

LIBRAS EM UMA SOLUÇÃO COMPUTACIONAL PARA APRENDIZAGEM / LIBRAS IN A COMPUTATIONAL SOLUTION FOR LEARNING

GILSON BATISTA SOUSA JUNIOR¹
EPAMINONDAS RODRIGUES DA CRUZ JUNIOR²
NELSON BATISTA LEITÃO NETO³

RESUMO:

Esta obra apresenta um estudo sobre algoritmo, orientação a objetos, compiladores, banco de dados e especificamente a Libras. Além disso, aborda as teorias e seus processos se tratando de uma revisão teórica a respeito da Língua Brasileira de Sinais em soluções computacionais voltadas para os surdos e ouvintes que desejam aprender a Libras e enriquecer o vocabulário com intuito de adquirir conhecimento linguísticos através de dicionário e tradutor. O avanço da área de tecnologia da informação é muito importante para o aprendizado, está presente nas atividades da maioria da população brasileira, podendo representar um passo adiante na evolução da educação e do desenvolvimento de software no Brasil.

PALAVRAS-CHAVE: Língua Brasileira de Sinais, Compiladores, Banco de Dados, Computação, Tecnologia da Informação.

LIBRAS IN A COMPUTATIONAL SOLUTION FOR LEARNING / LIBRAS IN A COMPUTATIONAL SOLUTION FOR LEARNING

ABSTRACT:

This work presents a study on algorithm, object orientation, compilers, database and specifically Libras. In addition, it approaches theories and their processes if it is a theoretical revision regarding the Brazilian Language of Signals in computational solutions for the deaf and hearing who wish to learn Libras and to enrich the vocabulary with the purpose of acquiring linguistic knowledge through dictionary and a translator. The advancement of the area of information technology is very important for learning, it is present in the activities of the majority of the Brazilian population, and may represent a step forward in the evolution of education and software development in Brazil.

¹ Graduado em Ciência da Computação (Universidade Paulista); Graduando em Medicina (Universidade Federal de Goiás). Email: gilsonbatistasousajr@gmail.com.

² Licenciatura em Educação Física (Centro Universitário Araguaia); Licenciatura em Pedagogia (Faculdade Integradas de Ariquemes); Pós graduado em Libras (FAVENI); Pós graduado em Docência do Ensino Superior e Educação Física Escolar (FAVENI); Pós graduando em Psicomotricidade da Educação Física Escolar (FAVENI); Mestre em Ensino de Ciências (Universidade Estadual de Goiás); Professor da FACMAIS e da UNIP. Email: junior6285119493@gmail.com.

³ Bacharel em Ciência da Computação (UFG) e Pós-Graduado em Redes de Computadores pela Universidade Católica de Goiás. Mestre em Educação, com foco em Ensino a Distância, pela PUC-Goiás. Email: nelson.neto@docente.unip.br.

KEY WORDS: Brazilian Language of Signals, Compilers, Database, Computing, Information Technology.

INTRODUÇÃO

Na Ciência da computação, o ramo tecnológico consiste em evoluir o mundo através de dispositivos, tanto software quanto hardware. E a computação está associada ao equipamento eletrônico capaz de realizar vários procedimentos de dados como receber, armazenar, processar lógica e o principal objetivo que é resolver problemas. Através da computação, o uso da metodologia, disciplina e autoconhecimento são aplicados a programação de computadores em questão de prática. Portanto, conforme ao longo dos anos em questão de desenvolvimento de software, o computador tornou uma tarefa muito utilizada pela sociedade o que os levaram a desenvolver projetos de melhorias e inovações, dando início ao algoritmo.

ALGORITMO

A evolução da computação deu início ao desenvolvimento da lógica de programação, usando algoritmos para elaboração de projetos, estruturas e dados para fornecer informações que visa determinar a solução para os usuários (MANZANO; FIGUEIREDO, 2016).

Na Ciência da Computação o “algoritmo está associado a um conjunto de operações e regras que são ordenados e definidos que destina a solução de um problema ou um conjunto de problemas” (MANZANO; FIGUEIREDO, 2016, p. 25). O algoritmo serve para resolução de problemas, um processo sistemático que se baseia em sequenciar a ordem de passos que consiste em determinar a realização de uma tarefa (SZWARCFITER; MARKENZON, 1994; SALIBA, 1992).

