ANÁLISES E COMPARAÇÕES ENTRE CERTAS LINGUAGENS DE PROGRAMAÇÃO

Isabeli Rosana Reik [1]

Matheus Dias Negrão [2]

RESUMO: O referido artigo irá apresentar uma análise e comparação entre certas linguagens de programação. A importância das linguagens de programação atualmente é inquestionável, pelo fato de que a grande parte da tecnologia produzida depende das mesmas. A linguagem como conhecemos hoje é recente, mas sua história já é mais antiga com as contribuições de Ada Lovelace. Foi realizada uma pesquisa sobre um grupo de linguagens, destacando suas vantagens e desvantagens, além de realizar dois algoritmos diferentes em cada uma delas para a comparação da quantidade de linhas de código utilizadas. No gráfico da soma simples, C++ e Java possuem o maior número de linhas, e no gráfico do hello world C++ teve também o maior número de linhas. Apesar dessas linguagens serem orientadas a objeto, isto não é um fator que influencia no número de linhas, mas sim provavelmente a construção da própria linguagens, a sua classificação e a inclusão de bibliotecas.

PALAVRAS CHAVES: Linguagem de programação. Paradigmas. Comparação.

INTRODUÇÃO:

As linguagens de programação surgiram junto com os primeiros computadores. Ada Lovelace é considerada a primeira programadora da história, ela escreveu o primeiro algoritmo que seria processado pela máquina analítica de Charles Babbage, porém os projetos eram muito à frente de sua época, e Ada não viu seu algoritmo em execução. O algoritmo consistia em "computar os valores de funções matemáticas, além de publicar uma coleção de notas sobre a máquina analítica. " (WIKIPEDIA, 20--).

Atualmente a importância de saber programar é grande, pois, os computadores estão ao nosso redor e estão presentes em quase todas as profissões, sendo que além de conseguir

[1] Discente do curso de ciência da computação, 2017/2, da Universidade Federal da Fronteira Sul - Campus Chapecó. Email: isabelireik2@gmail.com.

[2] Discente do curso de ciência da computação, 2017/2, da Universidade Federal da Fronteira Sul - Campus Chapecó. Email: matheus.dnegrao@gmail.com.

entender como os sistemas funcionam, a programação nos ajuda a tornarmos usuários melhores.

O presente artigo irá apresentar um compilado de algumas linguagens de programação, abordando alguns aspectos da sua história, suas características e de exemplos de suas implementações. Iremos fazer uma comparação entre elas, apontando vantagens e desvantagens entre as mesmas.

"Uma linguagem de programação é um método padronizado para expressar instruções para um computador. É um conjunto de regras sintáticas e semânticas usadas para definir um programa de computador. " (LAUER, 2008).

A principal motivação para a criação das linguagens de programação, é a maior praticidade para os humanos escreverem códigos em uma linguagem mais parecida com a linguagem usada no cotidiano, do que escrever complexos algoritmos em linguagem de máquina. Essa motivação fez com que a produtividade aumentasse, e a evolução das mesmas ocorresse de maneira acelerada com o passar do tempo, com a criação de sofisticadas linguagens de alto nível e novos paradigmas de programação.

As linguagens de programação podem ser classificadas em: alto nível, médio nível e baixo nível. As consideradas de alto nível são as que a sua sintaxe se assemelha a linguagem natural humana; além de que, elas podem ser migradas de computadores diferentes que seu funcionamento não é alterado. As linguagens de nível baixo, que "são linguagens totalmente dependentes da máquina, ou seja, que o programa que se realiza com este tipo de linguagem não pode ser migrado ou utilizado em outras máquinas. " (WEBINSIDER, 20--), geralmente são linguagens de descrição de hardware. E por último, as de nível médio, que se encontram em um ponto médio das duas anteriores, possuem habilidades de algumas funções das linguagens de nível baixo como algumas de nível alto.

Já em relação a paradigmas de programação, ele é definido como características que um grupo de linguagens compartilham em comum, dito como um mesmo padrão de programação. Os principais paradigmas utilizados são: imperativo/procedural, orientado a objeto e estruturado. Há também as linguagens chamadas multiparadigma, que não suporta somente um paradigma, mas sim vários deles. Todos os paradigmas citados acima serão esclarecidos no decorrer do artigo, porém será dado mais ênfase no estruturado e orientado a objeto, pois são os paradigmas mais presentes no mercado e os mais conhecidos pela

comunidade. Será também exposto e desenvolvido acerca do funcionamento de linguagem multiparadigma.

