Linguagem C++

Introdução, identificadores, tipos de dados

Prof. Bruno E. G. Gomes
IFRN

Linguagem de Programação

- Constituída por símbolos e por regras para combinar esses símbolos
  - Símbolos: operadores, palavras-chave, identificadores, números, etc.
  - Regras:
    - sintáticas (forma)
    - semânticas (significado)
- É uma linguagem formal
  - Sintaxe e semântica devem ser obedecidas com rigor
  - Não admite interpretações ambíguas, incorretas

Ling. de alto-nível X Ling. de baixo nível

- **Linguagem de baixo nível (Assembly)**: Mais próxima da linguagem de um microprocessador específico
  - O programador manipula diretamente instruções da máquina e acesso a registradores

- **Linguagem de alto nível**: Possui maior abstração quanto a detalhes específicos de *hardware*
  - O programador lida com tipos de dados (abstratos e/ou primitivos), estruturas de dados, etc.

Exemplo: programa em Assembly e C++

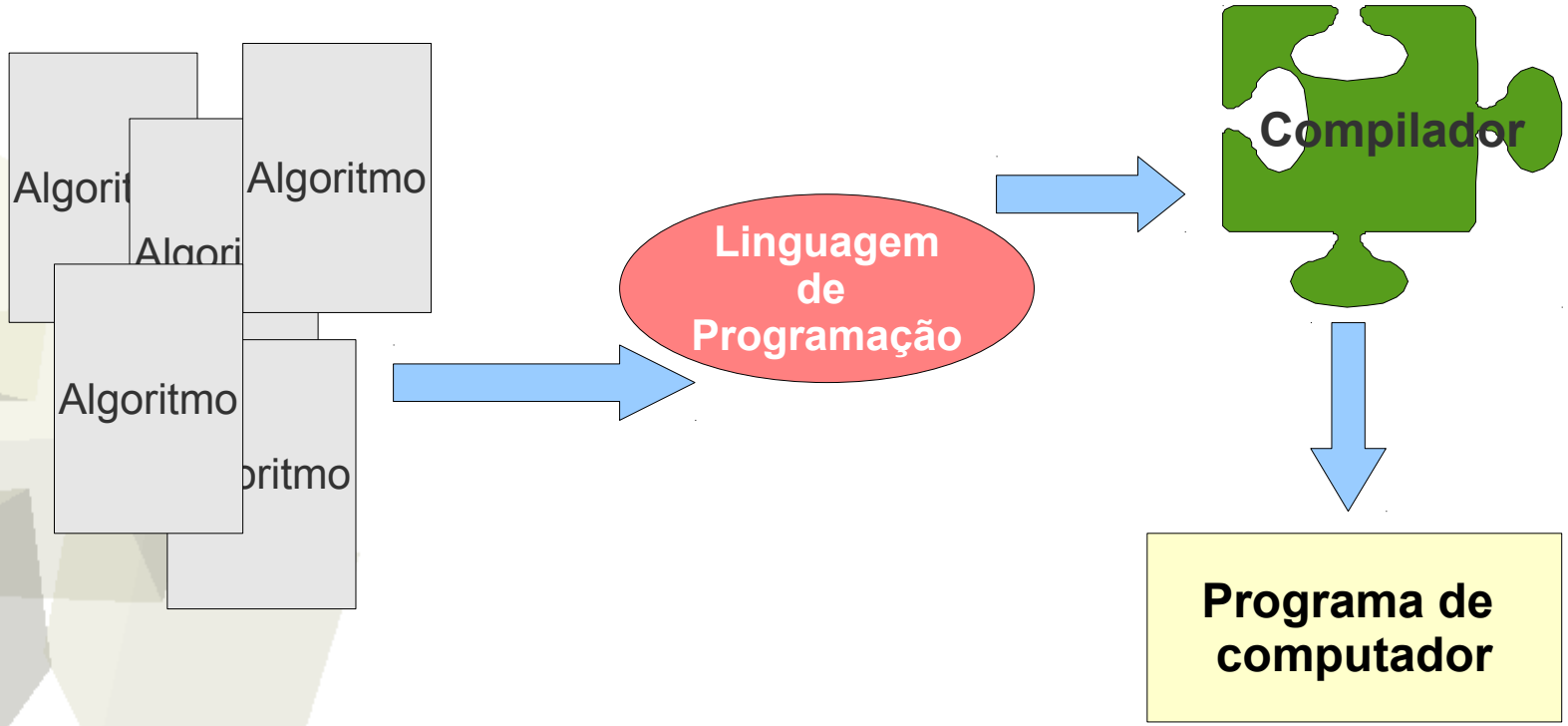
```
MOV CX, 0
IN  AX, PORTA
MOV DX, AX
LABEL:
INC CX
MOV AX, DX
MUL CX
OUT AX, PORTA
CMP CX,10
JNE LABEL
```

```
...
cin >> num;

for (n = 1; n <= 10; n++) {
   cout << num * n;
}
...
```

