Paradigmas de Linguagens de Programação em Python

Aplicações, Critérios e Categorias de Linguagens

Prof. Dr. Daniel Caetano

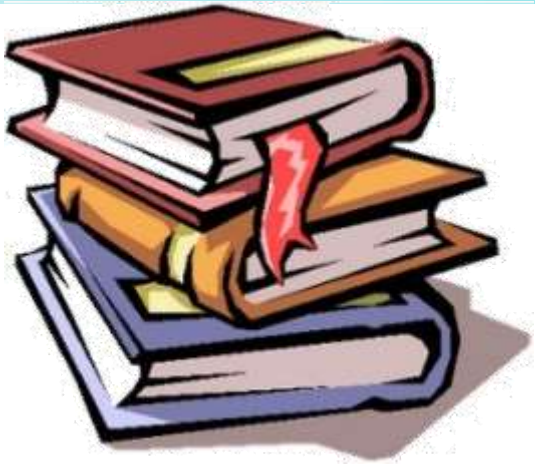
2020 - 2

Objetivos

- Conhecer os grandes domínios de aplicações e suas características
- Compreender os fatores que influenciam a arquitetura das linguagens e os principais paradigmas
- Conhecer os critérios para escolha de linguagens

- **Desafio Aula 02**

Bibliografia da Aula



Material	Acesso ao Material
Apresentação	https://www.caetano.eng.br/ (Paradigmas de Programação – Aula 2)
Livro Texto	Capítulo 1, páginas 5 a 21
Aprenda Mais!	• Vídeo: “O Poder do Paradigma” https://www.youtube.com/watch?v=X3ExqafLgwk • Vídeo: “Programação através de paradigmas” https://www.youtube.com/watch?v=Pg3UeB-5FdA

Antes de Mais nada...

- **Consulte o material da 1ª Aula!**
- **Otimize seus estudos**
  - Se preparar para conteúdo da semana seguinte!
- **Atividades e Desafios Semanais**
  - No site e mural da disciplina: https://www.caetano.eng.br/
- **Será controlada a presença**
  - Chamada ocorrerá sempre nos 15 minutos finais
- **Contato**

Professor	E-mail
Daniel Caetano	prof@caetano.eng.br

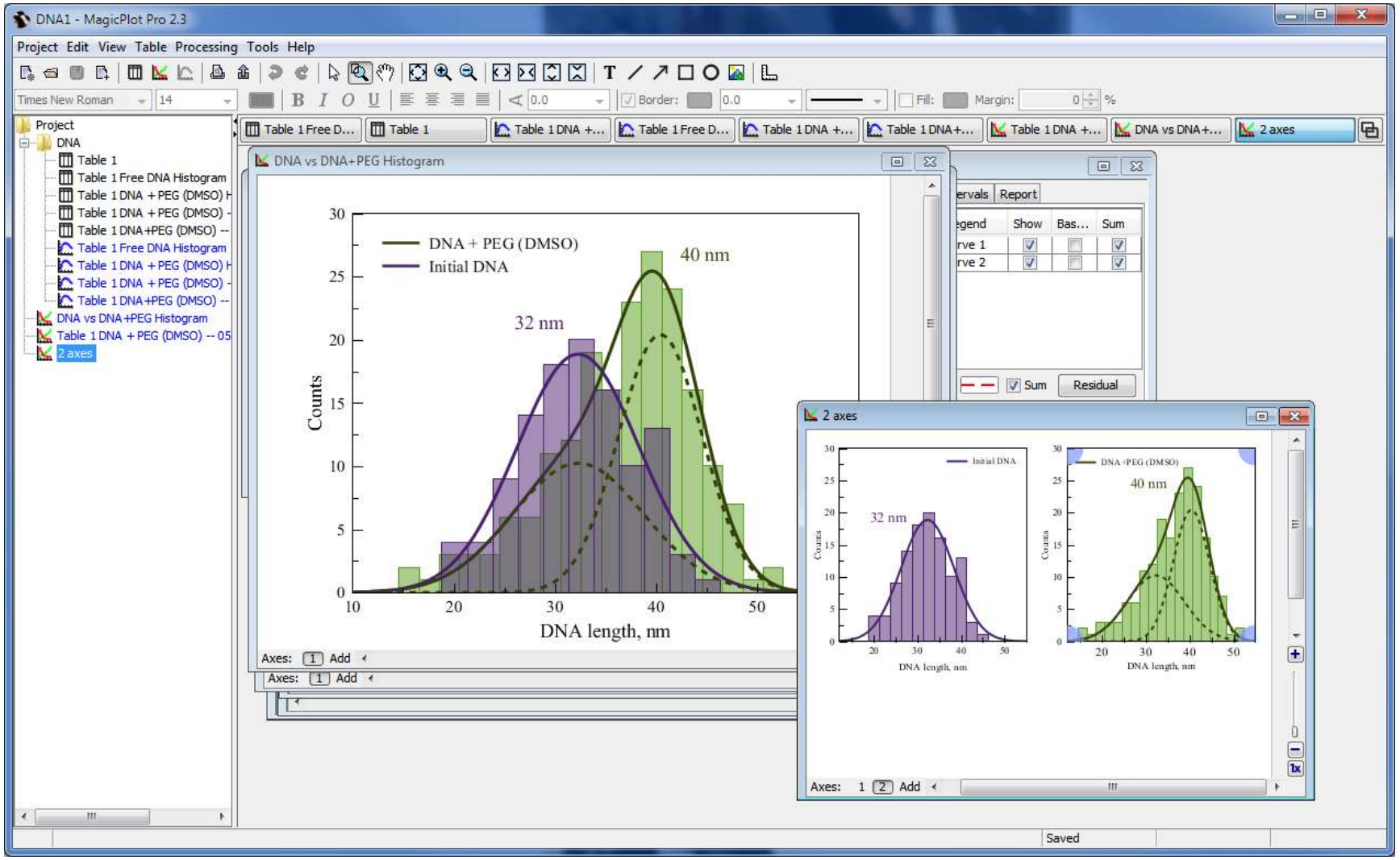
Evolução das Linguagens: Por Quê?