O presente artigo irá apresentar como base, e por maior utilização da comunidade da computação, as seguintes linguagens de programação divididas em paradigmas: procedural/imperativo (Cobol), estruturada (C, Pascal), orientada a objeto (Java, Ruby e PHP – também como linguagem de programação WEB) e multiparadigma (Swift, C++, Python).

O artigo se encontra estruturado da seguinte forma: um breve resumo, uma introdução ao assunto, a revisão da literatura com o aprofundamento do assunto, os procedimentos metodológicos do artigo, a análise e discussão dos resultados obtidos, considerações finais do artigo e por fim as referências.

REVISÃO DA LITERATURA:

As linguagens de programação num todo não começaram com a semântica e sintaxe que as linguagens atuais detêm, elas começaram como maneiras de armazenar instruções em cartões perfurados e realizar funções, e permaneceram assim até meados dos anos 1950, onde se deu início ao que é chamado de “Linguagem de Máquina”, ou seja, era uma forma de instruir os primeiros computadores ao que fazer. As linguagens de máquina eram complicadas e difícil entendimento para os programadores, o que tornava os “comandos difíceis de serem escritos e os tornava mais suscetíveis a erros humanos” (WIKIPEDIA, 20--).

Com o surgimento de novas pesquisas e o avanço tecnológico, as novas linguagens começaram a ter uma nova nomenclatura nos anos 1950, eram as “linguagens de ordem mais alta”, que deu início a independência da máquina ou arquitetura específica para funcionar (NOONAN, 2010[2007]). As linguagens de nível mais alto eram mais fáceis de serem entendidas e programadas, com o surgimento do FORTRAN em 1954 e o ALGOL em 195-, se teve o início da programação por blocos, onde cada bloco tinha uma função específica e não podia ‘invadir’ outros blocos, iniciando a era das linguagens com sintaxe e semântica mais presentes.

Entre o final de 1960 e meados de 1970, as linguagens de programação deram um salto em tecnologia e nos paradigmas, criando vários dos paradigmas utilizados atualmente na programação, como o Estruturado, imperativo e procedural. Em 1967 tivemos o primeiro conceito de “classes” com uma linguagem chamada Simula, esse é utilizado veementemente

hoje em linguagens como o Java. Vale citar o nascimento de uma das linguagens mais populares e mais utilizadas na época, O BASIC (acrônimo para ***B**eginner's **A**ll-purpose **S**ymbolic **I**nstruction **C**ode*; em português: Código de Instruções Simbólicas de Uso Geral para Principiantes), que detinha grande mercado e era bastante utilizada pela comunidade.

Também nesse período houve a criação da linguagem C, sucessora de da linguagem B, que tinha como propósito ser uma linguagem de alto nível para a programação dentro do sistema Unix, que tinha seu desenvolvimento parelho ao da linguagem (GUDWIN, 1997). Com o sucesso do sistema, a linguagem ficou famosa e ganhou espaço no mercado, sendo vista como linguagem de propósito geral, o C se tornou uma das mais importantes e influenciadoras linguagens de todos os tempos. O C ainda é bastante utilizado no meio acadêmico, como primeira linguagem de contato dos alunos, como por exemplo na UFFS.

O C ganhou um número de usuários enorme, o que fez com que o grupo de desenvolvedores da linguagem lançassem uma nova versão ainda melhor, o C++, que como o próprio nome já diz, *C Plus Plus.* O C++ começou com a nomenclatura de C with classes (em português: C com classes). A linguagem começou com testes de implementar a orientação a objeto, mas por muito tempo foi apenas vista como um superconjunto de C. Até que em 1999, uma nova ISO, tornou a linguagem independente de C. O C++ ganhou popularidade mais tarde por sua boa performance em jogos e nos sistemas da Microsoft.

A programação orientada objeto foi um importante marco na programação, pois a partir dela, puderam se fazer softwares melhores, e com menor custo. A POO possibilitou que os módulos da programação estruturada se conversassem e assumissem formas, alterando dados, executando funções e etc (GUDWIG, 1997). Este tipo de programação revolucionou o meio, e possibilitou a existência de programas que são comuns hoje em dia.

Na década de 1990, os programadores tinham um novo desafio, o advento da internet, e as linguagens tinham que se adequar aos novos padrões impostos pela indústria, o que fez com que novas linguagens tivessem que ser desenvolvidas para suportar o que ficou conhecido em inglês como RAD, que é uma sigla para em português “aplicações de desenvolvimento rápido”. Nesse período nasceram importantes linguagens que viriam a se tornar ferramentas importantes no trabalho de muitas pessoas. O Java, foi a mais conhecida delas, por apresentar um novo conceito de programação orientada objeto, o Java tinha como proposta as interfaces gráficas e resultados impressionantes, mas ganhou fama de lenta e pesada.

