UNIVERSIDADE DO ESTADO DE SANTA CATARINA - UDESC

CENTRO DE EDUCAÇÃO DO PLANALTO NORTE - CEPLAN

BACHARELADO EM SISTEMAS DE INFORMAÇÃO

INTERPRETADOR DE ALGORITMOS PORTUGOL EM HTML5

São Bento do Sul, SC

2016

GILBERTO EDMUNDO TAVARES

INTERPRETADOR DE ALGORITMOS PORTUGOL EM HTML5

Trabalho de Conclusão apresentado ao Curso de Bacharelado em Sistemas de Informação, da Universidade do Estado de Santa Catarina, como requisito para a obtenção do grau de Bacharel em Sistemas de Informação

Orientador: Dr. Luiz Cláudio Dalmolin

São Bento do Sul, SC

2016

GILBERTO EDMUNDO TAVARES

INTERPRETADOR DE ALGORITMOS PORTUGOL EM HTML5

Trabalho de Conclusão apresentado ao Curso de Bacharelado em Sistemas de Informação, da Universidade do Estado de Santa Catarina, como requisito para a obtenção do grau de Bacharel em Sistemas de Informação

Banca Examinadora

Orientador:

_____

Dr. Luiz Cláudio Dalmolin
Universidade do Vale do Itajaí

Membros:

_____

(titulação + nome completo)
IES de origem

_____

(titulação + nome completo)
IES de origem

São Bento do Sul SC, 24/11/2016

Dedico este aos meus familiares falecidos:
Meu irmão que sempre apoiou minha formação superior e carreira até oferecendo certa colaboração financeira.
E ainda mais recente meu pai que financiou meu pré ingresso e não pude compartilhar com ele esta conquista.

AGRADECIMENTOS

A minha mãe me apoiou em minha decisão de cursar faculdade em outra cidade. E sempre que possível, mesmo com dificuldade, ajudou como pode com móveis para meu primeiro quarto, algumas compras e mercado e algum dinheiro quando ficava mais crítico. Inclusive isso tudo pode ter sido um dos motivos da decisão dela em vender o imóvel em morávamos.

A toda a família Lischka que me acolheu inicialmente como hóspede na residência do casal Arnaldo e Jacira (prima da minha mãe). Foi também na empresa familiar deles juntos com seus filhos que tive meu primeiro emprego na cidade e me cedido para moradia o quartinho e dependências junto à empresa.

Ao meu irmão que me empregou as sábados e minhas cunhadas que me hospedavam, além de amigos. A todos os motoristas que me deram carona para que eu pudesse voltar com uma grana a mais do final de semana, economizando a passagem.

A Joseli que sua experiência passando pelo menos foi de grande ajuda. Tambm seu apoio e motivação, bem como dos meu amigos de infância José e Rafael que fizeram o mesmo.

A toda a experiência social, esportiva e de conheciento que as confraternizações com outros academicos, Jiudesc e os vários Lantinoware proporcionaram.

“A maioria dos bons programadores programa não por esperar ser pago ou adulado pelo público, mas porque é divertido programar.” (tradução nossa)

Linus Torvalds

RESUMO

Palavras-chaves: Lógica da programação. Algoritmos. Portugol. Interpretador. HTML5.

ABSTRACT

Key-words: Programming logic. Algorithms. Portugol. Interpreter. HTML5.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

HTML	*HyperText Markup Language*
UAL	UNESA *Algorithmic Language*
UNESA	Universidade Estácio de Sá

SUMÁRIO

1 INTRODUÇÃO

A base da programação é ensinar máquina a realizar determinada tarefa, para que isso seja possível é necessário que o usuário comunique-se com a máquina, numa linguagem que ela entenda, a linguagem de programação. A construção da linguagem de programação envolve ações definidas em passos que se deseja que a máquina execute, a estes passos damos o nome de algoritmos. Para isso pode-se dizer que independente do paradigma utilizado no processo de desenvolvimento ele envolve algoritmos (MEDEIROS, 2015).

