Capítulo 4: Condicionais

INF1004 e INF1005 – Programação 1

Pontifícia Universidade Católica
Departamento de Informática

Tomada de Decisão

- Até o momento, todas as instruções eram executadas, uma após a outra. No capítulo 3, vimos que a chamada de uma função transferia a execução para uma outra função, mas dentro do corpo de cada função as instruções eram executadas na ordem em que foram codificadas.
- Em geral, precisamos ter maior controle na sequência de instruções que devem ser executadas.
- É fundamental que seja possível tomar diferentes decisões baseado em **condições** que são avaliadas em tempo de execução ⇨TOMADAS DE DECISÃO!

Tomada de Decisão

- Em C, a tomada de decisão é construída através do comando ***if***:
  - Podemos ter o *if* simples se apenas a expressão booleana retornar "verdadeiro" (true):

```
if ( _expressão_booleana_ ) {
   _bloco_de_comandos_
   ...
}
```

Por exemplo:

```
if (nota < 5.0) {
   printf("Reprovado");
}
```

Tomada de Decisão

  - Como também tratar se a expressão booleana retornar "falso" (false):

```
...
if ( _expressão_booleana_ ) {
   _bloco_de_comandos_1
   ...
}
else {
   _bloco_de_comandos_2
   ...
}
```

Por exemplo:

```
if (nota < 5.0) {
   printf("Reprovado");
}
else {
   printf("Aprovado");
}
```

Tomada de Decisão

- A sintaxe da linguagem C permite ainda que a simples codificação em sequência de comandos *if-else* resulte na construção de seleção exclusiva dentre múltiplas condições:

```
...
if ( _condição_1_ ) {
   _bloco_de_comandos_1
   ...
}
else if ( _condição_2_ ) {
   _bloco_de_comandos_2
   ...
}
else if ( _condição_3_ ) {
   _bloco_de_comandos_3
   ...
}
...
```

Nestas construções, se a expressão booleana correspondente à primeira condição resultar em **verdadeiro**, apenas o primeiro bloco de comandos é executado, e as outras condições não são sequer avaliadas. Senão, se a expressão da segunda condição resultar em **verdadeiro**, apenas o segundo bloco de comandos é executado, e assim por diante.

Tomada de Decisão

Por exemplo:

```
if (nota < 3.0) {
  printf("Reprovado");
}
else if (nota >= 5.0) {
  printf("Aprovado");
}
else {
  printf("Em prova final");
}
```

Expressões Booleanas

- Uma expressão booleana é uma expressão que, quando avaliada, resulta no valor **falso** ou **verdadeiro**. A linguagem C não tem um tipo de dado específico para armazenar valores booleanos:
  - Em C, o valor booleano é representado por um valor inteiro:
    - » 0 significa falso e qualquer outro valor diferente de zero significa verdadeiro.
    - » Em geral, usa-se 1 para representar o valor verdadeiro, e qualquer expressão booleana que resulta em verdadeiro resulta no valor 1.

Expressões Booleanas

- Uma expressão booleana é construída através da utilização de **operadores relacionais**.
  - maior que (>),
  - menor que (<),
  - maior ou igual a (>=),
  - menor ou igual a (<=),
  - diferente de (!=),
  - igual a (==).
- Todos estes operadores comparam **dois operandos**, resultando no valor 0 (falso) ou 1 (verdadeiro).

Expressões Booleanas

- Expressões booleanas também podem ser formadas com **operadores lógicos**.
  - negação (!),
  - conjunção (&&) e
  - disjunção (||).
- Operadores lógicos combinam expressões ou valores booleanos, resultando em um valor booleano (0 ou 1).

Conjunção (AND)

Operando 1	Operando 2	Resultado
Falso	Falso	Falso
Falso	Verdadeiro	Falso
Verdadeiro	Falso	Falso
Verdadeiro	Verdadeiro	Verdadeiro

Disjunção (OR)

Operando 1	Operando 2	Resultado
Falso	Falso	Falso
Falso	Verdadeiro	Verdadeiro
Verdadeiro	Falso	Verdadeiro
Verdadeiro	Verdadeiro	Verdadeiro

Negação (NOT)

Operando	Resultado
Falso	Verdadeiro
Verdadeiro	Falso

Expressões Booleanas

Exemplo 1 (and):

```
...
if (media >= 5.0 && nota1 >= 3.0 && nota2 >=3.0 && nota3 >= 3.0) {
  printf("Aprovado");
}
...
```

Exemplo 2 (or):

```
...
if (media < 5.0 || nota1 < 3.0 || nota2 < 3.0 || nota3 < 3.0) {
  printf("Em prova final");
}
...
```