Também nessa época, houve a criação do Python, que tem como principal objetivo a clareza e beleza do código. Muitos acham que o nome é em homenagem ao monty Python, porém, o nome tem origem de um parque da cidade onde a linguagem foi criada que tinha Python no nome. A linguagem é orientada objetos e ganhou mais popularidade atualmente, por ser de fácil entendimento e de com um código limpo, o Python vem se tornando uma opção para novos programadores que optam por querer ter um ótimo futuro no meio.

Existiam várias linguagens, mas um criador queria unir várias características de várias linguagens em uma só, e assim nasceu o Ruby. Tão poderosa quanto as linguagens mais usadas e mais orientados objetos do que o recém-criado Python, era lema do criador da linguagem, Yukihiro Matsumoto. O Ruby se tornou mais popular em 2005 com a criação de um framework[3] chamado Ruby on Rails que fazia o Ruby se tornar WEB. Apesar de tudo, Ruby puro é bem parecido com Python em questão de uso, tendo a mesma base orientado objeto. A diferencial é o On Rails, que torna a ferramenta única.

Com a internet se popularizando, o PHP foi criado como um pacote de desenvolvimento de páginas pessoais, substituindo o Perl, que era utilizado na época, o PHP tinha como objetivo trabalhar junto ao HTML gerando conteúdo dinâmico na internet. Uma das atualizações mais importantes do PHP, PHP3, foi lançada em 1997 que trazia suporte ao SQL e também a orientação objeto, mas a versão foi logo abandonada para criação de uma nova, PHP4, que permitia a cópia de objetos, mas não contava com funções importantes. Então foi lançada a mais popular e duradoura versão do PHP, o PHP 5, foi a versão mais utilizada e após anos de desenvolvimento foi substituída pela atual PHP7.

Derivado do C e C++, o C# tinha uma proposta totalmente diferente dos seus irmãos mais velhos, a linguagem propunha uma escrita mais simples e didática, mas ao mesmo tempo extremamente poderosa em quesitos de abrangência, adotada pela Microsoft como linguagem em seus sistemas, junto ao Visual Studio[4]. A linguagem ganhou poder e hoje é uma das principais em desenvolvimento de aplicações do Office entre outros.

Apesar de não ser considerado uma linguagem de programação propriamente dita, o HTML ganhou muito espaço nos últimos anos, por ser de fácil entendimento, a chamada linguagem de estilização é o que dá a cara para os sites, a sua integração com o PHP, faz com

[3] É uma aplicação desenvolvida para facilitar o uso de uma linguagem.
[4] Programa de desenvolvimento de software criado pela empresa Microsoft.

que se torne uma das ferramentas mais poderosas no mundo Web, presente na maioria dos sites famosos como Facebook, Youtube e Google.

Com o lançamento do iPhone em 2007, criou-se um novo método de programação, uma nova plataforma para se programar, os dispositivos móveis. Isso trouxe um novo desafio para a comunidade de programadores, desenvolver aplicativos para um dispositivo totalmente novo com uma experiência totalmente nova. A Apple, criou uma linguagem de programação própria, chamada Swift, porém deixou ela em segredo até 2014, quando lançou como componente principal de desenvolvimento de seus dispositivos, e desde então vem sendo usada e descoberta pelos amantes e desenvolvedores da empresa.

Os dispositivos móveis não ficaram somente por parte da Apple, houve também uma ideia comprada pelo Google, o Android, que tem sua base no Linux, e seus aplicativos são desenvolvidos em Java, e desde então tem sido a principal linguagem da plataforma. O Google lançou um kit de desenvolvedor chamado Android Studio, onde os desenvolvedores escrevem os códigos em Java para poderem rodar nos dispositivos Android. Contudo, a plataforma também aceita Python, Go, e outras linguagens como ferramentas de desenvolvimento.

Atualmente, o Android não está somente presente em smartphones, mas também em televisões, carros, vestíveis, entre outros equipamentos, o que torna o leque de opções para se programar muito maior.

Derivado do Windows da Microsoft, o Windows Phone foi uma tentativa da empresa de entrar no meio dos dispositivos móveis, mas seus dispositivos não conquistaram o público tanto quanto os da Apple ou Google. A Microsoft desenvolvia seus aplicativos principalmente em C#, a linguagem era poderosa no sistema, mas com a má aceitação do Windows Phone pelo público, a plataforma foi descontinuada.