Criação de um programa de computador

- O *compilador* transforma o código em uma linguagem de programação em uma linguagem de mais baixo nível
  - Com o auxílio de outras ferramentas, essa linguagem é finalmente transformada em um programa



Linguagem C++

- Linguagem de programação de propósito geral desenvolvida no início dos anos 80 por Bjarne Stroustrup
  - Influenciada principalmente pela linguagem C (Dennis Ritchie)
  - Ainda hoje é uma linguagem bastante popular, principalmente para aplicações que requerem alto-desempenho

- Permite programar em dois estilos:
  - Estruturado (uso de funções)
  - Orientado a objetos (uso de Classes)

Desenvolvendo programas em C++

- O desenvolvimento em C++ é composto basicamente pelas etapas abaixo:
  1) Codificação do programa em C++ utilizando um editor de texto.
     → O arquivo deve ter extensão ".cpp" ou ".cc", ex.: *teste.cpp*, *dias.cc*

  2) Compilação/linkedição do programa (utilizaremos o compilador GNU-G++)
     → *g++ -o <nome_programa> <nome_arquivo_fonte>.cpp*

  3) Depuração (opcional) : o depurador permite examinar o programa em busca de erros que são difíceis de localizar

  4) Execução do programa
     → **./**<nome_programa>

Linguagem C++ estruturada

- Uso de blocos de código.
  - grupo de comandos (instruções) do programa que é tratado como uma unidade lógica
  - um dos comandos não pode ser executado sem que o outro também não o seja
    - Todo comando em C ou é um comando simples ou está inserido em um bloco.
    - Um bloco é formado por comandos entre chaves **{ }**
- Uso de funções para modularizar o código.
  - As funções são os blocos onde toda atividade do programa ocorre.
  - Possui algumas construções de laços e estruturas de decisão para controle do fluxo do programa (while, do-while, if, etc.).

Estrutura básica de um programa em C

```
declarações globais (variáveis, prótotipos de função, etc.)

<tipo_devolvido> main(<parâmetros>) {
        <comandos>

}

<tipo_devolvido> função_1(<parâmetros>) {
        <comandos>

}       ...

<tipo_devolvido> função_N(<parâmetros>) {
        declarações locais (variáveis, constantes)

}
```

Um pequeno programa em C++

```
/* Programa que imprime uma
   mensagem na tela do computador */


#include <iostream>


int main() {

    std::cout << “Primeiro programa em
                  C”;


    return 0;
}
```

O texto entre /* e */ é um comentário. '//' definem comentários em uma única inha.

Inclui funções de bibliotecas externas. No caso “iostream”, que possui funções de entrada e saída

A função main inicia a execução do programa

Chamada à função **cout**, utilizada para imprimir a mensagem na tela

Retorno da função **main**

Comandos (instruções)

- Um programa em C++ é uma sequência de instruções em um bloco de comandos
  - todo comando deve terminar com **;** (ponto-e-vírgula)
- Bloco de comandos
  - Uma ou mais instruções entre chaves.
  - **{** (chave aberta) - início de um bloco de comandos.
  - **}** (chave fechada) – fim de um bloco de comandos.

