Programação de Computadores I

Aula 07

Programação: Estruturas de seleção

José Romildo Malaquias

Departamento de Computação
Universidade Federal de Ouro Preto

2011-1

Valores booleanos

- Os **valores booleanos** (ou **valores verdade**) são ***verdadeiro*** e ***falso***.
- São muito utilizados para indicar se uma **condição** é satisfeita ou não, e executar comandos diferentes em cada caso.
- Diferentemente da maioria das linguagens de programação, C não possui um tipo de dados específico para os valores booleanos.
- Em C o tipo `int` é utilizado para os valores booleanos.
- O valor ***verdadeiro*** é representado por qualquer número inteiro diferente de zero, e o valor ***falso*** é representado por zero.

Operações relacionais

- **Operações relacionais** comparam dois valores, resultando em um valor booleando.
- `<expressão> == <expressão>`
  retorna verdadeiro quando as expressões forem iguais.
  Ex: `a == b`
- `<expressão> != <expressão>`
  retorna verdadeiro quando as expressões forem diferentes.
  Ex: `a != b`

Operações relacionais (cont.)

- `<expressão> > <expressão>`
  retorna verdadeiro quando a expressão da esquerda tiver valor maior que a expressão da direita.
  Ex: `a > b`
- `<expressão> < <expressão>`
  retorna verdadeiro quando a expressão da esquerda tiver valor menor que a expressão da direita.
  Ex: `a < b`

Operações relacionais (cont.)

- `<expressão> >= <expressão>`
  retorna verdadeiro quando a expressão da esquerda tiver valor maior ou igual que a expressão da direita.
  Ex: `a >= b`
- `<expressão> <= <expressão>`
  retorna verdadeiro quando a expressão da esquerda tiver valor menor ou igual que a expressão da direita.
  Ex: `a <= b`

Operações Lógicas

- **Operações lógicas** (*ou*, *e*, *não*, *etc*) são realizadas com valores lógicos, resultando em outro valor lógico (verdadeiro ou falso, como nas oeprações relacionais).

Operações Lógicas (cont.)

- `<expressão> && <expressão>`
  retorna verdadeiro quando ambas as expressões são verdadeiras.
  Sua tabela de verdade é

a	b	a && b
V	V	V
V	F	F
F	V	F
F	F	F

Ex: `a == 0 && b == 0`

Operações Lógicas (cont.)

- `<expressão>` || `<expressão>`
  retorna verdadeiro quando pelo menos uma das expressões é verdadeiras.
  Sua tabela de verdade é

a	b	a || b
V	V	V
V	F	V
F	V	V
F	F	F

Ex: `a == 0 || b == 0`

Operações Lógicas (cont.)

- `!` `<expressão>`
  retorna verdadeiro quando a expressão é falsa, e retorna falso quando a expressão é verdadeira.
  Sua tabela de verdade é

a	!a
V	F
F	V

  Ex: `! (a == 0)`

Operações Lógicas (cont.)

- A operações lógicas são frequentemente usadas em combinação com operadores relacionais, como por exemplo:
  - `((2 > 1) || (3 < 7))` : resultado VERDADEIRO
  - `((3 < 2) && (2 == 2))` : resultado FALSO
  - `((5 != 0) || (1 < 2))` : resultado VERDADEIRO

Estruturas de seleção

- Os **comandos de seleção** permitem selecionar entre várias alternativas de comandos, qual deve ser executada.
- Com os comandos **`if`** e **`if else`** a escolha das alternativas é baseada no resultado de uma condição, isto é, de uma expressão booleana que resulta em um valor verdadeiro ou falso.
- Com o comando **`switch`** a escolha da alternativa a ser executa é baseada no valor de uma expressão inteira.

