UFRN
UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE

Linguagem de Programação

Variáveis, Controle de FLuxo e Comandos de Seleção

ECT2303

helton.maia@ufrn.br

Conteúdo da Aula:

- Variáveis, Operadores e Expressões
- Comandos de Entrada e Saída
- Comandos de Decisão

Comandos de Entrada e Saída

Comandos de entrada e saída são utilizados para ler dados do usuário ou exibir informações na tela do computador.

- "cin": Ler dados de entrada do usuário, como valores numéricos ou strings. Exemplo: "**int x; cin >> x;**". O operador ">>" é usado para inserir os dados digitados pelo usuário na variável "x".
- "cout": Exibir informações na tela do computador, como mensagens ou valores de variáveis. Exemplo: "**cout << y;**". O operador "<<" é usado para inserir o valor da variável "y" na saída padrão.
- "endl": Para inserir uma quebra de linha na saída padrão. Exemplo: exibir uma mensagem na tela seguida por uma quebra de linha. "**cout << "Olá, ECT!**" << endl;".

Variáveis, Operadores e Expressões

Os tipos de dados podem ser divididos em duas categorias:

Tipos primitivos: incluem inteiros (int), caracteres (char), números de ponto flutuante (float e double), booleanos (bool) e vários outros.
Tipos definidos pelo usuário: são criados pelo programador para representar entidades mais complexas, como classes, estruturas e enums.

Variáveis, Operadores e Expressões

Variáveis: Uma variável é um local na memória que é reservado para armazenar um valor.

Você precisa declarar uma variável antes de usá-la, especificando o tipo de dados e o nome da variável.

Por exemplo:

```
int idade = 21;
```

Variáveis, Operadores e Expressões

Constantes são valores que não podem ser alterados durante a execução do programa.

Podem ser definidas utilizando a palavra-chave "const". Por exemplo, para declarar uma constante do tipo float chamada "pi" com valor 3.14, você pode escrever:

```
const float pi = 3.14;
```

Variáveis, Operadores e Expressões

Operadores são símbolos especiais que realizam operações em variáveis ou valores.

Alguns tipos de operadores:

- Aritméticos (+, -, *, /);
- Relacionais (==, !=, <, >);
- Lógicos (&&, ||, !);
- Operadores unários (sizeof, ++, --);
- Outros (operador de atribuição = e o operador ternário ?:)

Os operadores em C++ têm precedência, o que determina a ordem em que as operações são realizadas.

Operadores Unários

- Operador de incremento (++): aumenta o valor de uma variável em 1.
- Operador de decremento (--): diminui o valor de uma variável em 1.

Formas de uso: pré-incremento / pré-decremento e pós-incremento / pós-decremento.

Obs: O pré-incremento e pré-decremento ocorrem antes que a variável seja usada na expressão, enquanto o pós-incremento e pós-decremento ocorrem após a variável ser usada na expressão.

(Exemplo)

Operadores

unários:

```
int main() {
    int x = 5;

    // pré-incremento
    ++x;
    cout << "Valor de x após pré-incremento: " << x << endl;

    // pré-decremento
    --x;
    cout << "Valor de x após pré-decremento: " << x << endl;

    // pós-incremento
    x++;
    cout << "Valor de x após pós-incremento: " << x << endl;

    // pós-decremento
    x--;
    cout << "Valor de x após pós-decremento: " << x << endl;

    return 0;}
```

Operador sizeof

Utilizado para obter o tamanho em bytes de um tipo de dado ou de uma.

```
sizeof (tipo_de_dado)
sizeof (variavel)
```

Exemplo:

```
#include <iostream>
using namespace std;

int main() {
    int a;
    cout << "Tamanho em bytes de um inteiro: "
    << sizeof(int) << endl;
    cout << "Tamanho em bytes da variável: "
    << sizeof(a) << endl;
    return 0;
}
```

Comandos de Decisão

Comandos de decisão permitem executar diferentes blocos de código com base em determinadas condições.