Linha do Tempo

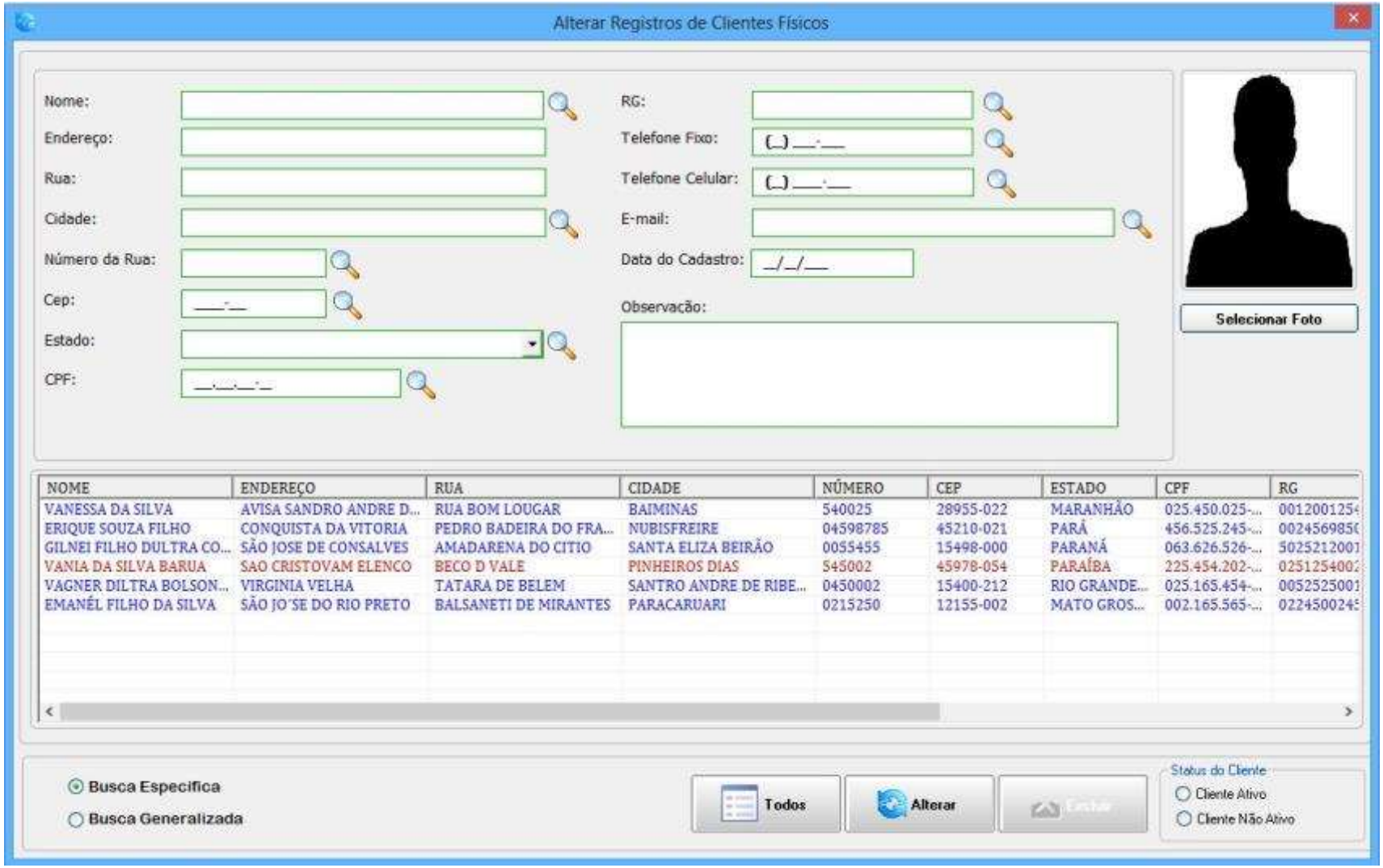
- Evolução das Linguagens
  - Início LM & Assebly
  - 195x/6x: FORTRAN, ALGOL 60, COBOL, Lisp
  - 197x: Pascal, Smalltalk, C, BASIC, Prolog
  - 198x: C++, Object Pascal, Objective C
  - 1991: VisualBASIC, Oak, Python
  - 1995: PHP, Ruby, Java
  - 2001: C#
  - ...

Por quê?

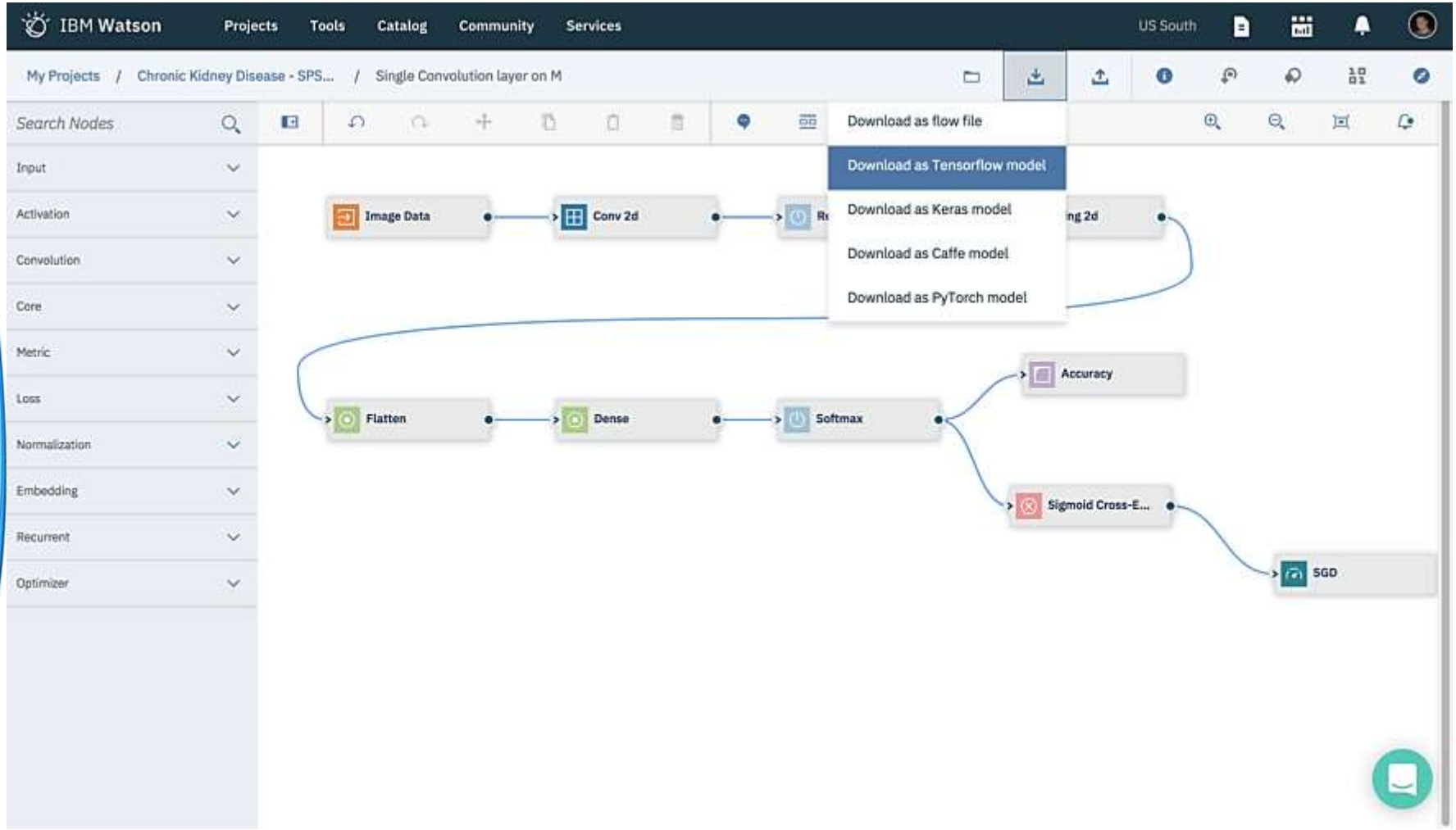
Observe...


DNA1 - MagicPlot Pro 2.3
Project Edit View Table Processing Tools Help
Project
DNA
Table 1
Table 1 Free DNA Histogram
Table 1 DNA + PEG (DMSO) H
Table 1 DNA + PEG (DMSO) -
Table 1 DNA +PEG (DMSO) --
Table 1 Free DNA Histogram
Table 1 DNA + PEG (DMSO) H
Table 1 DNA + PEG (DMSO) -
Table 1 DNA +PEG (DMSO) --
DNA vs DNA+PEG Histogram
Table 1 DNA + PEG (DMSO) -- 05
2 axes
DNA vs DNA+PEG Histogram
DNA + PEG (DMSO)
Initial DNA
40 nm
32 nm
Counts
DNA length, nm
Axes: 1 Add
Report
Show
Sum
Residual
2 axes
Saved