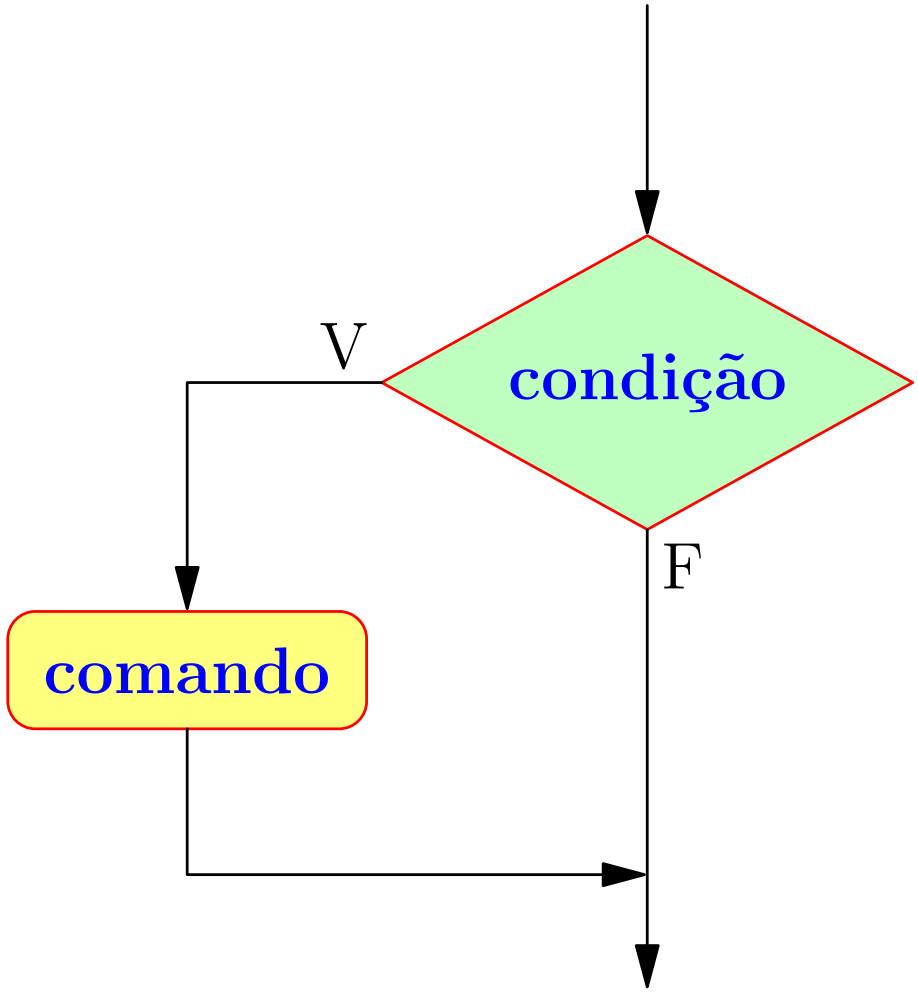
Comando `if`

- O comando **`if`** serve para alterar o fluxo de execução de um programa baseado no valor de uma condição, ou seja, de uma expressão lógica (verdadeiro ou falso).

```
if (expr_log)
    comando
```

- O comando no corpo do **`if`** somente será executado se a condição expressa por *expr_log* for verdadeira.

Comando `if` (cont.)



Comando `if` (cont.)

Exemplo

Ler a idade de uma pessoa e exibir a mensagem *menor* se a idade for menor que 18 anos.

Comando `if` (cont.)

```c
#include <stdio.h>
int main(void)
{
    int idade;
    printf("Digite a idade: ");
    scanf("%d", &idade);
    if (idade < 18)
        printf("menor\n");
    printf("fim\n");
    return 0;
}
```

Comando `if` (cont.)

Exemplo:
Escreva um programa que lê o valor total das vendas de um vendedor e calcula e exibe o prêmio que o vendedor deverá receber, correspondente a 7% do valor das vendas. Se o valor do prêmio for maior do que 200, o programa imprime ainda uma mensagem de congratulação.