PROCEDIMENTOS METODOLÓGICOS:

A pesquisa foi realizada a partir da leitura de artigos e sites em geral da internet, todos listados nas referências deste artigo. Todos os materiais utilizados reforçam os conhecimentos gerais e específicos das linguagens de programação aqui apresentadas e estudadas.

Primeiramente, foi feita uma pesquisa dos tipos de paradigmas de programação existentes até hoje, depois foram selecionados os mais conhecidos e utilizados. O resultado da seleção foram os seguintes paradigmas: imperativo/procedural, orientado à objeto,

estruturado e multiparadigma. Após, foi feita uma verificação de quais linguagens de programação se encaixariam nos paradigmas selecionados, e foi feita uma seleção com base na sua popularidade, importância na história e características específicas e/ou diferentes. O resultado da seleção das linguagens por paradigma foi o seguinte: para imperativo/procedural Cobol; para estruturada C e Pascal; para orientada à objeto Java, Ruby e PHP; e por fim para multiparadigma C++ e Python.

Estas linguagens foram analisadas durante todo o artigo, incluindo a implementação de códigos para a compreensão mais aprofundada das informações. Todo o estudo das linguagens foi feito desde a sua história até a sua sintaxe básica, com exemplos práticos na seção análise e discussão dos resultados. Em Relação aos códigos, foram construídos dois algoritmos, um deles realizava uma exibição de uma frase na tela "Hello World!!"[5], e o outro realizava a soma de dois números inteiro e exibia o resultado na tela.

Para realizar as análises das linguagens aqui citadas, será feita um comparativo das vantagens e desvantagens. E posteriormente foram construídos dois gráficos no programa Excel (2013), interpretando os resultados obtidos a partir dos códigos. Os códigos foram implementados no programa de editor de texto Sublime Text 3 (2017).

ANÁLISE E DISCUSSÃO DOS RESULTADOS:

Com os dados obtidos pela pesquisa realizada, foi realizado um resumo das características principais, citando as vantagens e desvantagens de algumas linguagens e paradigmas de programação. Começando com Pascal, que foi criada para ser uma linguagem que evitaria erros comuns na programação, possui uma programação baseada em blocos e um rígido sistema de controle das variáveis. Com o passar dos anos, ela foi sendo ultrapassada, porém recentemente foi modernizada, ganhando outro nome: Delphi.

Já Cobol, foi desenvolvida com a intenção de ser uma linguagem puramente comercial; suas características principais são: rápido acesso e atualização dos arquivos de dados, geração de muita informação, e as saídas dos sistemas são de fácil compreensão ao usuário.

A linguagem de programação C é uma das linguagens mais antigas que ainda são utilizadas e estudadas nos dias de hoje. Suas características marcantes são: portabilidade,

[5] "Hello World", em português significa Olá Mundo. É o mais famoso programa de computador realizado pelos programadores e estudantes da área.

compilação separada, recursos de baixo nível, geração de código eficiente, confiabilidade, simplicidade e facilidade de uso.

C++ se tornou uma linguagem extremamente forte e presente no mercado de trabalho, pelo fato de manter o código imperativo procedural do C (compatibilidade com C) acrescentando os conceitos da orientação à objeto. É uma linguagem que precisa de um compilador para ser executada. Possui uma 'tipagem' forte, o que significa que as variáveis precisam ter seu tipo declarado junto com a sua criação, além de que não é possível fazer certos tipos de operações, como por exemplo soma de um inteiro com um booleano.

O grande sucesso da linguagem Java se deve principalmente a características como: uma linguagem simples, um programador que começaria a aprender Java não demoraria muito tempo para produzir bons resultados com a codificação. Uma linguagem totalmente orientada à objeto, aplicando bem todos os conceitos desse paradigma. Familiaridade com outras linguagens conhecidas, como o C++. E o mais importante, a sua portabilidade de poder ser executada em qualquer hardware, pois sua compilação ocorre previamente para depois ser interpretada por uma máquina virtual (por esse motivo ela pode ser chamada de uma linguagem híbrida), que é instalada juntamente com o programa principal.

PHP é uma linguagem interpretada, por isso sua execução é mais rápida, sem a necessidade de ser compilada ou recompilada. Possui o mesmo conceito de multiparadigma do C++, com características da orientação à objeto, além das características funcionais e procedimentos. E por fim a sua característica mais marcante, possuir o padrão "de facto" (ZAPALOWSKI, 2011); que nada mais é a não existência de um padrão que está presente em todas as implementações em PHP, ou seja, possibilita a diferença de qualquer tipo de diferença entre cada implementação.