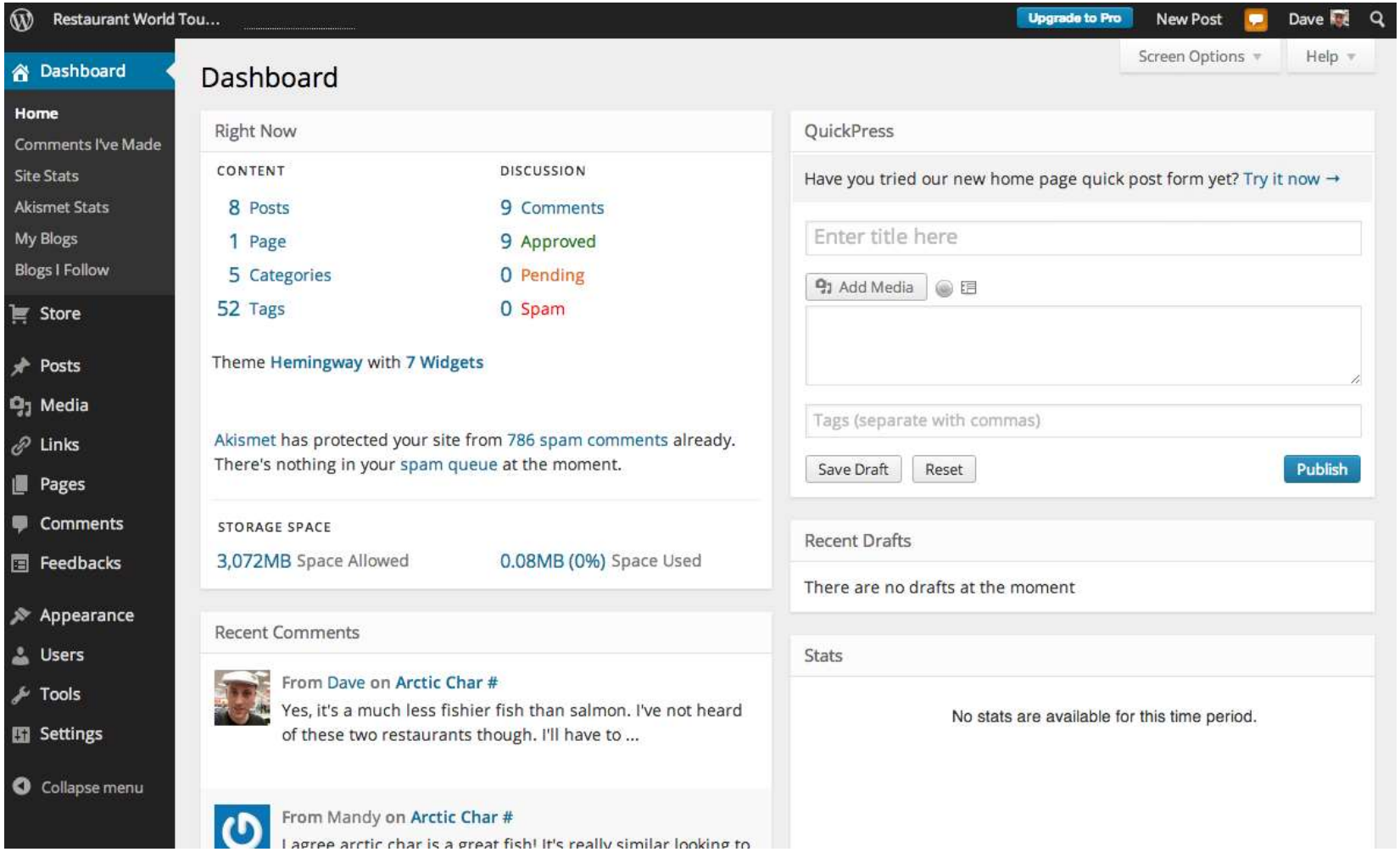
Observe...


Alterar Registros de Clientes Físicos
Nome:
Endereço:
Rua:
Cidade:
Número da Rua:
Cep:
Estado:
CPF:
RG:
Telefone Fixo:
Telefone Celular:
E-mail:
Data do Cadastro:
Observação:
Selecionar Foto
NOME
ENDEREÇO
RUA
CIDADE
NÚMERO
CEP
ESTADO
CPF
RG
VANESSA DA SILVA
AVISA SANDRO ANDRE D...
RUA BOM LOUGAR
BAIMINAS
540025
28955-022
MARANHÃO
025.450.025-...
ERIQUE SOUZA FILHO
CONQUISTA DA VITORIA
PEDRO BADEIRA DO FRA...
NUBISFREIRE
04598785
45210-021
PARÁ
456.525.245-...
GILNEI FILHO DULTRA CO...
SÃO JOSE DE CONSALVES
AMADARENA DO CITIO
SANTA ELIZA BEIRÃO
0055455
15498-000
PARANÁ
063.626.526-...
VANIA DA SILVA BARUA
SAO CRISTOVAM ELENCO
BECO D VALE
PINHEIROS DIAS
545002
45978-054
PARAÍBA
225.454.202-...
VAGNER DILTRA BOLSON...
VIRGINIA VELHA
TATARA DE BELEM
SANTRO ANDRE DE RIBE...
0450002
15400-212
RIO GRANDE...
025.165.454-...
EMANÉL FILHO DA SILVA
SÃO JO´SE DO RIO PRETO
BALSANETI DE MIRANTES
PARACARUARI
0215250
12155-002
MATO GROS...
002.165.565-...
Busca Especifica
Busca Generalizada
Todos
Alterar
Excluir
Status do Cliente
Cliente Ativo
Cliente Não Ativo

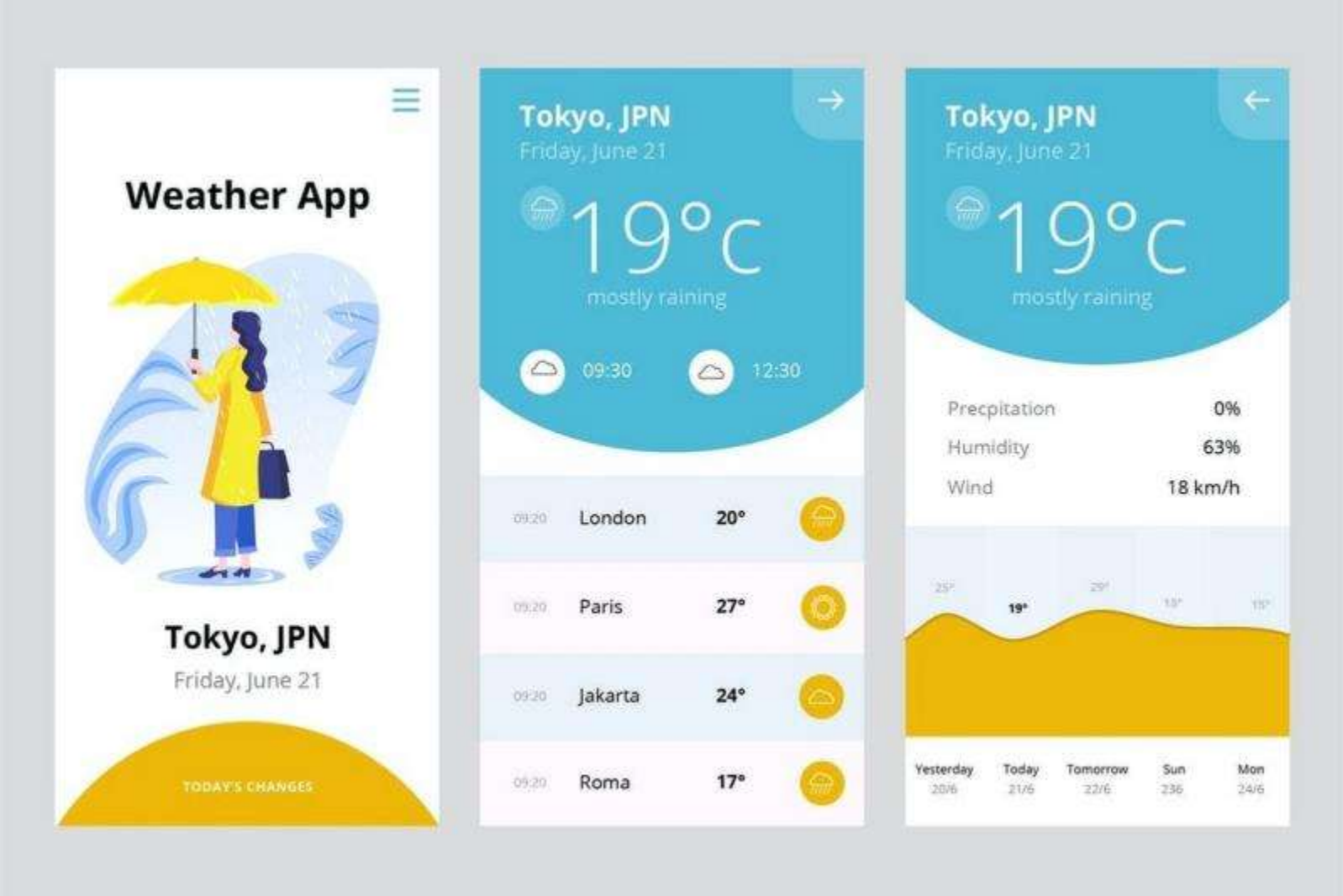
Observe...


