Instituto Federal do Rio Grande do Norte (IFRN)
***Campus* Pau dos Ferros**

Introdução a Programação

Prof. Dr. Aluisio Igor Rêgo Fontes

Contato: aluisio.rego@ifrn.edu.br

Pau dos Ferros, abril de 2018.

Programa

- **Programa pode ser definido como uma série de instruções que indicam como o computador irá realizar serviços;**
- **O programa deve definir a ordem em que as instruções devem ser executadas pelo computador;**
- **Um computador, geralmente, possui muitos programas, que podem estar sendo executados ao mesmo tempo.**

Algoritmo X Programa

- **Algoritmos são abstratos:**
  - **Independe de máquina e de linguagem de programação;**
  - **Pode ser representado de várias formas;**
  - **Pode ser feito por uma máquina ou por um humano.**

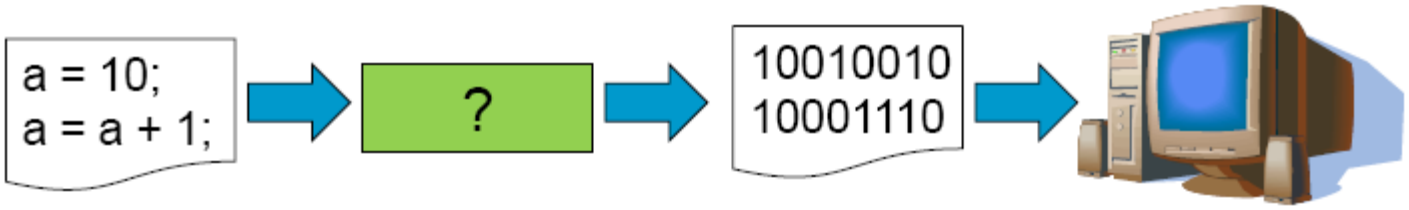
- **Programa é uma implementação real de um algoritmo utilizando uma linguagem de programação:**
  - **É executado por um computador.**

Linguagens de Programação

❖**Os programas tem que ser escritos em uma linguagem de programação:**

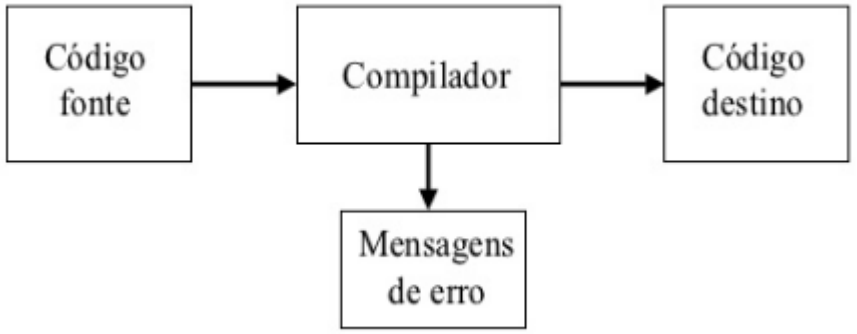
❖Uma linguagem que pode ser entendida pelo computador.

❖Uma linguagem que pode ser **traduzida para a linguagem** entendida pelo computador.



Introdução - Compilador

Compilador: Software básico de computador capaz de traduzir uma linguagem de alto nível (código fontes) em uma linguagem de baixo nível (executável)



Linguagens de Programação

❖As **linguagens de programação são** conjuntos de termos e regras que permitem a formulação de instruções para o computador. Geralmente essas instruções são escritas em formato de texto (em inglês na maioria das vezes);

❖A primeira e mais primitiva linguagem de computador é a própria linguagem de máquina (**0's e 1's**);

❖Programar era difícil e cansativo;

❖Os programas eram difíceis de serem entendidos por outros programadores.

Níveis de Abstração de Linguagens

- **Cada tipo de CPU tem sua linguagem de máquina específica:**
  - Instruções codificadas em binário;
  - Dependente de máquina.