Python é uma linguagem interpretada, com uma legibilidade bem alta fazendo que seja mais fácil de ser estudada por novos programadores, pelo fato da codificação parecer muito mais com a linguagem natural. E com a possibilidade de ser trabalhada com os paradigmas de orientação à objeto, imperativa e funcional, é também uma linguagem multiparadigma.

Ruby possui características parecidas com Python, com por exemplo o multiparadigma e ser uma linguagem interpretada. Mas também possui características que as diferem, no caso, Ruby possui o padrão "de facto" (ZAPALOWSKI, 2011) com uma grande

variedade de implementações diferentes, e a sua tradução para a linguagem de máquina possui muitos métodos diferentes.

Já em relação aos paradigmas, o paradigma imperativo/procedural foi o primeiro criado na história e também o mais utilizado atualmente, ele é baseado em estados e ações que manipulam esses estados (modelo de Von Neumann). Exemplos de linguagens: C, Pascal, Algol e Python. As suas vantagens em relação aos outros paradigmas são: eficiência, modelagem "natural" das suas aplicações dos problemas reais, é um paradigma dominante, e flexível. As desvantagens são: pouca legibilidade, instruções focadas no "como" e não no "o que".

O estruturado é geralmente utilizado em quem está começando a programar, por possuir três estruturas simples de entender: sequência, decisão e iteração. Esses conceitos são fortemente usados na programação linear, e na orientação à objeto, que também gradativamente foi a substituindo na produção de software em geral. Exemplos de linguagens: C, Pascal e Cobol. As vantagens são a legibilidade, fácil compreensão da sua estrutura, e a capacidade de dividir os problemas em subproblemas. Desvantagens: dados separados das funções, sistemas difíceis de manter, e mudanças na estrutura dos dados gera mudanças em quase os todos códigos.

O conceito de orientado à objeto é um dos mais atuais e utilizados nos dias de hoje; consiste na composição e interação entre unidades chamadas objetos, basicamente todas as variáveis são tratadas como objetos, que possuem atributos (estados), métodos (comportamentos) e o funcionamento do software se dá pela troca de mensagens entres os objetos. Exemplos: Python, Ruby, C++, Java e PHP. As vantagens são: as mesmas do paradigma imperativo, além de outras, como a alteração de um módulo não ter interferência em outros módulos; e também módulos mais independentes são mais fáceis de serem reaproveitados. Desvantagens: possui um raciocínio complexo, tornando difícil o trabalho de interpretação de alguns programadores.

Figura 1: Código "Hello World" em C

```c
#include <stdio.h>

int main(void){
    printf("Hello World!!\n");
    return 0;
}
```

Fonte: Elaboração Própria

Figura 2: Código "Hello World" em C++

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

int main(void){
    cout << "Hello, World!!";
    return 0;
}
```

Fonte: Elaboração Própria

Figura 3: Código "Hello World" em Java

```java
public class Helloworld{
   public static void main(String[] args) {
      System.out.println("Hello World!!");
   }
}
```

Fonte: Elaboração Própria

Figura 4: Código "Hello World" em Python

```python
print('Hello Word!!')
```

Fonte: Elaboração Própria

Figura 5: Código "Hello World" em Ruby

```ruby
puts "Hello World!!"
```

Fonte: Elaboração Própria

Figura 6: Código "Hello World" em PHP

```php
<?PHP
echo "Hello World!!";
?>
```

Fonte: Elaboração Própria

Figura 7: Código "Hello World" em COBOL

