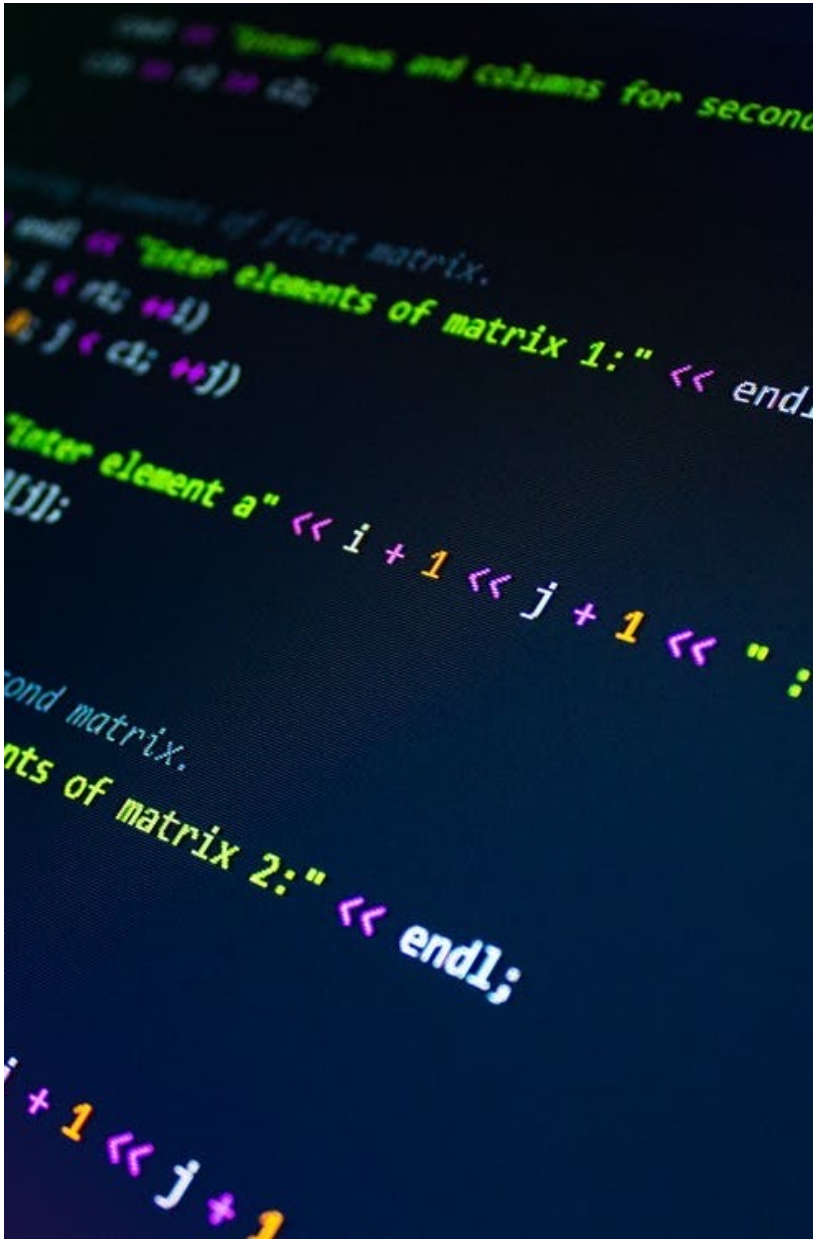
Insper

Supercomputação



Aulas 02/03

- Programação em C++

Algoritmo

- Sequência finita de passos executáveis que resolve um problema

Algoritmo

- Sequência finita de passos executáveis que resolve um problema

- Implementar um algoritmo:
    - Transformação de um algoritmo em um programa executável

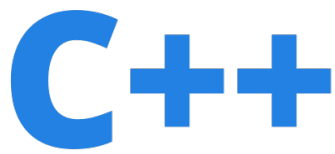
Quanto tempo um programa demora?

- Algoritmo
  - Complexidade computacional (classe de algoritmos)
  - Estruturas de Dados
- Implementação
  - Medido em segundos, para uma certa entrada
  - Tecnologias usadas (linguagens de programação, bibliotecas)
  - Hardware utilizado (clock de CPU, RAM, cache, # de núcleos, etc.)

Quanto tempo um programa demora?