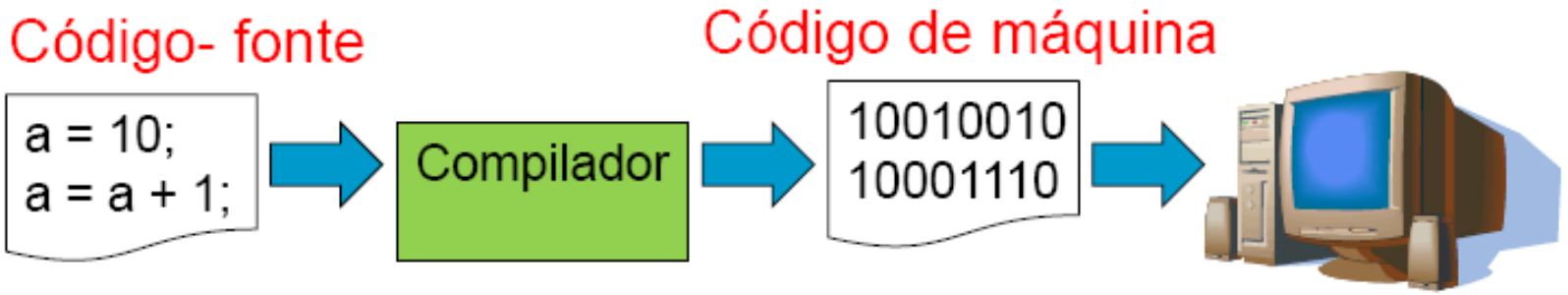
- **Linguagem Assembly é dependente de máquina, porém utiliza palavras reservadas para codificar instruções (mnemônicos);**

- **Outros níveis são independentes de máquina e facilitam a leitura e escrita de programas por parte do ser humano:**
  - Complexidade atual de programas exigem cada vez mais o emprego destas linguagens.

Como o computador entende um programa?

- Cada tipo de CPU executa apenas uma linguagem de máquina particular;
- Deve-se traduzir um programa para a linguagem de máquina;
- Um **compilador é um programa que traduz um** programa escrito (código fonte) em uma determinada linguagem de programação para outra linguagem (linguagem destino);
  - Se a linguagem destino for a de máquina, o programa pode depois de compilado, ser executado.
- Um **interpretador é um programa que traduz instrução** por instrução de um programa em linguagem de máquina e imediatamente executa a instrução.

Compilação


Código- fonte
a = 10;
a = a + 1;
Compilador
Código de máquina
10010010
10001110

Sintaxe e Semântica

- Uma linguagem de programação define as palavras e símbolos que se pode usar para escrever um programa;
- Uma linguagem de programação emprega um conjunto de regras (**sintaxe) que estabelece como palavras e** símbolos podem ser agrupados de maneira a formar instruções válidas de um programa;
- A **semântica de uma instrução define o significado** desta instrução no programa;
- Um programa que é sintaticamente correto não é necessariamente logicamente (semanticamente) correto.

Linguagens de Programação

❖**Foram desenvolvidas diversas linguagens de programação:**

- ❖FORTRAN
- ❖ALGOL
- ❖COBOL
- ❖PASCAL
- ❖BASIC
- ❖ADA
- ❖C e C++
- ❖JAVA
- ❖Pyton
- ❖PHP
- ❖...

❖**Estas novas linguagens foram afastando cada vez mais o programador do nível de máquina.**

Conectivo "e": *Conjunção*

- Proposições compostas em que está presente o conectivo **"e";**
- Simbolicamente representado por "∧".
- A sentença:

*"Marcos é médico **e** Maria é estudante"*

... pode ser representada apenas por: **p** ∧ **q.** Onde: p = *Marcos é médico* e q = *Maria é estudante*.

- Como se revela o **valor lógico** de uma *proposição conjuntiva*? Da seguinte forma: **uma conjunção só será verdadeira, se ambas as proposições componentes forem também verdadeiras**.

Conectivo "e": *Conjunção*

p	q	p∧q
V	V	V
V	F	F
F	V	F
F	F	F

- Uma maneira de assimilar bem essa informação seria pensarmos nas sentenças simples como promessas de um pai a um filho: *"**eu te darei uma bola E te darei uma bicicleta**"*. Ora, pergunte a qualquer criança! Ela vai entender que a promessa é para os dois presentes. Caso o pai não dê nenhum presente, ou dê apenas um deles, a promessa não terá sido cumprida. Terá sido falsa! No entanto, a promessa será verdadeira se as duas partes forem também verdadeiras!

Conectivo "ou": *Disjunção*

- Proposições compostas em que está presente o conectivo **"ou";**
- Simbolicamente representado por "**V**".
- A sentença:

*"Marcos é médico **ou** Maria é estudante"*

... pode ser representada apenas por: **p V q.** Onde: p = *Marcos é médico* e q = *Maria é estudante*.