IBM Watson
Projects
Tools
Catalog
Community
Services
US South
My Projects / Chronic Kidney Disease - SPS... / Single Convolution layer on M
Search Nodes
Input
Activation
Convolution
Core
Metric
Loss
Normalization
Embedding
Recurrent
Optimizer
Download as flow file
Download as Tensorflow model
Download as Keras model
Download as Caffe model
Download as PyTorch model
Image Data
Conv 2d
Flatten
Dense
Softmax
Accuracy
Sigmoid Cross-E...
SGD

Observe...


Restaurant World Tou...
Upgrade to Pro
New Post
Dave
Screen Options
Help
Dashboard
Home
Comments I've Made
Site Stats
Akismet Stats
My Blogs
Blogs I Follow
Store
Posts
Media
Links
Pages
Comments
Feedbacks
Appearance
Users
Tools
Settings
Collapse menu
Dashboard
Right Now
CONTENT
8 Posts
1 Page
5 Categories
52 Tags
DISCUSSION
9 Comments
9 Approved
0 Pending
0 Spam
Theme Hemingway with 7 Widgets
Akismet has protected your site from 786 spam comments already.
There's nothing in your spam queue at the moment.
STORAGE SPACE
3,072MB Space Allowed
0.08MB (0%) Space Used
Recent Comments
From Dave on Arctic Char #
Yes, it's a much less fishier fish than salmon. I've not heard of these two restaurants though. I'll have to ...
From Mandy on Arctic Char #
QuickPress
Have you tried our new home page quick post form yet? Try it now →
Enter title here
Add Media
Tags (separate with commas)
Save Draft
Reset
Publish
Recent Drafts
There are no drafts at the moment
Stats
No stats are available for this time period.

Observe...


Weather App
Tokyo, JPN
Friday, June 21
TODAY'S CHANGES
Tokyo, JPN
Friday, June 21
19°c
mostly raining
09:30
12:30
09:20 London 20°
09:20 Paris 27°
09:20 Jakarta 24°
09:20 Roma 17°
Tokyo, JPN
Friday, June 21
19°c
mostly raining
Precpitation 0%
Humidity 63%
Wind 18 km/h
Yesterday 20/6
Today 21/6
Tomorrow 22/6
Sun 23/6
Mon 24/6

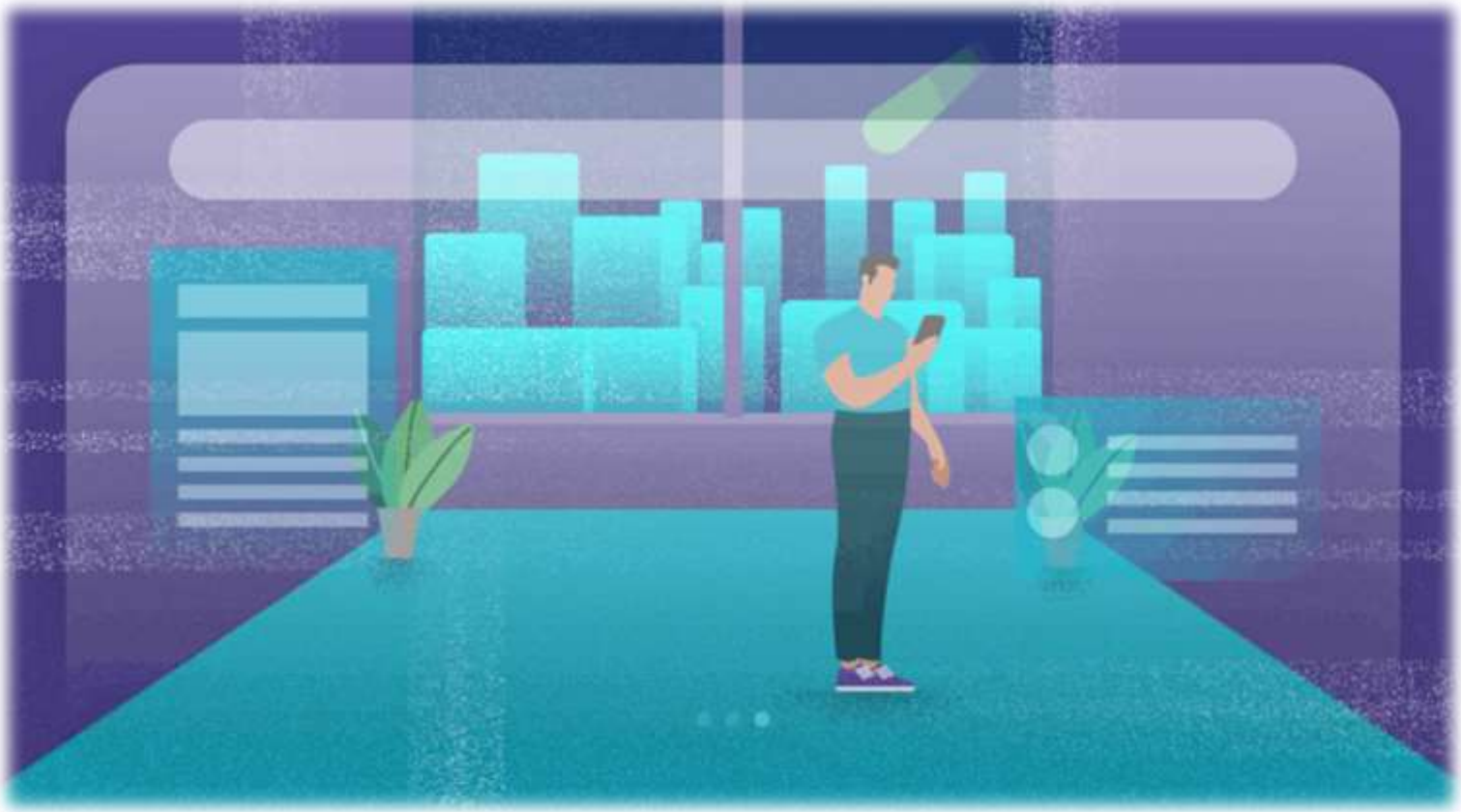
Motivos para Diversidade de Linguagens:

1. Domínios de Programação

https://www.menti.com/

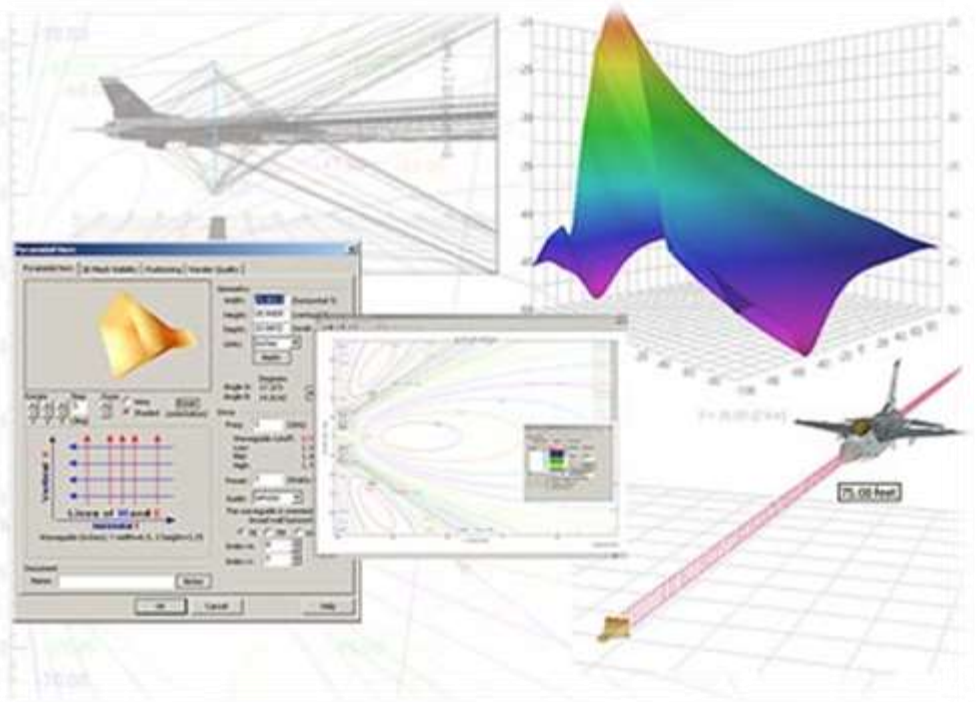
Domínios de Programação

- Grandes Categorias de Software
  - Suas características: influenciam as linguagens



Domínios de Programação

- Aplicações científicas
  - Primeiro tipo de aplicações (ALGOL60, FORTRAN, *C*)
  - Focada em cálculos e eficiência computacional



Domínios de Programação

- Aplicações comerciais/empresariais
  - Após 1ª guerra: bancos, empresas... (COBOL, *Java, C#*)
  - Foco em cálculos decimais, geração de relatórios.