Exemplo 3 (not):

```
...
if ( !(media < 5.0 || nota1 < 3.0 || nota2 < 3.0 || nota3 < 3.0) )
  {
  printf(“Aprovado");
}
...
```

Exemplo: Expressões Booleanas

- Vamos considerar como exemplo um programa para converter o critério de avaliação de alunos em escolas brasileiras para o critério utilizado em escolas americanas. Nas escolas brasileiras, a avaliação dos alunos é reportada por uma nota que varia de 0 a 10. Nas escolas americanas, a avaliação dos alunos é feita por conceito: A, B, C, D, ou F. Podemos assumir a seguinte equivalência entre os sistemas de avaliação:
  - A (9.0 a 10.0),
  - B (8.0 a 8.9),
  - C (7.0 a 7.9),
  - D (5.0 a 6.9), e
  - F (menor que 5.0)

Exemplo: Expressões Booleanas

```
#include <stdio.h>

int main (void){
  float nota;
  printf("Entre com a nota: ");
  scanf("%f",&nota);
  if (nota >= 9.0) {
    printf("A");
  }
  if (nota >= 8.0 && nota < 9.0) {
    printf("B");
  }
  if (nota >= 7.0 && nota < 8.0) {
    printf("C");
  }
  if (nota >= 5.0 && nota < 7.0) {
    printf("D");
  }
  if (nota < 5.0) {
    printf("F");
  }
  return 0;
}
```

Expressões Booleanas

- É importante salientar a forma como a linguagem C avalia expressões booleanas compostas por operadores lógicos:
  - Por exemplo, na avaliação da expressão (`nota >= 8.0 && nota < 9.0`), o computador primeiro avalia a expressão relacional `nota >= 8.0`. Dependendo do resultado desta expressão, a avaliação da segunda expressão relacional, `nota < 9.0`, pode ser omitida. Isto porque se o resultado da primeira expressão for falso, o resultado da expressão lógica como um todo será falso, independente do valor da segunda expressão, pois estamos usando o operador de conjunção (AND).
  - Assim, como o resultado final não depende do resultado da segunda expressão relacional, esta expressão não é sequer avaliada. Ela só é avaliada se a primeira expressão resultar em verdadeiro.
  - Situação similar ocorre quando usamos o operador de disjunção (OU). Neste caso, se a primeira expressão resultar em verdadeiro, a segunda expressão não é avaliada.

Exemplo: Expressões Booleanas

```
/* solução mais estruturada e mais eficiente */
#include <stdio.h>

int main (void){
  float nota;
  printf("Entre com a nota: ");
  scanf("%f",&nota);
  if (nota >= 9.0){
    printf("A");
  }else if (nota >= 8.0){
    printf("B");
  }else if (nota >= 7.0){
    printf("C");
  }else if (nota >= 5.0){
    printf("D");
  }else{
    printf("F");
  }
  return 0;
}
```

Blocos de Comandos

- Na linguagem C, podemos agrupar comandos em blocos, envolvendo-os com abre e fecha chaves ({...}), como fizemos para delimitar o bloco de comando **if** e **else** nas construções para tomada de decisões.
- Na verdade, podemos criar blocos de comandos em qualquer ponto do programa, bastando envolver comandos com chaves.
- Uma variável declarada dentro de um bloco existe enquanto os comandos do bloco estiverem sendo executados. Quando o bloco chega ao fim, as variáveis declaradas dentro dele deixam de existir.

Blocos de Comandos

- Segundo o padrão C89 da linguagem C, uma variável só pode ser declarada no início de um bloco de comandos (mudou no padrão C99).
- Nas construções do comando **if**, os blocos são importantes para identificar o conjunto de comandos cuja execução está submetida à avaliação da expressão booleana.
- No entanto, se um "bloco de comandos" for constituído por apenas um único comando, as chaves podem ser omitidas.

Bloco de Comandos – Voltando ao Exemplo

```
#include <stdio.h>

int main (void){
  float nota;
  printf("Entre com a nota: ");
  scanf("%f",&nota);
  if (nota >= 9.0)
    printf("A");
  else if (nota >= 8.0)
    printf("B");
  else if (nota >= 7.0)
    printf("C");
  else if (nota >= 5.0)
    printf("D");
  else
    printf("F");
  return 0;
}
```

Exemplo: Cálculo de Raízes em um Equação de 2º. Grau

- Como primeiro exemplo “mais complexo”, vamos discutir a construção de um programa para calcular as raízes de uma equação do segundo grau.
- Sabemos que as raízes de uma equação na forma $ax^2+bx+c=0$ são dadas por: $\frac{-b \pm \sqrt{b^2 - 4*a*c}}{2*a}$
- Este seria um problema de codificação direta de uma expressão matemática se não fosse pelo fato das raízes poderem não existir. Na verdade, a raiz quadrada só é definida para valores positivos.