Algoritmos podem não necessariamente serem computacionais, já que são um conjunto de passos finitos. Analogias simplistas são os ingredientes e modo de preparo de uma receita ou instruções para uma tarefa do cotidiano como a troca de um pneu furado em um automóvel (MEDINA; FERTIG, 2006). Porém se tratando de computação deve ter maior formalidade, seguindo uma linguagem pois diferente do ser humano uma máquina não é tão interpretativa. Desconsideradas aqui inteligências artificiais que são desenvolvidas por algoritmos para terem comportamento de compreensão humana. Isto pode ser analogamente algo como não entender idioma estrangeiro e as instruções estarem nesta linguagem. Porém em informática isso tem que ser mais especificado com palavras ou caracteres predeterminados de início e fim, de blocos, repetições, etc (DERSHEM; JIPPING, 1990).

Sendo o berço mais popular da informática os Estados Unidos da América e seus criadores em muitos casos dessa origem ou de países com mesmo idioma as primeiras linguagens de computação e a maioria delas são em inglês (SEBESTA, 2009).

Isso muda quando falamos do Portugol que possui variações, mas resumidamente sua origem vem como uma tradução de uma linguagem de programação simples e comum, com seus termos no idioma nativo do Brasil.

Tendo como foco o aprendizado de algoritmos para uma boa fundamentação de futuros programadores ou profissionais da área, e muitas vezes a facilidade com o inglês não é algo comum em nosso país, a linguagem Portugol se torna um artifício vantajoso (JESUS et al., 2004).

1.1 OBJETIVOS DO TRABALHO

1.1.1 Objetivo Geral

Apresentar o conceito de ferramenta computadorizada para se tornar referência em criação, edição, visualização e interpretação de algoritmos em pseudolinguagem, como alternativa às existentes. Sendo seu diferencial o acesso nativo a partir de navegadores *Web* modernos. Inclusive acessível quando uma conexão com a internet esteja indisponível.

1.1.2 Objetivos Específicos

Visando atingir o objetivo geral em sua totalidade lista-se a seguir objetivos específicos de modo sucinto:

- Desenvolver o protótipo, sendo seu código fonte aberto e disponível para que outros possam manter, ampliar e evoluir a aplicação;
- Fazê-lo compatível com a sintaxe desenvolvida na Universidade Estácio de Sá (UNESA) a UNESA *Algorithmic Language* (UAL). E mínimo produto viável para a primeira lição de ensino aprendizagem;
- Permitir o acesso via navegadores *Web* modernos, sem necessidade de instalações complementares;
- Aplicar aprimoramentos introduzidos na versão 5 da *HyperText Markup Language* (HTML) para uso sem conexão com a internet.

1.2 JUSTIFICATIVA DO TRABALHO

Praticar em *software* à abstração em papel auxilia e muito no processo de ensino e aprendizagem, em algoritmos além do pensamento lógico por etapas já pode ser desenvolvido e descrito para o repasse a um computador. “O Aprendizado de algoritmos não é uma tarefa muito fácil, só se consegue através de muitos exercícios” (LOPES; GARCIA, 2002, p. 1).

Para esse ensino é essencial conhecer dois conceitos principais lógica de programação e algoritmos. Primeiramente é apresentado um paralelo de situações do cotidiano com algoritmos, ao efetuar comparações de instruções passo a passo realizadas diariamente, como fritar um ovo ou trocar um pneu de automóvel. A seguir inicia-se a aplicação de linguagem estruturada em português para descrição dos algoritmos. Em alguns casos, utiliza-se de linguagens em inglês, mo por exemplo, Pascal. Há professores que preferem usar pseudo linguagem em português somente “em papel”, iniciando a codificação diretamente em linguagem C. Tem-se inclusive, também “em papel”, de analisar e interpretar os passos do algoritmo para saber quais serão os valores em memória e a saída na tela.

Simultaneamente, com a prática escrita, sem software específico, pode-se simular os mesmos algoritmos em interpretador ou compilador. Sendo em um interpretador sua construção dividida em partes, cada uma com uma função específica. Geralmente identifica-se: o analisador léxico, o analisador sintático, o analisador semântico, e o resultado de saída. Caso seja um compilador há, ao invés de resultado de saída, o gerador de código, que transforma o programa em linguagem de máquina composta de 0s e 1s (DELAMARO, 2004).