Domínios de Programação

- Aplicações de IA
  - Inferências e deduções (Prolog, Lisp, *C*, *Python*)
  - Computação simbólica e associações.

Domínios de Programação

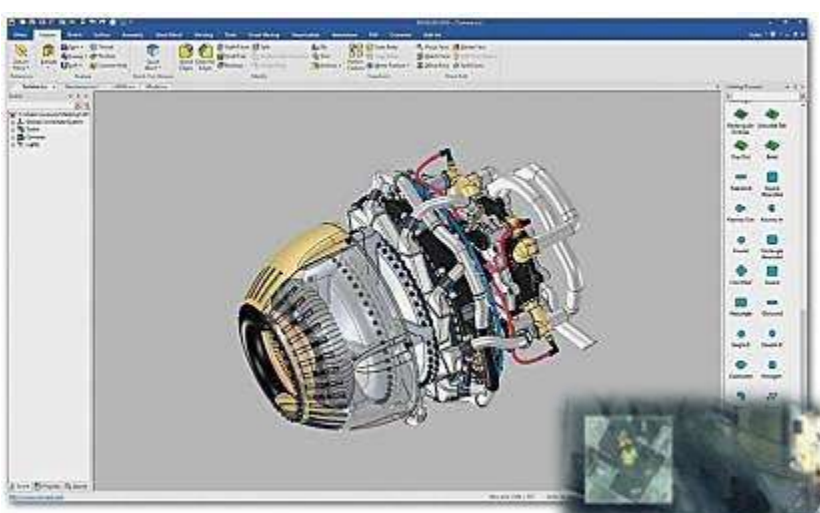
- Sistemas Básicos
  - Lidar diretamente como hardware (C, Assembly)
  - Foco em eficiência e baixo consumo de recursos.

Domínios de Programação

- Aplicações para Web
  - Acesso universal (JavaScript, Java)
  - Foco em difusão de conteúdo dinâmico.

Domínios de Programação

- Outros... Engenharia, jogos etc...
  - Combinação complexa de requisitos (C, Java, Python)
  - Linguagens de “propósito geral”



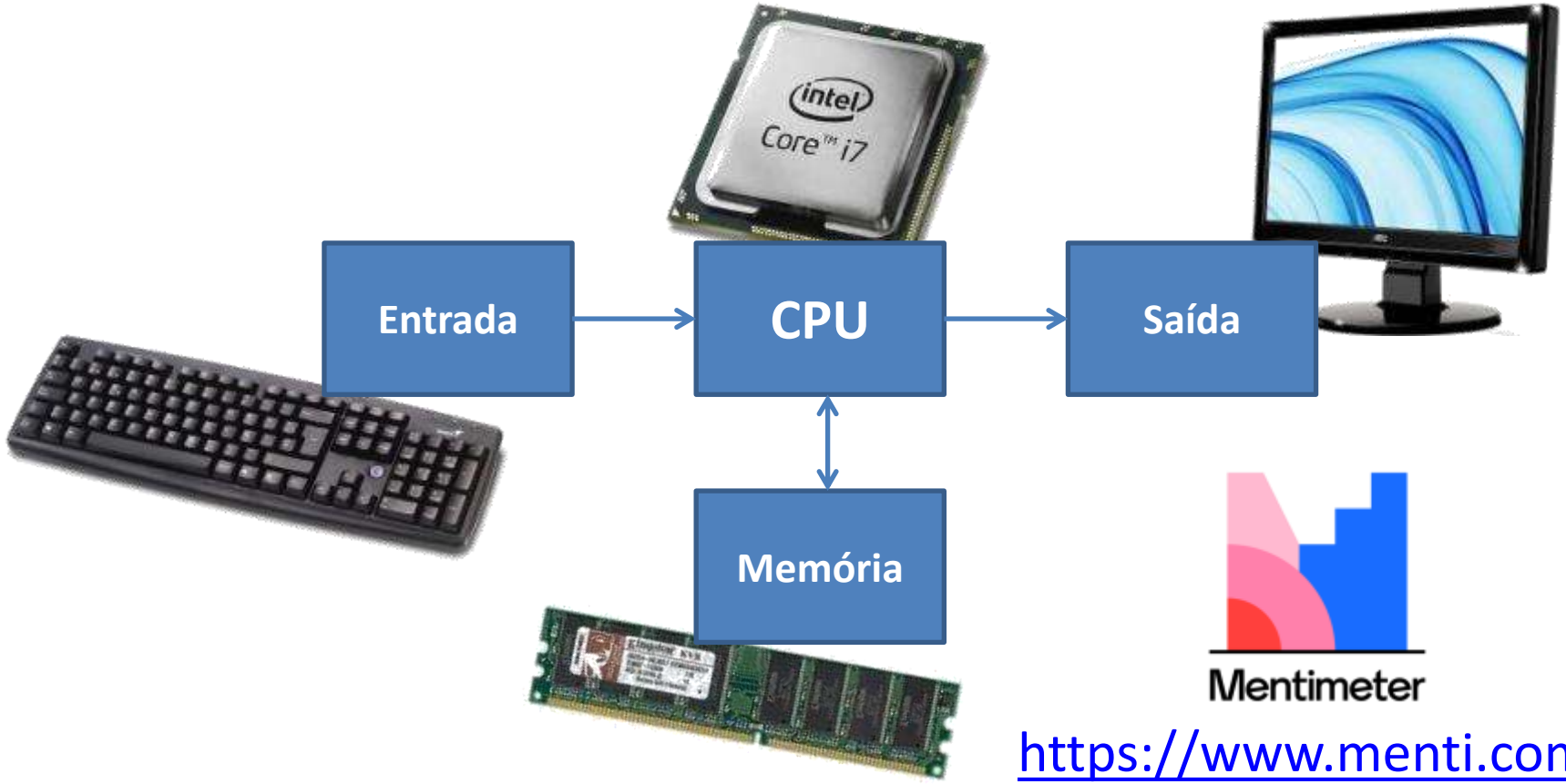
Motivos para Diversidade de Linguagens:

2. Quem Executa as Tarefas?

https://www.menti.com/

Arquitetura de Computadores

- Lógica da máquina → lógica da linguagem
- Arquitetura de von Neumann



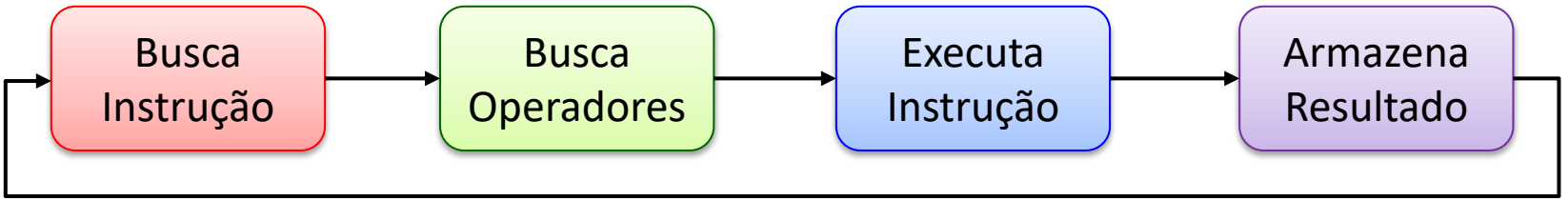
Mentimeter

https://www.menti.com/

Arquitetura de Computadores

- Arquitetura de von Neumann
  - Programas ficam na memória, como os dados
  - Procedimentos sequenciais para cálculo
  - Armazenamento de resultados na memória.
- Programas x Dados
  - Executar x Armazenar/Recuperar

```
x = calcula(y)
```