"if": Permite executar um bloco de código se uma determinada condição for verdadeira. Por exemplo, verificar se um número é positivo e exibir uma mensagem na tela.

```
int x;
cin >> x;
if (x > 0){
  cout << "O número é positivo.";
}
```

"if-else": Permite executar um bloco de código se uma determinada condição for verdadeira e outro bloco de código se a condição for falsa.

```
int x;
cin >> x;
if (x > 0) {
    cout << "Positivo."; }
else {cout << "Zero ou
Negativo."; }
```

Comandos de Decisão Encadeada

Para avaliar várias condições diferentes, temos:

```
if (condição1) {
    // Condição1 for verdadeira
} else if (condição2) {
    // Condição1 for falsa e a condição2 for verdadeira
} else if (condição3) {
    // Condição1 e a condição2 forem falsas e a condição3 for
    verdadeira
} else {
    // código a ser executado se nenhuma das condições acima for
    verdadeira
}
```

Comandos de Decisão Encadeada (exemplo)

```
int main() {
    int num;
    cout << "Digite um número inteiro: ";
    cin >> num;
    if (num > 0) {
        cout << "O número " << num << " é positivo." << endl;
    } else if (num < 0) {
        cout << "O número " << num << " é negativo." << endl;
    } else {
        cout << "O número " << num << " é zero." << endl;
    }
    return 0;
}
```

Comandos de Decisão

"switch": Permite executar diferentes blocos de código com base em diferentes valores de uma variável. Por exemplo, para verificar o valor de uma variável chamada "opcao" e executar um bloco de código correspondente.

```
int opcao;
cin >> opcao;
switch (opcao) {
   case 1: cout << "Opção 1 selecionada.";
      break;
   case 2: cout << "Opção 2 selecionada.";
      break;
   default: cout << "Opção inválida.";
}
```

Comandos de Decisão switch (exemplo)

```
int main() {
    int opcao;
    cout << "Digite um número de 1 a 3: ";
    cin >> opcao;
    switch (opcao) {
        case 1:
            cout << "Você escolheu a opção 1." << endl;
            break;
        case 2:
            cout << "Você escolheu a opção 2." << endl;
            break;
        case 3:
            cout << "Você escolheu a opção 3." << endl;
            break;
        default:
            cout << "Opção inválida." << endl;
            break;
    }
    return 0;
}
```

O usuário é solicitado a digitar um número de 1 a 3. Em seguida, um comando switch é usado para exibir uma mensagem correspondente à opção escolhida.

Obs: A utilização do ***break*** evita a execução dos casos subsequentes;

Comparando Comandos de decisão

Exemplo ***switch***

```
switch (x) {
  case 1:
    cout << "x vale: 1";
    break;
  case 2:
    cout << "x vale: 2";
    break;
  default:
    cout << "valor
desconhecido;
  }
```

Exemplo ***if-else*** equivalente

```
if (x == 1) {
  cout << "x vale 1";
}
else if (x == 2) {
  cout << "x vale 2";
}
else {
  cout << "valor de x
desconhecido";
}
```

Operador Ternário ?:

forma compacta de se escrever um comando de decisão que retorna um valor com base em uma condição.

```
condicao ? valor_se_verdadeiro : valor_se_falso;
```

- A condição é avaliada primeiro;
- Se a condição for verdadeira, o valor da expressão é o "valor_se_verdadeiro";
- Se a condição for falsa, o valor da expressão é o "valor_se_falso".

Operador Ternário ?: (exemplo)

```
#include <iostream>
using namespace std;

int main() {
    int a = 10;
    int b = 5;
    int maior = (a > b) ? a : b;
    cout << "O maior valor é: " << maior;
    return 0;
}
```

Se "a" for maior do que "b", a expressão "(a > b) ? a : b" retorna o valor de "a", caso contrário, retorna o valor de "b".

Exercício de aprendizagem

Seu programa deve solicitar que o usuário digite sua idade. Em seguida, usa uma série de comandos if-else devem ser utilizados para determinar se o usuário é menor de idade, adulto ou idoso, com base na idade inserida.

Solução:

```
#include <iostream>
using namespace std;

int main() {
    int idade;
    cout << "Digite sua idade: ";
    cin >> idade;

    if (idade < 18) {
        cout << "Você é menor de idade." << endl;
    } else if (idade >= 18 && idade < 60) {
        cout << "Você é adulto." << endl;
    } else {
        cout << "Você é idoso." << endl;
    }

    return 0;
}
```

Perguntas ?