Comando `if` (cont.)

```c
#include<stdio.h>
int main(void)
{
    double vendas;
    printf("Valor total das vendas: ");
    scanf("%lf", &vendas);
    double premio = vendas * 0.07;
    printf("Prêmio: %.2f\n", premio);
    if (premio >= 200)
        printf("Congratulações pelo prêmio!\n");
    return 0;
}
```

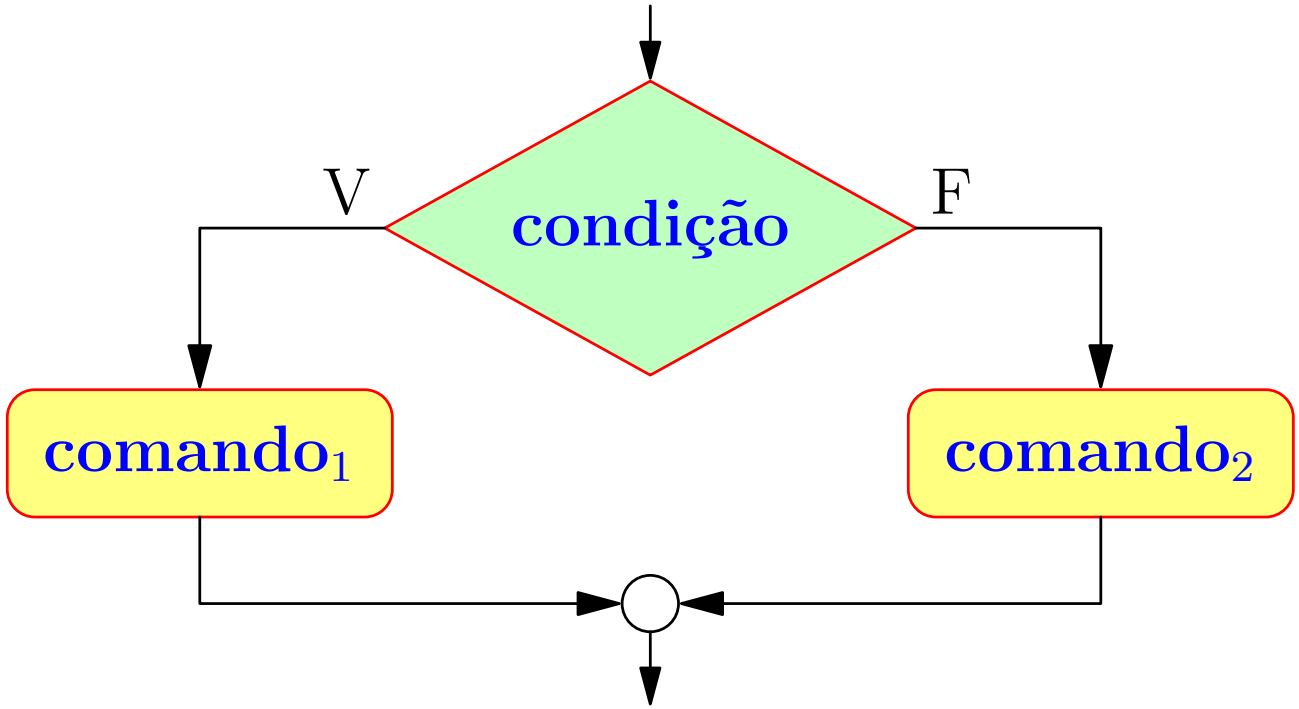
Comando `if else`

- O comando **`if else`** permite escolher entre duas alternativas com base no valor de uma condição.
- Uma **condição** é uma expressão booleana, e o seu valor pode ser *verdadeiro* ou *falso*.
- Portanto a escolha pode ser feita no máximo entre duas alternativas.

```
if (exprlog)
    comando1;
else
    comando2;
```

- Primeiro avalia-se a condição representada por *exprlog*. Se o resultado for verdadeiro, executa-se o primeiro comando *comando1* e o segundo comando *comando2* é ignorado. Caso contrário o comando *comando2* é executado e o comando *comando1* é ignorado.

Comando `if else` (cont.)



Comando `if else` (cont.)

Comando `if else` (cont.)