Motivos para Diversidade de Linguagens:

3. Metodologias de Projeto

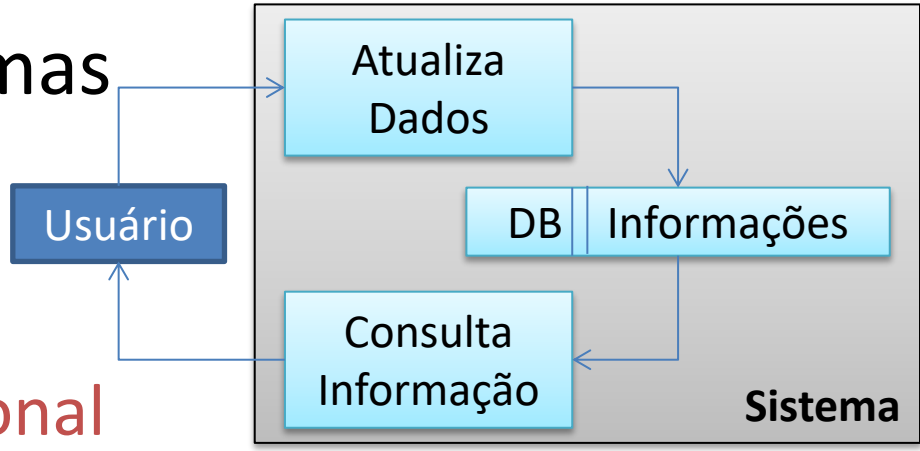
https://www.menti.com/

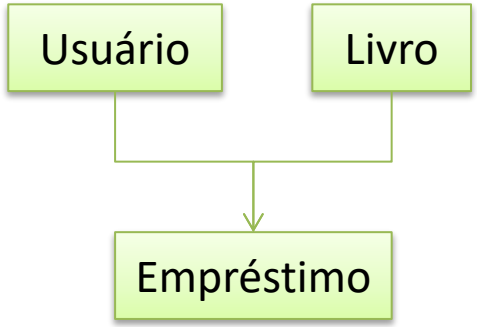
Software: Resolver um Problema

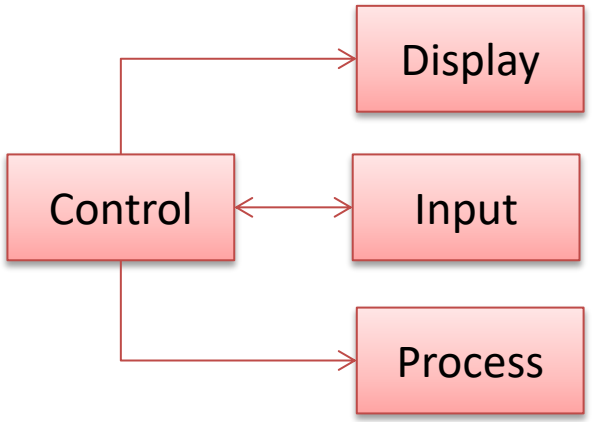
- Problemas Complexos x Custo x Tempo
  - "Sentar e programar" → Projetar
  - Análise e Projeto.
- Como implementar um sistema?
  - Compreender o domínio do problema
  - Propor modelo simplificado
  - Propor modelo detalhado
  - Implementar
  - Testar
  - Implantar.

Análise de Sistemas

- LM & ASM: Fluxogramas
- Análise Estruturada
  - Fluxo de Dados (DFD)
  - Decomposição Funcional
- Análise Orientada a Objetos







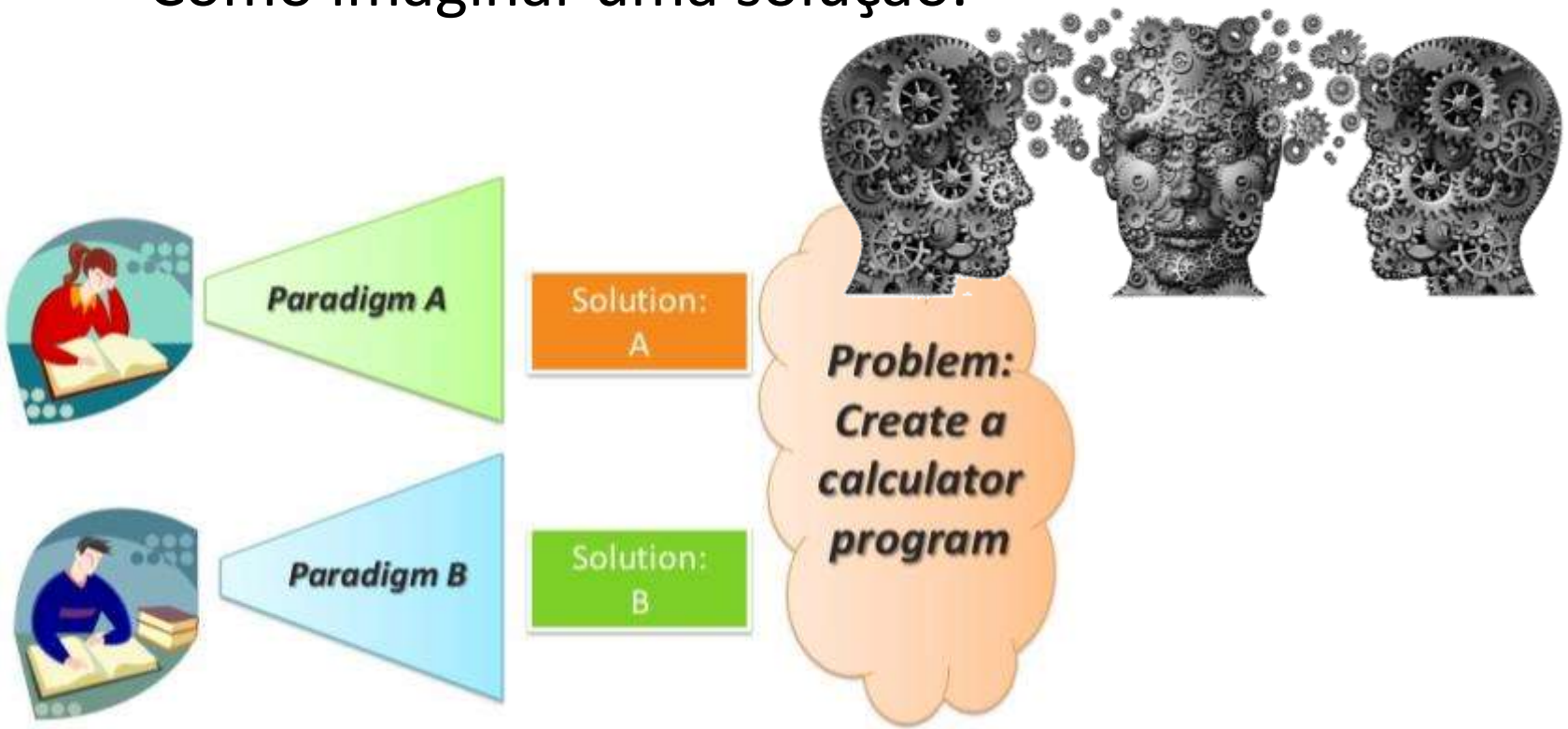
- O que muda menos ao longo do tempo??

CATEGORIAS DE LINGUAGENS

https://www.menti.com/

O que é um Paradigma?