```
IDENTIFICATION DIVISION.
PROGRAM-ID. HELLO-WORLD.
PROCEDURE DIVISION.
DISPLAY 'Hello world!!'.
STOP RUN.
```

Fonte: Elaboração Própria

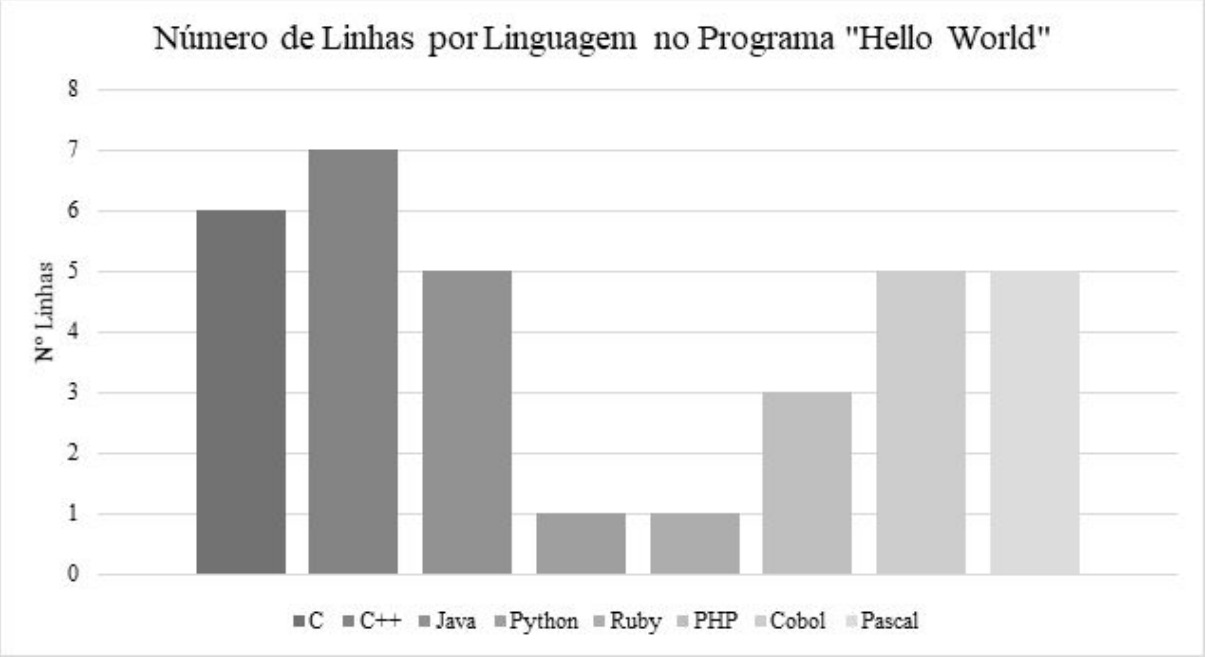
Figura 8: Código "Hello World" em Pascal

```
program HelloWord;

begin
    WriteLn('Hello World!!');
end.
```

Fonte: Elaboração Própria

Gráfico 1: Números de Linhas Por Linguagem no Programa "Hello World"



Fonte: Elaboração Própria

Figura 9: Código "Soma Simples de Dois Números" em C

```c
#include <Stdio.h>

int main(void){
    int a, b, r;

    printf("Digite dois numeros para soma: \n");
    scanf("%d", &a);
    scanf("%d", &b);
    r = a + b;
    printf("Resultado da soma: %d\n", r);

    return 0;
}
```

Fonte: Elaboração Própria

Figura 10: Código "Soma Simples de Dois Números" em C++

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

int main(void){
    int a, b, r;

    cout << "Digite dois numeros para soma: \n";
    cin >> a;
    cin >> b;
    r = a + b;
    cout << "Resultado da soma: " << r << endl;

    return 0;
}
```

Fonte: Elaboração Própria

Figura 11: Código "Soma Simples de Dois Números" em Java

```java
import java.util.Scanner;

public class Soma{
    public static void main(String[] args){
        Scanner entrada = new Scanner(System.in);
        int a, b, r;

        System.out.println("Digite dois numeros para soma:
        a = entrada.nextInt();
        b = entrada.nextInt();
        r = a + b;
        System.out.println("Resultado da soma: " + r);
    }
}
```

Fonte: Elaboração Própria

Figura 12: Código "Soma Simples de Dois Números" em Python

```python
print("Digite dois numeros para soma: ")
a = int(input(''))
b = int(input(''))
r = a + b
print("Resultado da soma: {}" .format(r))
```

Fonte: Elaboração Própria

Figura 13: Código "Soma Simples de Dois Números" em Ruby

```ruby
puts "Digite dois numeros para soma: "
a = gets
b = gets
r = a.to_i + b.to_i
puts "Resultado da soma: #{r.to_i}"
```