Ao utilizar um programa em navegador via Internet, fazendo com que seja desnecessário instalar e utilizar sempre o mesmo computador, objetiva-se que os exercícios serão resolvidos com maior facilidade. Também é importante que as mensagens de erro sejam claras, e em português já que torna mais acessível a leitura para quem ainda não possui facilidade com a língua inglesa.

1.3 LIMITAÇÕES DO TRABALHO

Limitou-se o trabalho numa imposição que vem preservar que o tempo disposto seja viável para sua execução.

Por seguir um processo bem diferenciado de construção optou-se por sua mínima execução. Permitindo apenas algorítmos muito simples pois um dos focos é a disponibilidade independente de sistema operacional ou instalações.

Outra limitação foi a não possibilidade de gravar e manter executáveis da saída gerada do algoritmo. Devido ao entendimento que é ponto não essencial para a prática do conhecimento e que uma possível solução no modelo pretendido seria dispendioso e demorado.

Devido a depender de estrutura adicional como servidores e hospedagem além de programação adicional desta interface também não há a gravação automática dos códigos para acesso irrestrito. Cabendo ao salvamento e abertura pela máquina. Levou-se em conta que há vários serviços para tal atividade com no caso cabendo ao usuário.

O sistema de tratamento de erros foi explorado até onde complexidade dos algoritmos possíveis dentro do proposto permitiu. Realizados assim neles os testes de erros possíveis.

1.4 ESTRUTURA DO TRABALHO

O presente trabalho está estruturado em cinco capítulos que abordam o Processo de Ensino e Aprendizagem comum na atualidade e as Ferramentas Utilizadas, detalhes sobre o ensino com a utilização de Processamento de Linguagem Natural em Algoritmos.

Após o primeiro o anterior introdutório o capítulo dois descreve principais termos tem sua teoria descrita além de apresentar as principais ferramentas utilizadas no ensino e prática de lógica de programação.

Os capítulos seguintes apresentam no três a metodologia utilizada e no quatro aplicação em seu desenvolvimento até o seu resultado, apresentando alguns temos pertinentes somente ao processo.

Por fim o capítulo cinco conclui-se com as considerações finais, recomendações, sugestões e trabalhos futuros, podendo ser pelo autor ou outros interessados.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 ALGORITMOS

A última das notas de Ada Lovelace, que foram republicadas em 1953, apresenta os primeiros conceitos sobre programação, descrevendo um algoritmo. Desde então estes foram difundidos, sendo utilizados até a atualidade (SANTIAGO; DAZZI, 2003).

2.1.1 Ensino Aprendizagem

2.2 PSEUDOLIGUAGEM E PORTUGOL

2.3 FERRAMENTAS EXISTENTES

- Portugol/Plus
- Visual Alg
- G-Portugol
- Portugol IDE
- Portugol Studio
- ASA
- ATMUF
- AWTM
- AMBAP
- CIFluxProg
- RAFF
- SistLog
- Ambiente SICAS
- C-Tutur
- PL-Detective

2.3.1 UAL e Editor UAL

2.3.2 Demais Aplicativos

2.3.3 Ambiente Virtual de Aprendizagem - AVA

Udacity

Codeacademy

SoloLearn

Code School

Kan Academy

2.3.4 Programação em Blocos

Blockly

2.4 *ONLINE JUDGE*

Sistema de Apoio a Competições de Programação é a denominação em português de *Online Judge* por Campos e Ferreira (2004), nomeado como BOCA. E mais recente foi desenvolvido por acadêmico da URI (Universidade Regional Integrada do Alto Uruguai e das Missões) com base nesse trabalho anterior o URI *Judge Online* (TONIN; BEZ, 2012).