Exemplo:
Ler dois números e exibir o maior deles.

Comando `if else` (cont.)

```c
#include <stdio.h>
int main(void)
{
    double num1, num2;
    printf("Digite dois números: ");
    scanf("%lf%lf", &num1, &num2);
    if (num1 > num2)
        printf("O maior número é %f\n", num1);
    else
        printf("O maior número é %f\n", num2);
    return 0;
}
```

Comando `if else` (cont.)

Exemplo:
Dado um número inteiro, informar se ele é par ou ímpar.

Comando **if else** (cont.)

```c
#include <stdio.h>
int main(void)
{
    int num;
    printf("Digite um numero inteiro: ");
    scanf("%d", &num);
    if (num % 2 == 0) //divisível por dois?
        printf("%d é par\n", num);
    else
        printf("%d é ímpar\n", num);
    return 0;
}
```

Comando `if else` (cont.)

Exemplo:
Faça um algoritmo que leia os valores A, B, C e imprima na tela se a soma de $A + B$ é menor que C ou maior igual a C.

Comando **if else** (cont.)

```c
#include<stdio.h>
int main(void)
{
    int A, B, C;
    printf("Inserir 3 números:\n");
    scanf("%d %d %d", &A, &B, &C);
    if (A+B >= C)
        printf("%d + %d >= %d\n", A, B, C);
    else
        printf("%d + %d < %d\n", A, B, C);
    return 0;
}
```

Comando de bloco

- Cada alternativa que escrevemos nos comandos condicionais deve ser formados por um único comando.
- Então o que fazer quando precisamos executar mais de um comando dentro de uma única alternativa?
- Se simplesmente escrevemos um comando depois do outro:

```
if (expr_log)
   comando1;
   comando2;
```

não vai funcionar, pois apenas o *comando1* faz parte do **if**. *comando2* será executado sempre, logo depois que o comando **if** terminar.

Comando de bloco (cont.)

- Solução: usar um comando de bloco.
- O **comando de bloco** é formado por uma sequência de comandos colocados entre chaves.

```
if (expr_log)
{
   comando1; //executado se expr_log for verdadeira
   comando2;
   comando3;
}
else
{
   comando4; //executado se expr_log for falsa
   comando5;
}
```

Comando de bloco (cont.)

Exemplo:
Dado um número inteiro, exibir uma mensagem informando se o número é par ou ímpar, e exibir seu quadrado quando ele for par, ou seu cubo quando ele for ímpar.

Comando de bloco (cont.)

```c
#include <stdio.h>
int main(void)
{
    int num;
    printf("Digite um número inteiro: ");
    scanf("%d", &num);
    if (num % 2 == 0) //divisível por dois?
    {
        printf("%d é par.\n", num);
        printf("o quadrado de %d é %d\n", num, num*num);
    }
    else
    {
        printf("%d é ímpar.\n", num);
        printf("o cubo de %d é %d\n", num, num*num*num);
    }
    return 0;
}
```

Comando de bloco (cont.)

Exemplo:
Faça um algoritmo que leia o nome, o sexo e o estado civil de uma pessoa. Caso o sexo seja femenino e estado civil seja casado, solicitar o tempo de casamento (em anos). As entradas para sexo serão: F para femenino ou M para masculino, e para estado civil: C para casdo ou S para solteiro.

Comando de bloco (cont.)

```c
#include<stdio.h>
int main(void)
{  char nome[50];
   char sexo, estadoCivil;
   int anos;
   printf("Nome: ");
   scanf(" %49[^\n]%*[^\n]", nome);
   printf("Sexo (M: masculino, F: femenino): ");
   scanf(" %c", &sexo);
   printf("Estado civil (C: casado, S: solteiro): ");
   scanf(" %c", &estadoCivil);
   if (sexo == 'F' && estadoCivil == 'C')
   {
      printf("Tempo de casamento: ");
      scanf("%d", &anos);
   }
   return 0;
}
```

Aninhamento de `If`s

```
if (expr_log)
  if (expr_log2)
    comando1; //executado se expr_log e
              //expr_log2 forem verdadeiras
  else
    comando2; //expr_log verdadeira e expr_log2 falsa
else
  comando3; //executado se expr_log é falsa
```

Aninhamento de `If`s (cont.)