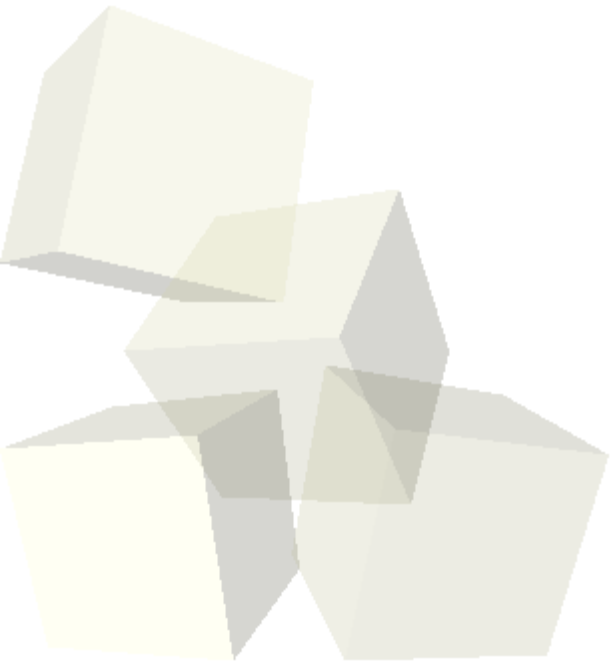
Identificadores em C++

- São *nomes* de variáveis, constantes, funções ou de qualquer outra construção definida pelo usuário
- Pode ter 1 ou mais caracteres:
  - O primeiro caractere deve ser uma letra ou sublinhado (_)
  - Os demais caracteres devem ser letras, números ou sublinhados
- Em C++, Count, count e COunT são identificadores distintos

Identificadores

- Exemplos:
  - (corretos) count, teste1, nomes, nome_2, high_balance
  - (incorretos) 1count, hi!there, pessoa...nome

- Também não podem ser utilizados como identificadores:
  - nomes de funções da biblioteca (cout, cin, etc.)
  - nomes de palavras reservadas da linguagem (if, continue, while, do, etc.)



Palavras Reservadas

- São identificadores que são parte da definição da linguagem e que portanto não podem ser utilizados como identificador pelo programador

- As palavras reservadas de C++ são:

asm do if public this auto double inline register throw
break else int return try case enum long short typedef catch
explicit mutable signed union char extern namespace
sizeof unsigned class float new static using const for
operator struct virtual continue friend private switch void
default goto protected template volatile delete while

Tipos de dados primitivos (básicos)

- Há quatro tipos primitivos de dados em C:
  - **char** (caractere),
  - **int** (inteiro),
  - **float** e **double** (reais)
- Há ainda a palavra “**void**” para designar "falta de tipo"
- O tipo **char** armazena caracteres ou inteiros pequenos
- Os tipos **float** e **double:**
  - utilizados para se operar com números muito grandes ou
  - quando se precisa trabalhar com números reais

Tipos primitivos

Tipo	Tam. em bytes	Faixa de valores
char	1	-128 a 127
int	4	-2147483648 a 2147483647
float	4	3.4e-38 a 3.4e+38
double	8	1.7e-308 a 1.7e+308

Modificadores de tipos

- Podem ser utilizadas para modificar a faixa de valores de cada tipo, exceto para **void**

- **signed**
  - tipo com sinal: -TAM_MÍNIMO .. TAM_MÁXIMO

- **unsigned**
  - faixa do tipo sem sinal: 0 .. TAM_MÁXIMO
  - exemplo: aumentar a faixa de um inteiro para números positivos
  - não pode ser aplicado a **float** e **double**.

Modificadores de tipos

- **short**
  - Diminui a faixa de valores do tipo
  - Ex: **short int**

- **long**
  - Aumenta a faixa de valores para o tipo
  - **long float** não existe (seria o mesmo que double)

Alguns tipos primitivos modificados

Tipo	Tamanho (bytes)	Intervalo
unsigned char	1	0 a 255
unsigned int	4	0 a 4294967295
short int	2	-32768 a 32767
unsigned short int	2	0 a 65535
long int	4	-2.147.483.648 a 2.147.483 647
unsigned long int	4	0 a 4.294.967.295
long double	10	3.4e-4932 a 1.1e4932