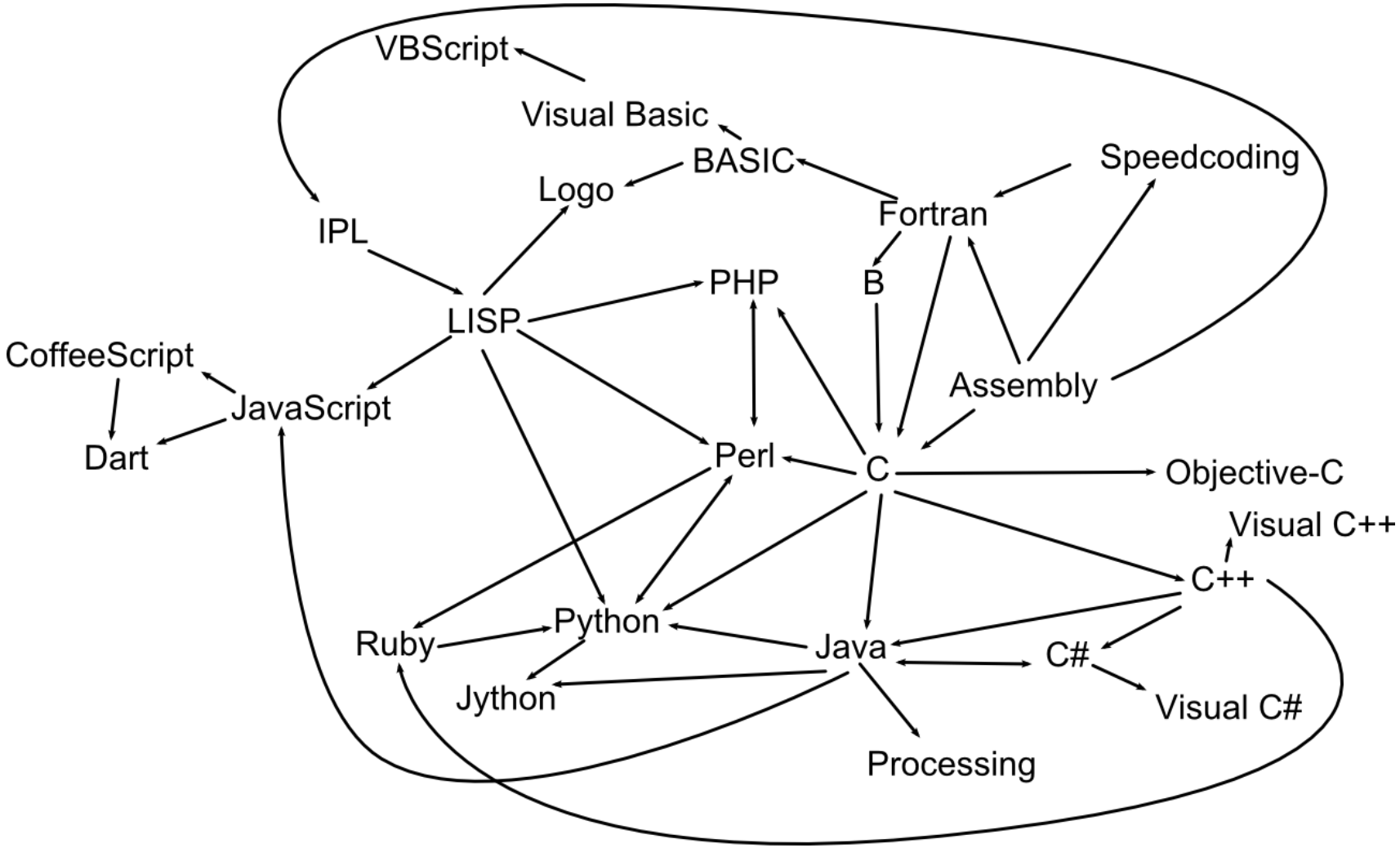
- Supercomputação começa quando desafios de programação acaba
- Dado um “bom” algoritmo, vamos definir:
  - Linguagem de programação adequada
  - Paralelismo indicado
  - Implementação paralela eficiente

Linguagem C++



- Linguagem de programação criada no Bell Labs, em 1985
- É uma das linguagens de programação mais usadas no mundo

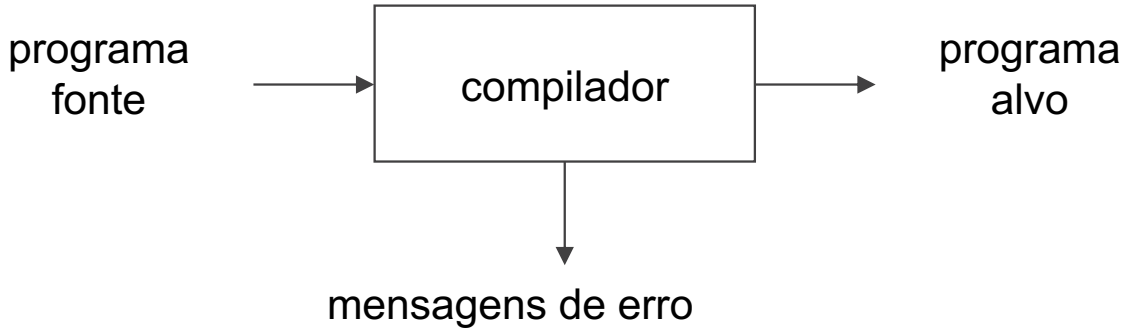
- C++ é derivado da linguagem C, contudo possui uma série de recursos adicionais que traz grandes vantagens para o desenvolvedor