2.5 TECNOLOGIAS WEB

2.5.1 Linguagem de Marcação de Texto

2.5.2 Liguagem de Programação para Web

ECMAScript

WebAssembly

2.5.3 Navegadores Web

2.5.4 Aprimoramentos para Uso *Offline*

2.6 COMPILADORES E INTERPRETADORES

2.6.1 Análise Morfológica

2.6.2 Análise Léxica

2.6.3 Tokenização e Parseamento

2.6.4 Análise Sintática

2.6.5 Análise Semantica

3 MÉTODOS DE PESQUISA

3.1 ESTUDO DA FERRAMENTA EDITORUAL

3.2 ESTUDO DA OUTRAS FERRAMENTA RELACIONADAS

3.3 DEFINIÇÃO DE REQUISITOS

3.4 DESENVOLVIMENTO

3.4.1 Interface Gráfica

3.4.2 Módulo de Classificação

3.4.3 Módulo de Interpretação

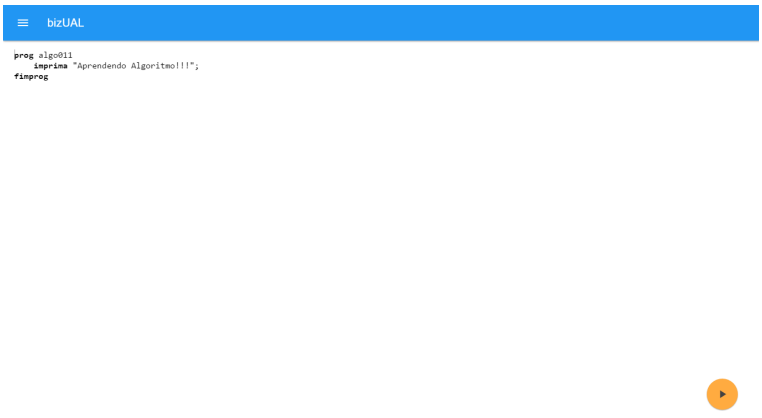
4 DESENVOLVIMENTO DA APLICAÇÃO

Essencialmente a aplicação foi desenvolvida em linguagens Web, o JavaScript sendo a linguagem de programação, HTML como linguagem de marcação e CSS como estilização. Nesse modelo já existem editores de código fonte de linguagens de programação,dos mais utilizados temos ACE e o CodeMirror, optado pelo primeiro pois em análise entendeu-se como sendo mais prático para o uso e criação do modelo da linguagem. Considerou-se a criação do modelo gramatical para destaque da sintaxe como dificuldade relativamente baixa.

Para resultar na árvore gramatical encontrou-se o ACORN um parser já escrito em JavaScript. De modo trabalhoso foi necessário adaptar o código fonte para compreender a estrutura do algorítmos proposto, principalmente devido à não utilização de parênteses para o comando de impressão em tela na linguagem UAL.

Como saída o interpretada o JS-Interpreter foi encontrado como a única opção. Ele já se utiliza do mesmo ACORN mencionado acima e interpreta a própria linguagem JavaScript. Pelo mesmo motivo citado acima, falta de parênteses no comando, se tornou uma tarefa um tanto dispendiosa fazer com que ele se adaptasse ao modelo.

Figura 1 – Interface Desktop

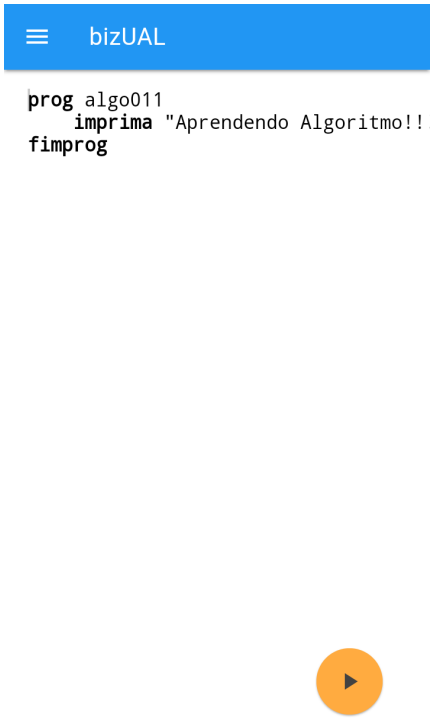


Fonte: Autor

Como interface gráfica foi mantido o mínimo necessário. Somente o editor de texto e um botão para executar. Foram visualizadas renderizações e tamanhos de telas variados, computadores (Figura 1), tablets, smartphones (Figura 2), etc. Seguindo conceitos do Google Material Design foi adicionada uma barra no topo como botão para menu o nome da aplicação, denominada Bizu, e um botão flutuante no canto inferior direito como ícone “play” para a execução do algoritmo.