- Forma de interpretar e pensar o mundo
  - Como interpretar problemas complexos....
  - Como imaginar uma solução.



Paradigmas de Linguagem

1. Linguagens **Imperativas**
   - Influência: arquitetura de von Neumann
   - Variáveis e procedimentos
- Linguagens imperativas podem ser
  - Estruturadas/Procedurais
    - Influência: análise estruturada
    - Ex.: COBOL, FORTRAN, C, Pascal...
  - Orientada a Objetos
    - Influência: an. orientada a objetos
    - Ex.: Smalltalk, C++, Python, Java, C#

Paradigmas de Linguagem

2. Linguagens **Declarativas**

- Em oposição às imperativas
- Abstraem a ideia de variável e/ou sequência

- Linguagens declarativas podem ser:
  - Funcionais
    - Influência: funções matemáticas
    - Ex.: Haskell, Erlang, R, XSLT
  - Lógicas
    - Influência: lógica matemática
    - Ex.: Prolog, LISP

Paradigmas de Linguagem

- Na prática...
  - Muitas linguagens são multiparadigma

- C: imperative, procedural
- C++: imperative, object-oriented, generic, functional style(not functional)
- C#: imperative, declarative, functional, generic, object-oriented(class-based), component-oriented
- Java: concurrent, class-based, functional(Java8)
- JavaScript: imperative, functional, object-oriented
- Python: imperative, functional, procedural, object-oriented
- Ruby: imperative, functional, object-oriented
- SQL: declarative, data-driven

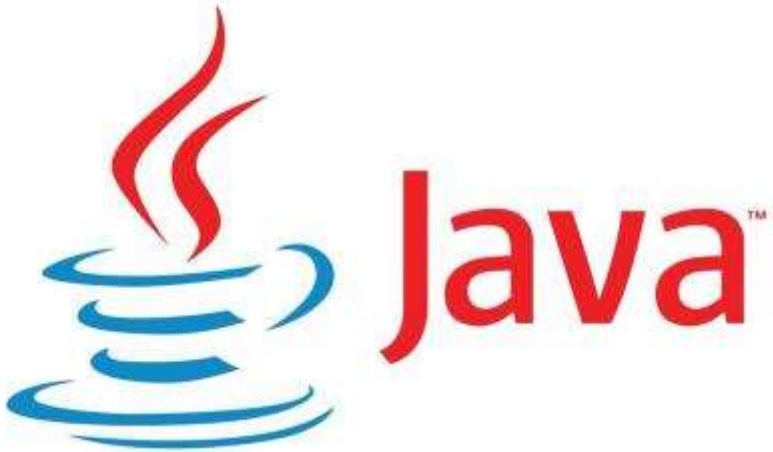
Exemplo – Imperativa Estruturada

```
************************************************************************
*
* Máximo divisor comum de dois inteiros positivos m e n.
* Algoritmo de Euclides
*
************************************************************************
*
        program mdcom
      implicit none
*
        integer n, m, divisor, dividendo, mdc, resto

        write( 0, * ) 'Calcular maximo divisor comum de M e N ? '
        read ( *, * ) m, n
        dividendo = m
        divisor = n

        resto = mod( dividendo, divisor )
        do while ( resto .ne. 0 )
        dividendo = divisor
          divisor = resto
          resto = mod( dividendo, divisor )
        end do
        mdc = divisor

        write( 0, * ) 'MDC de ', m, ' e ', n, ' = ', mdc

        end program mdcom
```

Exemplo: Imperativa Estruturada

```c
#include <stdio.h>
#include <assert.h>

//
// Função mdc
//
int mdc(int num1, int num2) {

    int resto;

    do {
        resto = num1 % num2;

        num1 = num2;
        num2 = resto;

    } while (resto != 0);

    return num1;
}

//
// Testes
//
int main() {

    assert(3 == mdc(24, 9));
    assert(10 == mdc(30, 20));

    return 0;
}
```

Exemplo: Imperativa O.O.




```
class Lamp {
    boolean isOn;

    void turnOn() {
        // initialize variable with value true
        isOn = true;
        System.out.println("Light on? " + isOn);

    }

    void turnOff() {
        // initialize variable with value false
        isOn = false;
        System.out.println("Light on? " + isOn);
    }
}


class Main {
    public static void main(String[] args) {

        // create objects l1 and l2
        Lamp l1 = new Lamp();
        Lamp l2 = new Lamp();

        // call methods turnOn() and turnOff()
        l1.turnOn();
        l2.turnOff();
    }
}
```

Exemplo: Pyhton (Or. a Objetos)

```
class ComplexNumber:
    def __init__(self, r=0, i=0):
        self.real = r
        self.imag = i

    def get_data(self):
        print(f'{self.real}+{self.imag}j')


# Create a new ComplexNumber object
num1 = ComplexNumber(2, 3)

# Call get_data() method
# Output: 2+3j
num1.get_data()

# Create another ComplexNumber object
# and create a new attribute 'attr'
num2 = ComplexNumber(5)
num2.attr = 10

# Output: (5, 0, 10)
print((num2.real, num2.imag, num2.attr))

# but c1 object doesn't have attribute 'attr'
# AttributeError: 'ComplexNumber' object has no attribute 'attr'
print(num1.attr)
```

Exemplo: Declarativa Funcional

```
module Main (main) where

numDivs :: Integer -> Integer
numDivs n
  = toInteger $ length [x | x <- [2 .. ((n `quot` 2) + 1)], n `rem` x == 0] + 2

triaList :: [Integer]
triaList = [foldr (+) 0 [1 .. n] | n <- [1 ..]]
triaList2 = go 0 1
  where go cs n = (cs + n) : go (cs + n) (n + 1)

sol :: Integer -> Integer
sol n = head $ filter (\ x -> numDivs (x) > n) triaList2
main = print $ sol 150
```

Exemplo: Declarativa Lógica

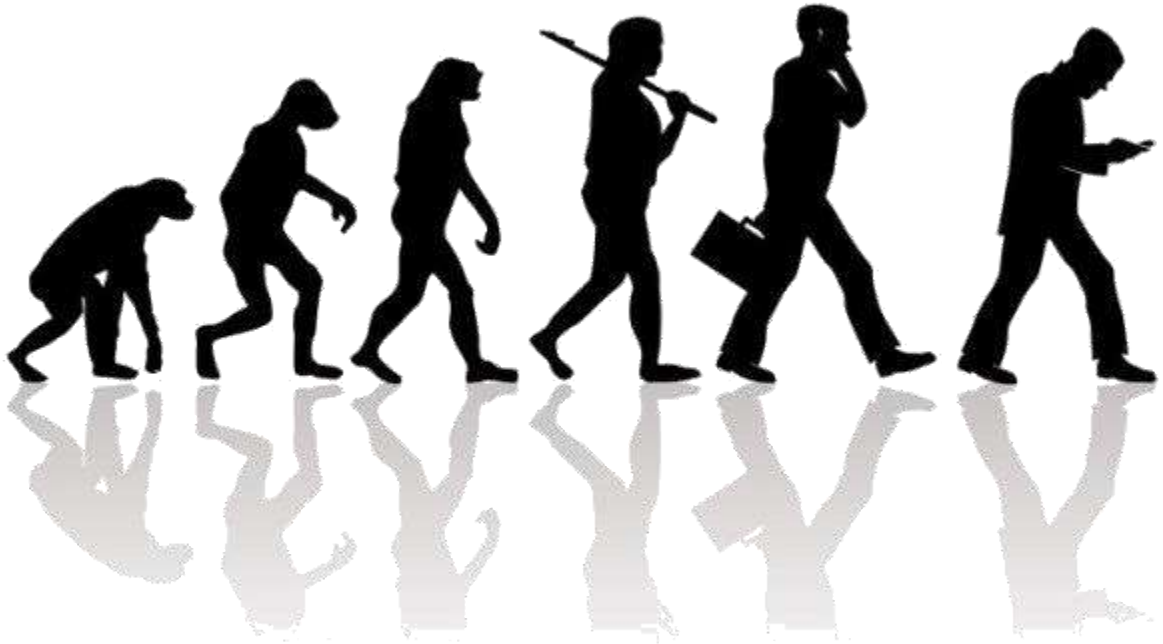
```
(defun encode (lista)
        (if (eql lista nil)
                nil
                (cons (list (length (pega lista)) (car lista)) (encode (tira lista)))
        )
)

(defun pega (lista)
    (cond ((eql lista nil) nil)
          ((eql (cdr lista) nil) lista)
          ((equal (car lista) (cadr lista))
              (cons (car lista) (pega (cdr lista))))
          (t (list (car lista)))
    )
)

(defun tira (lista)
    (cond ((eql lista nil) nil)
          ((eql (cdr lista) nil) nil)
          ((equal (car lista) (cadr lista))
              (tira (cdr lista)))
          (t (cdr lista))
    )
)
```

Critérios para Escolha de Linguagem

https://www.menti.com/

Critérios de Avaliação

- Será que os critérios sempre foram os mesmos?