Fonte: Elaboração Própria

Figura 14: Código "Soma Simples de Dois Números" em PHP

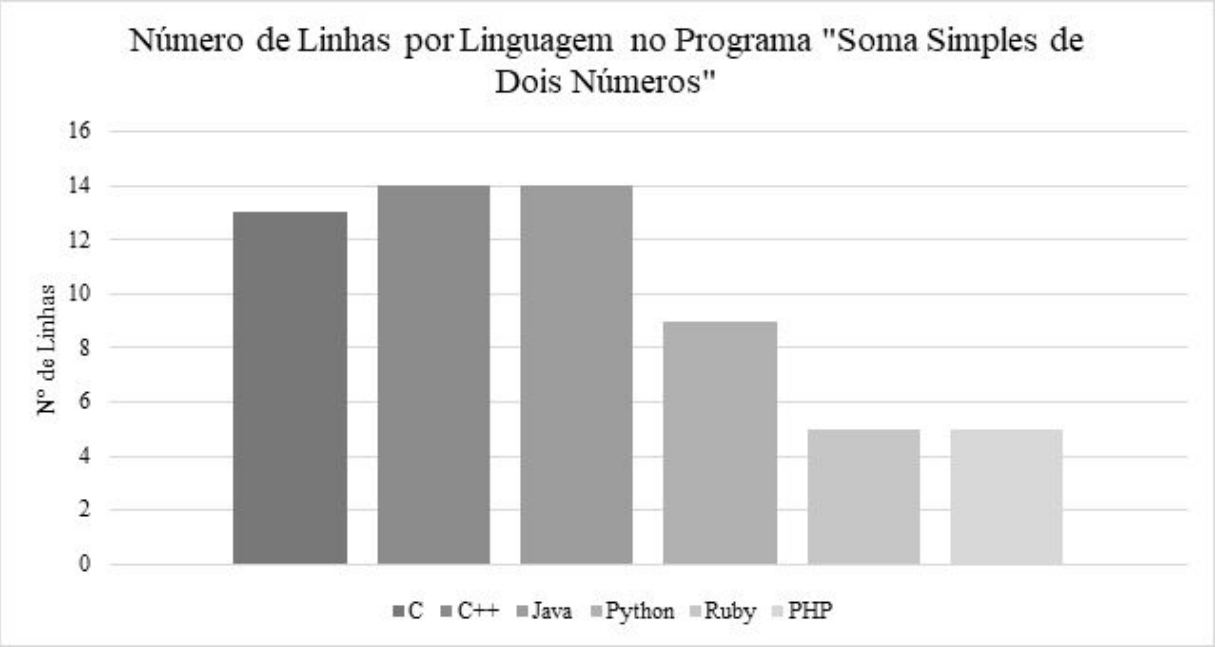
```php
<?php

echo "Digite os numeros para soma: \n";
$a = readline('');
$b = readline('');
$r = $a + $b;
echo "Resultado da soma: " . $r, "\n";

?>
```

Fonte: Elaboração Própria

Gráfico 2: Número de Linhas Por Linguagem no Programa "Soma Simples de Dois Números



Fonte: Elaboração Própria

Os resultados das quantidades de linhas em cada um dos dois algoritmos foram demonstrado no gráficos 1 e 2, o entendimento se dá de forma mais clara e concisa, e além de que a verificação pode ser realizada com as observações do algoritmos nas próprias linguagens nas figuras 1 a 14 do artigo.

No gráfico 1, podemos observar que para a resolução deste problema houve uma grande diferença do número de linhas de código entre as linguagens. Apesar do problema ser extremamente simples, não houve uma grande semelhança entre as sintaxe, pelo fato de muitas linguagens precisarem incluir bibliotecas no início do código, como por exemplo Java e C. Entretanto, podemos observar como uma linguagem de alto nível, bem construída, pode ser uma aliada do programador. Python e Ruby, linguagens de alto nível, foram recentemente criadas, apenas precisaram de uma linha para resolver um problema simples; totalmente diferente de C++, mesmo também sendo uma linguagem de alto nível, precisou de 7 linhas para resolver o mesmo problema. Outra questão que podemos observar, é a diferença entre uma linguagem que precisa de um compilador e outra que utiliza um interpretador; as compiladas geralmente possuem mais linhas de códigos justamente pela sintaxe necessária, ao contrário das interpretadas que só fazem a chamada da função que exibi na tela.

E no gráfico 2 é possível observar que a resolução deste problema também houve uma diferença do número de linhas de código entre as linguagens. O problema da soma simples não é complexo de ser resolvido, porém as diferenças apareceram praticamente pelos mesmos motivos discutidos no gráfico 1.

CONSIDERAÇÕES FINAIS:

Com os gráficos obtidos é possível verificar que as linguagens com o maior número de linhas foi o C++, no gráfico "Hello World!!" e da soma simples, e Java, no gráfico da soma simples. As duas linguagens são do paradigma orientado à objeto; porém, isso não parece ser um influenciador de quantas linhas o código vai ter porque, em último lugar, ficaram Python e Ruby, no gráfico "Hello world!!", e Ruby e PHP no gráfico da soma simples. Todas elas também são linguagens orientadas à objeto. Um fato que pode ser afirmado como um influenciador nas linhas de código é como a linguagem foi construída. Algo que pode acrescentar linhas é a inclusão de bibliotecas no início do código, como ocorre em Java, C++ e C, e também se uma linguagem é compilada ou interpretada.