Facilitando a tarefa de construção da interface foi utilizada a biblioteca Material Design Lite (MDL) disponibilizada oficialmente pelo Google. As cores foram escolhidas pela funcio-

Figura 2 – Interface Smartphone



Fonte: Autor

nalidade disponível no site da biblioteca. A escolha do nome se deve a partir da palavra bizu adicionada a sigla da sintaxe. Foi utilizado a funcionalidade app cache a partir de um arquivo manifesto listando o conteúdo que o navegador deve guardar cópia local. Podendo assim ter a internet desconectada após o primeiro acesso que a aplicação continuará acessível. Todo o código foi desenvolvido em repositório disponível no github e quando concluído em sua versão mínima movido para seu destino final e podendo ser acessado.

4.1 FERRAMENTAS UTILIZADAS

4.1.1 Editor de Código ACE

4.1.2 *Parser* Acorn

4.1.3 JS-Interpreter

4.2 PROCESSO CRIACIONAL

4.2.1 Git e Github

4.2.2 Node.js e Grunt

4.2.3 SASS

4.2.4 TDD

4.3 INTERFACE GRÁFICA

4.3.1 *Material Design*

MDL

4.3.2 Inspirações

5 CONSIDERAÇÕES FINAIS

5.1 RELAÇÃO ENTRE OS OBJETIVOS E OS RESULTADOS OBTIDOS

5.2 LIMITAÇÕES DA PESQUISA

5.3 SUGESTÕES PARA TRABALHOS FUTUROS

REFERÊNCIAS

CAMPOS, Cassio P. de; FERREIRA, Carlos E. Boca: um sistema de apoio a competições de programação. In: **XII WEI - Workshop de Educação em Computação no XXIV Congresso da Sociedade Brasileira de Computação, Salvador, UFBA - Universidade Federal da Bahia**. Salvador: Anais do XXIV Congresso da Sociedade Brasileira de Computação, 2004.

DELAMARO, Márcio Eduardo. **Como Construir um Compilador Utilizando Ferramentas Java**. 1. ed. São Paulo: Novatec, 2004. v. 1. ISBN 8575220551.

DERSHEM, Herbert L; JIPPING, Michael J. **Programming languages: structures and models**. [S.l.]: Wadsworth Publ. Co., 1990.

FUEGI, J.; FRANCIS, J. Lovelace babbage and the creation of the 1843 'notes'. **IEEE Annals of the History of Computing**, v. 25, n. 4, p. 16–26, Oct 2003. ISSN 1058-6180.

JESUS, E. A. de; SANTIAGO; DAZZI, R. L. S. R. de. Ferramenta para criação e teste de algoritmos utilizando fluxogramas ou portugol. In: **Congresso Brasileiro de Computação, 2004, Itajaí**. Itajaí: Congresso Brasileiro de Computação, 2004.

LOPES, Anita; GARCIA, Guto. **Introdução à programação: 500 algoritmos resolvidos**. Rio de Janeiro: Campus, 2002. ISBN 9788535210194.

MEDEIROS, Antônio Vinícius Menezes. Um interpretador online para a linguagem portugol. 2015.

MEDINA, Marco; FERTIG, Cristina. **Algoritmos e programação**. São Paulo: Novatec, 2006.

SANTIAGO, Rafael de; DAZZI, Rudimar Luís Scaranto. Ferramentas que auxiliam o desenvolvimento da lógica de programação. In: **XII SEMINCO - Seminário de Computação, 2003, Blumenau. Anais do XII SEMINCO**. Blumenau: Furb, 2003. p. 113–120.

SEBESTA, Robert W. **Conceitos de linguagens de programação**. Porto Alegre: Bookman, 2009. ISBN 8577808629.

TONIN, Neilor A.; BEZ, Jean Luca. Uri online judge: a new classroom tool for interactive learning. In: **WORLDCOMP'12 - The 2012 World Congress in Computer Science, Computer Engineering, and Applied Computing, 2012, Las Vegas, NV. The 2012 International Conference on Frontiers in Education: Computer Science and Computer Engineering (FECS 2012)**. USA: CSREA Press, 2012. v. 1, p. 242–246.