- Como se revela o **valor lógico** de uma *proposição disjuntiva*?

Conectivo "ou": *Disjunção*

p	q	pvq
V	V	V
V	F	V
F	V	V
F	F	F

❖ Lembremos da promessa de um pai a um filho: *"**eu te darei uma bola OU te darei uma bicicleta**"*. Neste caso, a criança já sabe, de antemão, que a promessa é por apenas um dos presentes! Bola **ou** bicicleta! Ganhando de presente apenas um deles, a promessa do pai *já valeu*! Já foi verdadeira! E se o pai for *abastado* e resolver dar os dois presentes? Pense na cara do menino! Feliz ou triste? Felicíssimo! A promessa foi mais do que cumprida. Só haverá um caso, todavia, em que a bendita promessa não se cumprirá: se o pai esquecer o presente, e não der nem a bola e nem a bicicleta. Terá sido falsa toda a *disjunção*.

Operador Unário: "*não*"

- **Como negar uma proposição?**
  - Basta acrescentar a palavra "*não*" antes da sentença.
- **E se a sentença original já for uma negativa?**
  - Basta excluir a palavra *"não".*

p	~p
V	F
F	V

Operador Unário: "*não*"

- **Como negar uma proposição?**
  - Basta acrescentar a palavra "*não*" antes da sentença.
- **E se a sentença original já for uma negativa?**
  - Basta excluir a palavra *"não".*

p	~p
V	F
F	V

Exercício

Qual o resultado das seguintes expressões lógicas:

a) V e (V ou F)
b) V e não (V ou F)
c) (F ou V) e não (F)
d) (V e V) ou F
f) (F ou V) e F

Exercício

Qual o resultado das seguintes operações

1) 2 * 3 + 5

2) 2 + 6 / 2

3) 5 * 8 + 4 / 2

4) 6 * 9 – 15 / 3

5) 4 * 3 / 6 – 2

6) (2+3)*5

7) (2-3)*(5+2)*5*2

Precedência dos operadores aritméticos

Precedência	Descrição
1	parênteses
2	pot, raiz
3	*, / , resto
4	+, -

Operadores Aritméticos

pot: potência entre dois números.

- **Ex:** 10 **pot** 2 = 100 / 6 **pot** 3 = 216

raiz: operação de radiciação entre dois números.

- **Ex:** 9 **raiz** 2 = 3 / 216 **raiz** 3 = 6

div: resultado inteiro de uma divisão.

- **Ex:** 10 **div** 4 = 2 / 22 **div** 6 = 3

resto: resto da divisão inteira de dois números inteiros.

- Ex: 10 **resto** 4 = 2 / 22 **resto** 6 = 4

Estrutura de um programa

- Todas as linhas começando com 2 barras (//) serão comentários e não terão nenhum efeito no comportamento do programa
- O programador deve inserir pequenas observações ou explicações dentro do código
- #include<iostream>
  - Linha que começa com # são directivas
  - Biblioteca de C++ para entrada e saída básica
- Using namespace std
  - Todas os elementos de bibliotecas padrão C++ são declaradas dentro de um namespace chamado std

Estrutura de um programa

- Int main()
  - Início da função principal
  - É sempre a primeira função a ser executada
  - A função principal é o ponto de partida para a execução de todos os programas C++
  - É obrigatório ter a int main ou void main
- cout << “Hello World! \n”
  - Cout representa a saída padrão em C++;
  - \n quebra uma linha na saída do terminal

Compilador

1. Entre na pasta pelo terminal onde se encontra seu código .cpp
    1. Cd (caminho da pasta)
2. g++ seucodigo.cpp -o seuprograma
3. ./seuprograma

G++ seucodigo.cpp -> Compile o código
-o seuprograma -> gera o arquivo binário (executável)

Segundo código

```
/* my second program in C++
with more comments */
#include <iostream>
using namespace std;
int main ()
{
    cout << "Hello World! "; // prints Hello World!
    cout << "I'm a C++ program"; // prints I'm a C++
    return 0;
}
```