Os resultados obtidos concordam com o trabalho realizado por ZAPALOWSKI, 2011, um dos resultados obtidos, na resolução do problema "Hello World!!", foram próximos com os apresentados neste artigo.

O trabalho realizado possui grande utilidade para quem está começando a estudar programação, pelo fato de ser uma ótima base introdutória sobre algumas das centenas de linguagens de programação existentes. É possível compreender como uma linguagem de alto nível influencia no desenvolvimento do software, as vantagens dos diferentes paradigmas de programação, saber quais linguagens são mais amigáveis para quem está iniciando, e quais são as mais complexas utilizadas nos projetos de grandes sistemas.

REFERÊNCIAS:

ANDROID STUDIO. **Android Studio o IDE oficial do android**. Disponível em: <goo.gl/qtMgwz> Acesso em: 1 dez 2017.

ANOTAÇÕES DE HERCULES. **Tipagem fraca/forte e interferência de tipo**. Disponível em: <goo.gl/2drL9M> Acesso em: 30 nov 2017.

APPLE. **Swift.** Uma linguagem aberta e poderosa para todo mundo criar apps incríveis. Disponível em: <https://www.apple.com/br/swift/> Acesso em: 1 dez 2017.

CASAVELHA, Eduardo. Breve história da linguagem C. **Intellectuale**. Disponível em: <goo.gl/59bC3n/> Acesso em: 1 dez 2017.

COIMBRA, Everton. **A evolução da linguagem de programação C#**. Disponível em: <goo.gl/FsKJmw> Acesso em: 1 dez 2017.

FELICIANO, P.; LAMEGO, G. C. **Linguagem de programação C++.** Disponível em: <goo.gl/whhtaU> Acesso em: 19 nov 2017.

INTRODUÇÃO à linguagem C. UNICAMP. Gerência de atendimento ao cliente, centro de computação. Disponível em: <goo.gl/rvLiRY> Acesso em: 20 nov 2017.

JUNGTHON, G.; GOULART, C. M. **Paradigmas de programação**. Taquara. Disponível em: <goo.gl/UQF12W> Acesso em: 22 nov. 2017.

LAUER, D. Comparação entre linguagens de programação. **Scribd**, Curitiba, 2008. Disponível em: <goo.gl/pE9GuZ> Acesso em: 17 nov 2017.

MAZZOLA, V. B. **Características básicas da linguagem Pascal**. Disponível em: <goo.gl/nEEqt2> Acesso em: 19 nov 2017.

MICROSOFT. **How to create your first app for Windows Phone 8**. Disponível em: <goo.gl/tSwrWr> Acesso em: 1 dez 2017.

VENNERS, Bill. **Artima**, 13 jan 2003. Diponível em: <goo.gl/g5vQcE> Acesso em: 1 dez 2017.

VISUAL STUDIO. Home page. Disponível em: <goo.gl/YXDFwY> Acesso em: 10 dez 2017.

WEBINSIDER. **A importância de aprender programação.** Disponível em: <goo.gl/QP9Ajm/> Acesso em: 26 nov 2017.

WIKIPEDIA. **Ada Lovelace**. Disponível em: <goo.gl/Bk3PPR> Acesso em: 28 nov 2017.

WIKIPEDIA. **COBOL**. Disponível em: <https://pt.wikipedia.org/wiki/COBOL> Acesso em: 29 nov 2017.

WIKIPEDIA. **Framework**. Disponível em: <goo.gl/fQ4bPQ> Acesso em: 10 dez 2017.

WIKIPEDIA. **História das linguagens de programação.** Disponível em: <goo.gl/bRyup7> Acesso em: 1 dez 2017.

WIKIPEDIA. **Java** (linguagem de programação). Disponível em: <goo.gl/MxHJty> Acesso em: 23 nov 2017.

WIKIPEDIA. **Linguagem de programação**. Disponível em: <goo.gl/m26PJL> Acesso em: 28 nov 2017.

WIKIPEDIA. **PHP**. Disponível em: <goo.gl/4tMUxk> Acesso em: 30 nov 2017.

ZAPALOWSKI, Vanius. **Análise quantitativa e comparativa de linguagens de programação.** 2011. 45 f.Trabalho de conclusão de curso - Faculdade de Ciência da computação, UFRGS, Porto Alegre, 2011.

ZENZELUK, J. H.; RIBEIRO, S. **Um estudo para a evolução do PHP com a linguagem orientada a objeto**. Faculdade Guairacá. Disponível em: <goo.gl/WRDcNm> Acesso em: 18 nov. 2017.