http://3.bp.blogspot.com/-B344Mbjbq94/U5Ha-SlELrI/AAAAAAAAE5I/Tg61Jd-VAdE/s1600/the-family-tree-of-programming-languages.png

O processo de compilação

- Compilador é um software que lê a especificação de um programa em uma linguagem-fonte o traduz em um programa em uma linguagem-alvo



O processo de compilação

- Para compilar um programa em C++ você pode usar a chamada g++

```
$ g++ your_file.cpp -o your_program
```

- Se você estiver no MacOS, uma possível linha para compilar é:

```
$ clang++ -std=c++11 -stdlib=libc++ -Wall hello.cpp -o hello
```

C++, Hello World

- Um programa “Hello World” em C++ é muito parecido com um em C.

```
#include <iostream>

int main() {

  std::cout << "Hello World!\n";

}
```

Pontos comuns com C

- Função principal: int main(int argc, char *argv[]);
- Comentários: /* */ //
- Tipos de dados: int, float, double, char
- Variações de dados: unsigned, short, long
- Qualificadores de variáveis: const, static
- Casting: (int) (float) (char)
- Operações: +, -, *, /, %
- Atribuição: =, +=, -=, *=, /=, %=, >>=, <<=, &=, ^=, |=
- Incremento e decremento: ++ --
- Operadores lógicos: !, &&, ||
- Operador condicional ternário : (? :)
- Operador vírgula: (,)

Pontos comuns com C

- Operadores Bitwise : (&, |, ^, ~, <<, >>)
- Construção Condicional : if, else, switch
- Loops: while, do-while, for
- Comparadores: ==, !=, >, <, >=, <=
- Diretivas de pré-processamento (#define, etc.)
- Declaração de funções: type func(...) { … }
- Vetores e Matrizes: type name [elements][…];
- Enumeradores: enum
- Organização de dados: structs
- Redefinidor de tipos: typedef

C++ - Input/Output (Streams)

- C++ utiliza uma abstração conveniente denominada Streams para executar a entrada e saída de dados.

stream	description
cin	standard input stream
cout	standard output stream
cerr	standard error (output) stream
clog	standard logging (output) stream

C++ - Saída de dados (iostream)

- Utilize o std::cout com dois sinais de menor (<<) para indicar que quer enviar os dados para o console.

```
std::cout << "Bom dia" << std::endl;
```

- std::cout aceita qualquer tipo de dados.

C++ - Entrada de dados (iostream)

- Use o std::cin com dois sinais de maior (>>) para indicar que quer capturar os dados do console

```
string texto;
std::cin >> texto;
```

C++ - Namespaces

- O namespace declara uma região de escopo para os identificadores
- Exemplo de uso de namespace:

```
std::cout << “Ola”;
```

- Outra possibilidade:

```
using namespace std;
cout << “Ola”;
```

C++ - Inicialização de variáveis

- Tradicionalmente

```
type identifier = initial_value;
```

- Inicialização com valor inicial

```
type identifier (initial_value);
int x (0);
```

- Ou

```
type identifier {initial_value};
int x {0};
```

C++ - Inicialização Vetores/Matrizes

- C++ permite preencher um vetor sem definir seu tamanho, se você deixar os colchetes vazios []
- Nesse caso, o compilador assumirá automaticamente o tamanho para o vetor que corresponde ao número de valores incluídos entre as chaves { }.

```
int foo [] = { 16, 2, 77, 40, 12071 };
```

Tipos de dados

- Principais novos tipos de dados que foram introduzidos em C++
    - bool: true / false
    - string: armazena textos e possui recursos para tratar textos

```
#include <string>
string texto;
texto = "exemplo de texto”;
std::cout << texto << std::endl;
```

Atividade prática

Nesta aula iremos implementar códigos em C++ tratando os seguintes aspectos:

- estrutura básica
- variáveis
- condicionais
- laços
- vector
- funções – passagem por valor e passagem por referência

Iremos desenvolver código em tempo real.

Aproveite para relembrar a lógica de programação necessária

E foque no algoritmo que foi escolhido par resolver a situação problema.

Obrigado