Variáveis e tipos de dados

Espaço da memória utilizado para armazenar dados

Tipos em C++

Tipo	Tamanho	Valores Possíveis
bool	1 byte	true e false
byte	1 byte	0 a 255
sbyte	1 byte	-128 a 127
short	2 bytes	-32768 a 32767
ushort	2 bytes	0 a 65535
int	4 bytes	-2147483648 a 2147483647
uint	4 bytes	0 to 4294967295
long	8 bytes	–9223372036854775808L to 9223372036854775807L
ulong	8 bytes	0 a 18446744073709551615
float	4 bytes	Números até 10 elevado a 38. Exemplo: 10.0f, 12.5f
double	8 bytes	Números até 10 elevado a 308. Exemplo: 10.0, 12.33
decimal	16 bytes	números com até 28 casas decimais. Exemplo 10.991m, 33.333m
char	2 bytes	Caracteres delimitados por aspas simples. Exemplo: 'a', 'ç', 'o'

Exemplo - Soma

```
#include <iostream>
using namespace std;
int main ()
{
   // declaring variables:
   int a, b, result;
   cin >> a;
   cin >> b;

   result = a+b;

   cout << “O valor da soma é: ” << result;

   return 0;

}
```

Exemplo - Soma

```
#include <iostream>
using namespace std;
int main ()
{
   // declaring variables:
   int a, b;
   cin >> a;
   cin >> b;

   result = a+b;

   cout << “O valor da soma é: ” << result;

   return 0;

}
```

Exemplo – Operações matemáticas

```
1  #include <iostream>
2  #include <cmath>
3  using namespace std;
4  int main(){
5
6      float a,b,c,d,e,f,g;
7
8      cin >> a >> b;
9      c = a + b;
10     d = a - b;
11     e = a * b;
12     f = a / b;
13     g = pow(a,b);
14
15     cout << "A + B = C -> " << a << " + " << b << " = " << c << endl;
16     cout << "A - B = D -> " << a << " - " << b << " = " << d << endl;
17     cout << "A * B = E -> " << a << " * " << b << " = " << e << endl;
18     cout << "A / B = F -> " << a << " / " << b << " = " << f << endl;
19     cout << "A ^ B = G -> " << a << " ^ " << b << " = " << g << endl;
```

Estrutura condicional Simples

```
if(condição)
{           comando1;
            comando2;
            comando3;
 }
//Os comandos 1, 2 e 3 só serão executados se a condição for verdadeira.


 if(condição)
  {
            comando1;
            comando2;
  }
 else
 {
            comando3;
             comando4;
  } //Se a condição for verdadeira, o comando1 e o comando2 serão
 executados, caso contrário, o comando3 e o comando4 serão executados
```

Operadores Condicionais ou Relacionais em C++

Símbolo	Nome do Operador	Exemplo	Significado
>	Maior que	x > y	x é maior que y?
>=	Maior ou igual	x >= y	x é maior ou igual a y ?
<	Menor que	x < y	x é menor que y?
<=	Menor ou igual	x <= y	x é menor ou igual a y ?
==	Igualdade	x == y	x é igual a y?
!=	Diferente de	x != y	x é diferente de y?

```
if(x == 3)
        cout << “Número igual a 3”;

if(x != 3)
        cout << “Número diferente de 3”;

if(x >= 3)
        cout << “Número maior ou igual a 3”;
```

Operadores Lógicos

Tabela E	Tabela OU
V e V = V	V ou V = V
V e F = F	V ou F = V
F e V = F	F ou V = V
F e F = F	F ou F = F

```
if(x == 3)
        cout<<"Número igual a 3";

if(x > 5 && x<10)
        cout<<"Número entre 5 e 10";

if(x == 5 && y == 2) || (y == 3)
        cout<<"x é igual a 5 e y é igual a 2, ou y é igual a 3";

if(x == 5 && (y == 2 || y == 3))
        cout<<"x é igual a 5, e y é igual a 2 ou y é igual a 3";
```

Exercício Resolvido

Faça um algoritmo que receba duas notas de aluno, calcule e mostre a média aritmética das notas e a mensagem de “Aprovado”, se a média obtida for maior ou igual a 6.0 ou, caso contrário, “Reprovado”

```
#include <iostream>
Using namespace std;
void main () {
        float n1, n2, media;
        cout << " \n Digite a nota 1:";
        cin >> n1;
        cout << "\n Digite a nota 2:";
        cin >> n2;
        media = (n1+n2)/2;
        cout << "O valor da média:" << media;

        if(media >=6)
                cout << "\n APROVADO!";
        else
                cout << "\n REPROVADO!";
}
```