Exemplo:
Mostrar se um número par é divisível por 3.

Aninhamento de `If`s (cont.)

```c
#include <stdio.h>
int main(void)
{
  int num;
  printf("Digite um número inteiro: ");
  scanf("%d", &num);
  if (num % 2 == 0) // o número é par?
    if (num % 3 == 0) // o número é divisível por 3?
      printf("É par e divisível por 3\n");
    else
      printf("É par mas não é divisível por 3\n");
  else
    printf("Não é par\n");
  return 0;
}
```

Aninhamento de `If`s (cont.)

Exemplo:
Dado um número par determinar se ele é ou não divisível por 3. Se o número for ímpar mostrar se ele é divisível ou não por 5.

Aninhamento de `If`s (cont.)

```c
#include <stdio.h>
int main(void)
{
    int num;
    printf("Digite o numero: ");
    scanf("%d", &num);
    if (num % 2 == 0) // divisível por dois => PAR
        if (num % 3 == 0)
            printf("Par divisível por 3\n");
        else
            printf("Par não divisível por 3\n");
    else // Se não for par, é ÍMPAR
        if (num % 5 == 0)
            printf("Ímpar divisível por 5\n");
        else
            printf("Ímpar não divisível por 5\n");
    return 0;
}
```

Aninhamento de `If`s (cont.)

Exemplo:
Ler as quatro notas escolares de um aluno e imprimir uma mensagem dizendo que o aluno foi aprovado se o valor da média escolar for maior ou igual a 7,0. Se a média for entre 5,0 (inclusive) e 7,0, informar que o aluno está em recuperação. Se a média for inferior a 5.0 o aluno foi reprovado.

Aninhamento de `If`s (cont.)

```c
#include<stdio.h>
int main(void)
{
    double n1, n2, n3, n4;
    printf("Digite as 4 notas do aluno: ");
    scanf("%lf%lf%lf%lf", &n1, &n2, &n3, &n4);
    double media = (n1 + n2 + n3 + n4) / 4.0;
    printf("A média do aluno é %.2f\n", media);
    if (media >= 7)
        printf("Aluno aprovado\n");
    else if (media >= 5)
        printf("Aluno em recuperação\n");
    else
        printf("Aluno reprovado\n");
    return 0;
}
```

Estrutur de seleção múltipla: comando `switch`

- Permite a escolha de uma entre **várias alternativas**.
- A seleção é feita através de uma **expressão de controle**.
- Cada alternativa é uma lista de comandos **rotulada** com uma **constante** (literal).
- A expressão de seleção e as constantes devem ser de um mesmo **tipo integral** (`char`, `int`, `short int`, `long int`, etc.)
- `float`, `double` e `string` não são permitidos.
- A lista de comandos de uma alternativa só termina com o comando `break` ou no final do `switch`.

