
8
Aluno aprovado com 2 faltas e média 7.00.
```

O comando *if* – *sequência de if(s)*

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.      int faltas;
5.      float nota1, nota2, media;
6.      printf ("Digite faltas, nota1 e nota2:\n");
7.      scanf ("%d %f %f", &faltas, &nota1, &nota2);
8.      media = (nota1 + nota2) / 2;
9.      if (faltas < 3 && media > 5)
10.         printf ("Aluno aprovado (%d faltas e média = %4.2f). \n", faltas, media);
11.     if (faltas >= 3)
12.         printf ("Aluno reprovado por faltas (%d faltas).", faltas);
13.     if (media <= 5)
14.         printf ("Aluno reprovado por nota (média = %4.2f ).", media);
15.     return 0;
16. }
```

```
Digite faltas, nota1 e nota2:
2
4
5
Aluno reprovado por nota (média = 4.50 ).
```

TODAS as comparações são feitas.
Código não eficiente!

O comando *if – else* (encadeado)

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.      int faltas;
5.      float nota1, nota2, media;
6.      printf ("Digite faltas, nota1 e nota2:\n");
7.      scanf ("%d %f %f", &faltas, &nota1, &nota2);
8.      media = (nota1 + nota2) / 2;
9.      if (faltas < 3 && media > 5)
10.         printf ("Aluno aprovado (%d faltas e média = %4.2f). \n", faltas, media);
11.     else
12.         if (faltas >= 3)
13.             printf ("Aluno reprovado por faltas (%d faltas).", faltas);
14.         else
15.             printf ("Aluno reprovado por nota (média = %4.2f ).", media);
16.     return 0;
17. }
```

```
Digite faltas, nota1 e nota2:
2
4
5
Aluno reprovado por nota (média = 4.50 ).
```

Evita comparações desnecessárias.
Código mais eficiente!

Sequência de *if(s)* e *if – else* (encadeado)

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.      int x,y;
5.      printf ("Digite dois números inteiros:\n");
6.      scanf ("%d %d", &x, &y);
7.      if (x > y)
8.          printf ("O maior valor é %d.\n", x);
9.      if (y > x)
10.         printf ("O maior valor é %d.\n", y);
11.     if (x == y)
12.         printf ("Os valores são iguais.\n");
13.     return 0;
14. }
```

TODAS as comparações são feitas!

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.      int x,y;
5.      printf ("Digite dois números inteiros:\n");
6.      scanf ("%d %d", &x, &y);
7.      if (x > y)
8.          printf ("O maior valor é %d.\n", x);
9.      else
10.         if (y > x)
11.             printf ("O maior valor é %d.\n", y);
12.         else
13.             printf ("Os valores são iguais.\n");
14.     return 0;
15. }
```

Evita comparações desnecessárias!

Operador Condicional Ternário ?:

O uso do operador condicional pode substituir o comando *if-else*. Ele deixa tais expressões mais "compactas".

Sintaxe:

condição ? instrução-A : instrução-B;

- **condição: expressão lógica.**
- **instrução-A: executada caso a condição seja verdadeira.**
- **instrução-B: executada caso a condição seja falsa.**

Operador Condicional Ternário ?:

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.      int x, y;
5.      printf ("Informe dois valores:\n");
6.      scanf ("%d %d", &x, &y);
7.      printf ("Máximo = %d", x>y ? x : y);
8.      return 0;
9.  }
```

```
Informe dois valores:
8
5
Máximo = 8
```

O comando *switch*

Usado quando precisamos escolher uma entre várias alternativas.

Tem o mesmo efeito do *if-else* encadeado, mas com formato mais flexível e mais claro.

```
switch (expressão constante)
{
    case constante-1:
        instruções;
        break;
    case constante-2:
        instruções;
        break;
    ...
    default:
        instruções;
}
```

- **O comando *switch* avalia a expressão constante.**
- **Compara esta expressão com a constante (rótulo) de cada case até encontrar um case correspondente, ou seja, de igual valor.**
- **Caso encontre, executa TODAS as instruções seguintes até chegar no comando *break* (que causa uma saída imediata do *switch*).**
- **Caso não encontre, executa as instruções associadas ao *default*.**

O comando *switch*

```
switch (expressão constante)
{
    case constante-1:
        instruções;
        break;
    case constante-2:
        instruções;
        break;
    ...
    default:
        instruções;
}
```

- **Expressão constante e constante devem ser do tipo *char* ou *int*.**
- **NÃO é possível usar variáveis ou expressões para as constantes.**
- **O corpo do *switch* deve estar entre "{}". A constante de cada case deve ser finalizada com ":". Caso múltiplas instruções sigam um case, não é necessário envolvê-las em "{}".**
- **O caso *default* é opcional e, embora seja geralmente posicionado no final do bloco *switch*, ele pode aparecer em qualquer posição entre os case´s especificados.**

O comando *switch* (com *break*)

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.    int numA = 8, numB = 2; char op;
5.    printf ("Escolha um operador: < + - * / > ");
6.    op = getchar();
7.    switch(op)   {
8.        case '+':
9.            printf ("Soma = %d\n", numA + numB); break;
10.       case '-':
11.           printf ("Subtração = %d\n", numA - numB); break;
12.       case '*':
13.           printf ("Multiplicação = %d\n", numA * numB); break;
14.       case '/':
15.           printf ("Divisão = %d\n", numA / numB); break;
16.       default:
17.           printf ("Operador desconhecido");   }
18.   return 0;
19. }
```