Exemplo: Cálculo de Raízes em um Equação de 2º. Grau

- Se, dentro de um programa, tentarmos avaliar uma expressão matemática cujo resultado é indefinido, o resultado do programa certamente não será o desejado.
- Isto inclui ações como:
  - tentar extrair a raiz quadrada de um número negativo,
  - calcular o logaritmo de um número negativo,
  - ou mesmo fazer uma divisão por zero.
- Por este motivo, devemos avaliar estas expressões apenas após certificarmos que os operandos são válidos.

Exemplo: Cálculo de Raízes em um Equação de 2º. Grau

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

int main (void){
  double a, b, c;  /* coeficientes */
  double x1, x2;   /* raízes */
  double delta;
  printf("Entre com os coeficientes (a b c):");
  scanf("%lf", &a);
  scanf("%lf", &b);
  scanf("%lf", &c);
  if (a == 0.0) {
    printf("Valor de 'a' nao pode ser zero.");
    return 1;
  }
  delta = b*b - 4*a*c;
  if (delta < 0) {
    printf("Raizes reais inexistentes.");
  }
  else if (delta == 0.0) {
    x1 = -b / (2*a);
    printf("Uma raiz real: %f", x1);
  }
  else {
    delta = sqrt(delta);
    x1 = (-b + delta) / (2*a);
    x2 = (-b - delta) / (2*a);
    printf("Duas raizes reais: %f e %f", x1, x2);
  }
  return 0;
}
```

Exemplo: Cálculo de Volumes

- Vamos construir um programa que permita calcular o volume de vários tipos de objetos diferentes. A idéia é apresentar um menu para o usuário com os tipos de objetos suportados. O usuário então escolhe a opção desejada, entrar com os dados correspondentes e o programa exibe o volume computado.

- Para este nosso exemplo, vamos considerar o cálculo de volume dos seguintes objetos geométricos:
  - caixa de lados a, b e c: *volume* = a*b*c
  - esfera de raio r: *volume* = 4/3*PI*r^3
  - cilindro de raio r e altura h: *volume* = PI*r^2*h
  - cone de raio r e altura h: *volume* = 1/3*PI*r^2*h

Exemplo: Cálculo de Volumes

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

#define PI 3.1415

          void calcula_volume_caixa (void){
            float a, b, c;
            printf("Entre com os lados da caixa:");
            scanf("%f %f %f", &a, &b, &c);
            printf("Volume calculado para caixa:
          %f", a*b*c);
          }
          void calcula_volume_esfera (void){
            float r;
            printf("Entre com o raio da esfera:");
            scanf("%f", &r);
            printf("Volume calculado para esfera: %f",
          4.0/3.0*PI*pow(r,3);
          }
          void calcula_volume_cilindro (void){
            float r, h;
            printf("Entre com o raio e altura do cilindro:");
            scanf("%f %f", &r, &h);
            printf("Volume calculado para o cilindro: %f",
          PI*pow(r,2)*h);
          }
          void calcula_volume_cone (void){
            float r, h;
            printf("Entre com o raio e altura do cone:");
            scanf("%f %f", &r, &h);
            printf("Volume calculado para o cone: %f",
          PI*r*r*h/3.0);
          }
```

Exemplo: Cálculo de Volumes (continuação)

```
int main (void){
  int escolha;
  /* exibe menu na tela */
  printf("Escolha uma opcao:\n");
  printf("1 - Caixa\n");
  printf("2 - Esfera\n");
  printf("3 - Cilindro\n");
  printf("4 - Cone\n");
  /* lê opção escolhida e chama função correspondente */
  scanf("%d", &escolha);
  if (escolha == 1) {
    calcula_volume_caixa();
  }
  else if (escolha == 2) {
    calcula_volume_esfera();
  }
  else if (escolha == 3) {
    calcula_volume_cilindro();
  }
  else if (escolha == 4) {
    calcula_volume_cone();
  }
  else {
    printf("Opcao invalida.");
  }
  return 0;
}
```

A instrução `switch`

- A instrução switch é usada quando queremos testar várias possibilidades de fluxo de código mas não queremos usar vários “else if”. Desta forma, cada possibilidade é testada em um bloco case.
- Veja um exemplo:

```
#include <stdio.h>
int main(void) {
  int valor = 4;
  switch(valor){
    case 0:
      printf("Valor e igual a 0");
      break;
    case 1:
      printf("Valor e igual a 1");
       break;
    case 2:
      printf("Valor é igual a 2");
      break;
    default:
      printf("Nenhuma das anteriores");
  }
  printf("\n\n");
  return 0;
}
```

Veja que, se nenhuma das condições testadas em um bloco case for satisfatória, a parte default da instrução switch será executada.