Mentimeter

https://www.menti.com/

Critérios de Avaliação

- **No início:**
  - Computadores caros e lentos
  - Aplicativos simples
  - Principal critério: eficiência

- **Posteriormente:**
  - Computadores baratos e rápidos
  - Aplicativos complexos
  - Critérios
    - Eficiência de desenvolvimento
    - Facilidade de manutenção

Critérios de Avaliação

- **Em que podemos traduzir?**
  - Eficiência de desenvolvimento
  - Facilidade de manutenção

Mentimeter

https://www.menti.com/

Critérios de Avaliação

- **Critérios práticos**
  - Legibilidade
  - Facilidade de escrita
  - Confiabilidade
  - Custo
  - Portabilidade
  - ...

- "Pesos" variam caso a caso!

Critérios de Avaliação

1. Legibilidade

- ***Facilidade de compreensão***
- Simplicidade geral
  - Muitas formas de fazer a mesma coisa pode prejudicar
    - Exemplo: incrementar em C++ ou Java:

```
contador = contador + 1
contador += 1
contador++
++contador
```

  - Sobrecarga de operadores
    - Exemplo:

```
x = a + b
```

- **a** e **b** números; x = número
- **a** uma cor e **b** uma letra; x = ?

- E simplicidade extrema?
- Também é problema!

Ex.: assembly!

Critérios de Avaliação

1. Legibilidade

- Ortogonalidade
  - Poucas primitivas, muitas combinações
  - Exemplo:

Assembly i8080

```
MOV r, r
MVI r, n
```

Assembly Z80

```
LD a, b
```

- Tipos de dados
  - Adequação torna a leitura mais clara
  - Exemplo:

BASIC

```
logado = 1
```

Java / Python

```
logado = true
```

Critérios de Avaliação

1. Legibilidade

- Projeto da sintaxe
  - Palavras especiais... Símbolos...
  - Exemplo:

C / C++ / Java

```
while (x<10) {
      x = x + 1;
    }
```

Pascal / Delphi

```
while x<10 do
    begin
    x := x + 1;
    end
```

Pyhton

```
while x<10:
    x = x + 1
```

- Forma e significado
  - Significados mutantes são ruins
    - Exemplo: **static** em C

Critérios de Avaliação

2. Facilidade de Escrita

- ***Adequação de uma linguagem ao domínio***
- Simplicidade e Ortogonalidade
  - Muitas construções, mais desconhecimento
  - Exemplo:

C / C++

```
x = ++y, y/2;
```

Critérios de Avaliação

2. Facilidade de Escrita

- ***Adequação de uma linguagem ao domínio***
- Expressividade
  - Várias maneiras de expressar as coisas
  - Exemplo:

C / C++ / Java

```
cont = cont + 1;
cont++;
```

C / C++ / Java

```
x = 0;
while (x<10) {
    x++;
    }
```

C / C++ / Java

```
for (x=0; x<10;x++) {
    ...
    }
```

Facilidade de Escrita x Facilidade de Leitura

Critérios de Avaliação

3. Confiabilidade

- ***Comportamento conforme especificação***
- Legibilidade e facilidade de escrita
  - Certamente evita erros!
- Verificação de tipos
  - Cadastrar **cliente** não cadastra um **inteiro**! (Java, C...)
- Tratamento de exceções
  - Obrigar a tratar situações de erro (Java, C++, C#, Python)
- Apelidos
  - Perigo: vários nomes para mesmo valor na memória

Critérios de Avaliação

4. Custo

- Custo de treinamento (simplicidade, ortogonalidade)
- Custo de escrita (facilidade de escrita)
- Custo de compilação
- Custo de execução
- Custo de implementação (ambiente)
- Custo da baixa confiabilidade
- Custo de manutenção (legibilidade e facilidade de escrita).

Critérios de Avaliação

5. Portabilidade

- ***Capacidade de transportar para outros sistemas***
- Padronização da linguagem

ATIVIDADE

Atividade 1

- Grupos
  - Entrar na sala do grupo para discussão: 15 minutos
- Discutir as seguintes questões
  - **Grupo 1**: Por que é útil que o desenvolvedor conheça as características das várias linguagens, mesmo que não vá projetar uma linguagem?
  - **Grupo 2**: Quais as desvantagens estão relacionadas à uma linguagem ter recursos demais?
  - **Grupo 3**: O que significa um programa ser confiável? Identifique 3 aplicações que exijam alta confiabilidade.
  - **Grupo 4**: Por que verificar os tipos de dados é importante? Qual o problema que usar tipos traz?
  - **Grupo 5**: A linguagem mais usada é sempre a melhor? Argumente!

Atividade 1 - Discussão

- Respostas de cada grupo
  - **Grupo 1**: Por que é útil que o desenvolvedor conheça as características das várias linguagens, mesmo que não vá projetar uma linguagem?
  - **Grupo 2**: Quais as desvantagens estão relacionadas à uma linguagem ter recursos demais?
  - **Grupo 3**: O que significa um programa ser confiável? Identifique 3 aplicações que exijam alta confiabilidade.
  - **Grupo 4**: Por que verificar os tipos de dados é importante? Qual o problema que usar tipos traz?
  - **Grupo 5**: A linguagem mais usada é sempre a melhor? Argumente!

Atividade 2

- Grupos: discussão de 15 minutos
  - **Grupo 1**: Vocês acreditam que a capacidade de abstração é influenciada por nosso domínio de linguagens? Argumentem!
  - **Grupo 2**: Como vocês defenderiam a ideia de se usar uma única linguagem para qualquer tipo de software?
  - **Grupo 3**: Como vocês defenderiam a ideia de **não** se dever adotar uma única linguagem para qualquer tipo de software?
  - **Grupo 4**: Quais (dois) aspectos de custo o grupo considera mais relevante? Argumentem!
  - **Grupo 5**: Avalie com os critérios apresentados o fato de a maioria das linguagens permitir dois tipos de comentários: a) de uma única linha e b) de várias linhas.

Atividade 2 - Discussão

- Respostas de cada grupo
  - **Grupo 1**: Vocês acreditam que a capacidade de abstração é influenciada por nosso domínio de linguagens? Argumentem!
  - **Grupo 2**: Como vocês defenderiam a ideia de se usar uma única linguagem para qualquer tipo de software?
  - **Grupo 3**: Como vocês defenderiam a ideia de **não** se dever adotar uma única linguagem para qualquer tipo de software?
  - **Grupo 4**: Quais (dois) aspectos de custo o grupo considera mais relevante? Argumentem!
  - **Grupo 5**: Avalie com os critérios apresentados o fato de a maioria das linguagens permitir dois tipos de comentários: a) de uma única linha e b) de várias linhas.

ENCERRAMENTO

Resumo e Próximos Passos

- Grandes domínios de aplicações
- Fatores que influenciam as linguagens
- Os principais paradigmas de linguagens
- Os critérios para a escolha de linguagens
- **Pós Aula**: Aprenda Mais, Pós Aula e Desafio!
  - No padlet: https://padlet.com/djcaetano/paradigmas

- Trade-offs e Compilação x Interpretação
- Ambientes de Programação

PERGUNTAS?