```
Escolha um operador aritmético: +
Soma = 10
```

O comando *switch* (sem *break*)

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.    int numA = 8, numB = 2; char op;
5.    printf ("Escolha um operador: < + - * / > ");
6.    op = getchar();
7.    switch(op)   {
8.        case '+':
9.            printf ("Soma = %d\n", numA + numB);
10.       case '-':
11.           printf ("Subtração = %d\n", numA - numB);
12.       case '*':
13.           printf ("Multiplicação = %d\n", numA * numB);
14.       case '/':
15.           printf ("Divisão = %d\n", numA / numB);
16.       default:
17.           printf ("Operador desconhecido");   }
18.   return 0;
19. }
```

```
Escolha um operador aritmético: +
Soma = 10
Subtração = 6
Multiplicação = 16
Divisão = 4
Operador desconhecido
```

```
Escolha um operador aritmético: -
Subtração = 6
Multiplicação = 16
Divisão = 4
Operador desconhecido
```

```
Escolha um operador aritmético: *
Multiplicação = 16
Divisão = 4
Operador desconhecido
```

Sem o *break* o controle "vaza" de um caso para o outro!

O comando *switch*

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.    int numA = 8, numB = 2; char op;
5.    printf ("Escolha um operador: < + - * / > ");
6.    op = getchar();
7.    switch(op)   {
8.        case '+':
9.            printf ("Soma = %d\n", numA + numB); break;
10.       case '-':
11.           printf ("Subtração = %d\n", numA - numB); break;
12.       case '*':
13.           printf ("Multiplicação = %d\n", numA * numB); break;
14.       case '/':
15.           if (numB != 0)  {
16.               printf (" A divisão pode ser realizada!\n");
17.               printf (" Divisão = %d\n", numA / numB);   } break;
18.       default:
19.           printf ("Operador desconhecido");    }
20.   return 0;
21. }
```

```
Escolha um operador aritmético: /
A divisão pode ser realizada!
Divisão = 4
```

Podemos usar qualquer comando associado a um case!

Cuidado com o Enter no buffer!
Soluções:

a) Acrescente antes da linha 5:
getchar();

ou

a) Substitua a linha 6 por:
scanf (" %c", &op);

Atividades

1) Utilizando a estrutura de condição *if ()*, escreva um programa que solicite ao usuário os coeficientes a, b e c de uma equação do 2º grau. Calcule o delta e as raízes da equação.

- Se delta > 0, apresente na tela: "A equação tem duas raízes reais diferentes = ? e ?".
- Se delta = 0, apresente na tela : "A equação tem duas raízes reais e iguais a = ?".
- Se delta < 0, apresente na tela : "A equação não tem raízes reais".

2) Repita o exercício anterior usando a estrutura de condição *if-else ()*. É possível implementar o exercício 1 utilizando *switch ()*? Justifique a sua resposta.

3) Pesquise a fórmula para calcular o Índice de Massa Corpórea (IMC) de uma pessoa. Solicite os dados necessários ao usuário. Em função do resultado do IMC, apresente na tela uma mensagem conforme a tabela ao lado.

Teste o programa das atividades 1 e 2 para os seguintes valores:

- Para a = 1, b = -3, c = -10 (raízes 5 e -2)
- Para a = 9, b = -12, c = 4 (raiz 2/3)
- Para a = 3, b = 3, c = 5 (não tem raiz)

IMC	CLASSIFICAÇÃO
MENOR QUE 18,5	MAGREZA
ENTRE 18,5 E 24,9	NORMAL
ENTRE 25,0 E 29,9	SOBREPESO
ENTRE 30,0 E 39,9	OBESIDADE
MAIOR QUE 40,0	OBESIDADE GRAVE

Referências

- **FORBELLONE, A. L. V., EBERSPACHER, H. F. Lógica de Programação – A Construção de Algoritmos e Estruturas de Dados, 3ª Edição, São Paulo, Pearson Prentice Hall, 2005.**
- **SOUZA, M. A. F., GOMES, M. M., SOARES, M. V., CONCILIO, R. Algoritmos e Lógica de Programação, 3ª Edição, São Paulo, Cengage, 2019.**
- **PUGA, S., RISSETTI, G. Lógica de Programação e Estrutura de Dados, 3ª. Edição, Prentice Hall, 2016.**
- **DEITEL, H. M., DEITEL, P. J. C: Como Programar. LTC, 2011.**
- **ASCENCIO, A. F. G., CAMPOS, E. A. V. Fundamentos da Programação de Computadores – Algoritmos, Pascal e C/C++. Pearson Prentice Hall, 2012.**
- **VAREJÃO, F. M. Introdução à Programação: Uma nova abordagem usando C. Campus, 2015.**
- **CELES, W., CERQUEIRA, R., RANGEL, J. L. Introdução a Estrutura de dados com Técnicas de Programação em C. Campus, 2016.**
- **MIZRAHI, V. V. Treinamento em Linguagem C – Curso Completo, 2ª Edição, Pearson Makron Books, 2008.**
- **SCHILDT, H., C Completo e Total, 3ª Edição, Makron Books, 1997.**