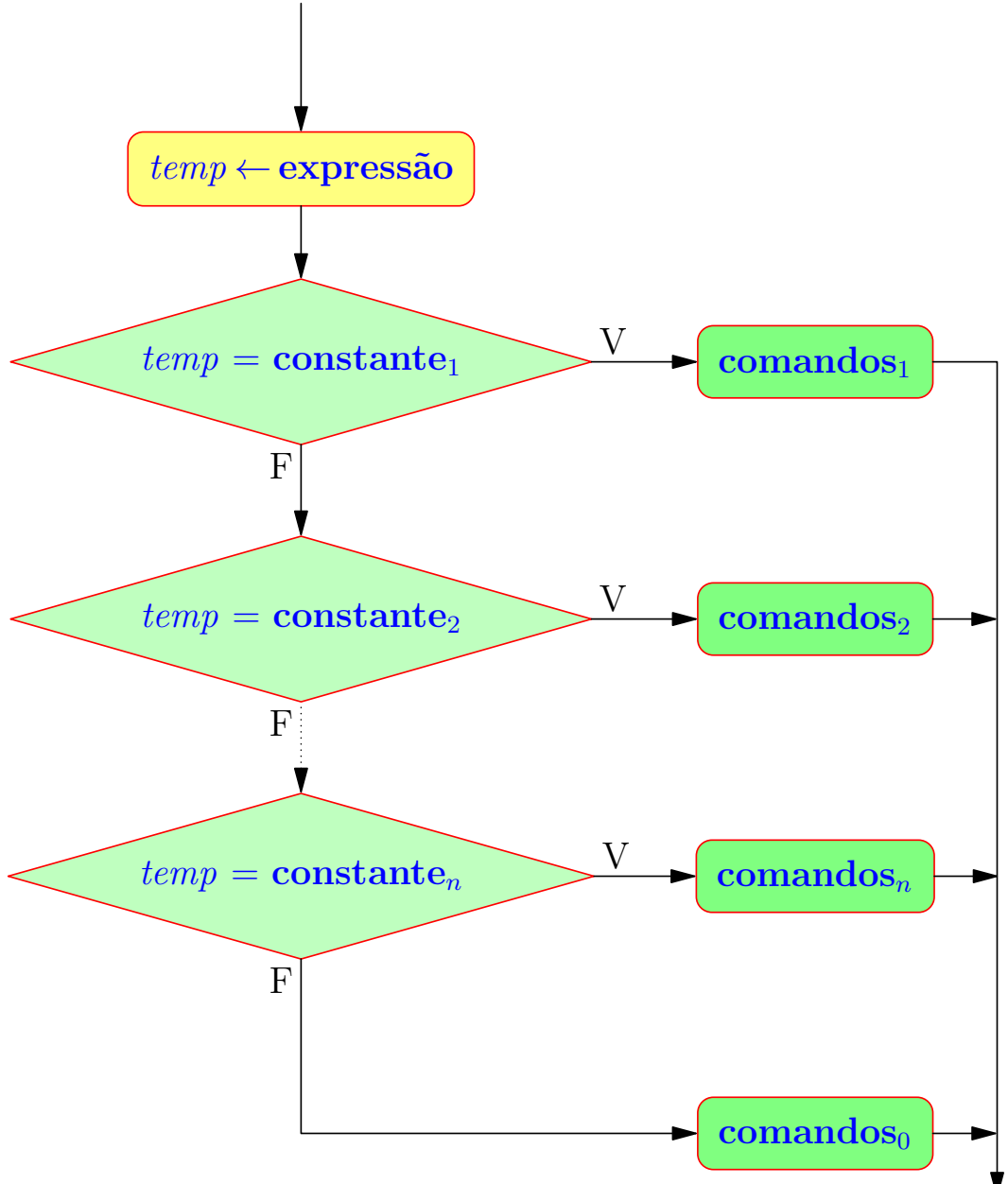
Estrutur de seleção múltipla: comando **`switch`** (cont.)

Execução do `switch`

1. Avalia-se a expressão de controle.
2. Seleciona-se a primeira alternativa cujo rótulo é igual ao valor da expressão de controle, e executa-se a lista de comandos correspondente.

Estrutur de seleção múltipla: comando **switch** (cont.)

```
switch (expressão)
{
case cte1:
    comandos1;
    break;
case cte2:
    comandos2;
    break;
...
case cten:
    comandosn;
    break;
default:
    comando0;
}
```



Estrutur de seleção múltipla: comando **`switch`** (cont.)

- O `switch` executa todos os comandos a partir da alternativa escolhida até o final do `switch` ou até encontrar o comando `break`.
- É necessário usar o comando **`break`** no final de cada alternativa (exceto a última) para evitar que os comandos das alternativas seguintes sejam executados.
- As alternativas são mutuamente exclusivas somente quando cada caso termina com o comando `break`.
- Quando o valor da expressão não coincidir com aqueles especificados nas alternativas e houver a alternativa rotulada com **`default`** então esta alternativa será escolhida.

Estrutur de seleção múltipla: comando **`switch`** (cont.)

- O `switch` só permite comparar expressões com constantes.
- Se precisarmos **comparar** com **variáveis** ou verificar **faixas de valores**, devemos usar o comando `if`.

Estrutur de seleção múltipla: comando **switch** (cont.)

Exemplo: Os dois trechos de programa abaixo são equivalentes:

```
switch (x)
{
case 1:
  printf("x é 1");
  break;
case 2:
  printf("x é 2");
  break;
default:
  printf("x é outro");
}
```

```
if (x == 1)
  printf("x é 1");
else
{
  if (x == 2)
    printf("x é 2");
  else
    printf("x é outro");
}
```

Estrutur de seleção múltipla: comando `switch` (cont.)

Exemplo

Escreva um programa que leia o código de um determinado produto e mostre a sua classificação de acordo com a tabela abaixo.

código	**classificação**
1	Alimento não-perecível
2	Alimento perecível
3	Vestuário
4	Limpeza

Estrutur de seleção múltipla: comando **switch** (cont.)

```
#include <stdio.h>
int main(void)
{   int cod;
    printf("Código do produto: ");
    scanf("%d", &cod);
    switch (cod)
    {
    case 1: printf("Alimento nao-perecível\n");
            break;
    case 2: printf("Alimento perecível\n");
            break;
    case 3: printf("Vestuário\n");
            break;
    case 4: printf("Limpeza\n");
            break;
    default:printf("Produto não existente\n");
    }
    return 0;
}
```

Estrutur de seleção múltipla: comando **`switch`** (cont.)

Exemplo

Dado um caracter, escreva na tela se ele é ou não uma vogal (pode considerar apenas letras minúsculas).

Estrutur de seleção múltipla: comando **switch** (cont.)

```c
#include <stdio.h>
int main(void)
{   char letra;
    printf("Inserir um caracter: ");
    scanf("%c", &letra);
    switch (letra)
    {
    case 'a':
    case 'e':
    case 'i':
    case 'o':
    case 'u': printf("Vogal!!\n");
              break;
    default : printf("Nao é uma vogal\n");
    }
    return 0;
}
```

Exercícios em aula

1. Faça um algoritmo que leia uma variável e some 5 caso seja par ou some 8 caso seja ímpar. Imprimir o resultado desta operação.
2. Encontrar o dobro de um número caso ele seja positivo e o seu triplo caso seja negativo, imprimindo o resultado.
3. O IMC (Indice de Massa Corporal) é um critério da Organização Mundial de Saúde para dar uma indicação sobre a condição de peso de uma pessoa adulta. A fórmula é $IMC = peso/(altura)^2$. Elabore um algoritmo que leia o peso e a altura de um adulto e mostre sua condição de acordo com a tabela abaixo.

IMC em adultos	condição
abaixo de 18.5	abaixo do peso
entre 18.5 e 25	peso normal
entre 25 e 30	acima do peso
acima de 30	obeso

Exercícios propostos

1. Dada uma letra, escreva na tela se essa letra é ou não uma vogal (pode considerar apenas letras minúsculas).
2. Escreva um programa que recebe um operador aritmético e dois operandos e calcule a operação indicada. As operações possíveis são soma(+), subtração(-), multiplicação(*) e divisão(/).

Exercícios propostos (cont.)

3. Elabore um algoritmo que calcule o que deve ser pago por um produto, considerando o preço normal de etiqueta e a escolha da condição de pagamento. Utilize os códigos da tabela a seguir para ler qual a condição de pagamento escolhida e efetuar o cálculo adequado.

Código	Condição de pagamento
1	À vista em dinheiro ou cheque, recebe 10% de desconto
2	À vista no cartão de crédito, recebe 15% de desconto
3	Em duas vezes, preço normal de etiqueta sem juros
4	Em duas vezes, preço normal de etiqueta mais juros de 10%

FIM