A importância do algoritmo é o planejamento de projetos cujo objetivo seja satisfazer as necessidades dos clientes e usuários, desenvolvendo ferramentas eficientes visando proporcionar melhorias no cotidiano da sociedade, tais como o trabalho, a educação ou de uso pessoal.

O uso de software como ferramenta, acordo com Lafore (1999),

As ferramentas do programador é um auxílio muito necessitado na educação e nas empresas públicas e privadas. Com o auxílio é feita as estruturas de armazenamento de dados e nem todas são utilizadas para armazenagem. Os dados são acessados direto pelo usuário, entretanto, algumas estruturas não são feitas para acesso direto e sim pelo programa, a fim de facilitar a acessibilidade às ferramentas e outras operações. Esse desenvolvimento é para garantir a eficiência de trabalho relacionado á gestão e áreas afins com o intuito de melhorar e crescer a empresa (1999, p. 78).

A importância das ferramentas para os desenvolvedores é satisfazer as necessidades da empresa, tornando os trabalhos eficientes conforme a criação de projetos.

Entretanto, as ferramentas e programas de computadores são criados a base de algoritmo, que são operações e regras determinada pelo usuário, com o objetivo de tornar as tarefas cotidianas mais eficientes (MANZANO; FIGUEIREDO, 2016).

Para Cormen et al., um procedimento computacional relacionado ao algoritmo, define valores ou um conjunto de valores que consiste em entradas e produz valor ou conjunto de valores como saída. Segundo as definições proposta pelo autor, o termo algoritmo, é mais conhecido como:

A Regras formais, sequenciais e bem definido a partir do entendimento lógico de um problema a ser resolvido a partir da criação de tarefas e a execução do mesmo, com o objetivo de transformar em um programa em que os dados de entrada são transformados em dados de saída (2002, p. 03).

Infere-se, portanto, que a definição pode ter inúmeras soluções para os problemas em que, é possivelmente uma proposta duvidosa. Do ponto de vista computacional, o problema é praticamente um desvio de percurso, determinado como regras, da qual impede de atingir os objetivos com eficiência. Dessa forma, para os autores, a resolução de problemas computacionais baseado em algoritmos é aprofundar na lógica de programação com o intuito de amenizar as dificuldades no desenvolvimento de aplicações.

PROGRAMAÇÃO ORIENTADA A OBJETOS

O conceito de Programação Orientada a Objetos surgiu por volta de 1960, devido a necessidade de simulações em computadores (MANZANO; FIGUEIREDO,

2016). Segundo os autores, a orientação a objetos é uma técnica de programação que vem sendo discutida desde há muito tempo e tem ocasionado muitas dúvidas para quem desenvolve as programações.

Entretanto, para Mendes (2009), muitos alunos possuem dificuldades em entender a diferença entre classe e objeto, bem como qual o significado real de um atributo e método. Dessa forma, a orientação a objetos é o paradigma de programação onde se usam classes e objetos a partir de modelos, que representa e processam dados usando os programas de computadores (SANTOS, 2013).

Outrossim, segundo os autores, sem a técnica estruturada, há preocupações quanto a qualidade no desenvolvimento do programa, reaproveitamento de código e o tempo de desenvolver, começa a ser bastante demorado. Dessa necessidade, a técnica evoluiu e é tornado a base fundamental para a criação de programas orientada a objetos. Baseado na aplicação de abstração, a programação orientada a objetos é uma filosofia de trabalho, não sendo uma ferramenta em si, mas a forma de pensar (MANZANO; FIGUEIREDO, 2016).

Essa forma, de acordo com os autores, é a forma de usar a lógica em solucionar o problema do mundo real em um computador. Para Manzano e Figueiredo (2016), é preciso utilizar a estrutura de dados de um programa baseado na estrutura do mundo real, por isto é necessário aos desenvolvedores mudar a forma de pensar em um problema computacional.

A orientação a objeto possui fundamentos essenciais para uma boa elaboração de projetos de programação e de acordo com Manzano e Figueiredo (2016), a Orientação a Objetos é fundamentada na seguinte estrutura que são: classe, objeto, atributo e método.

Na orientação a objetos, de acordo com Manzano e Figueiredo (2016), a classe consiste em um conjunto de objetos compartilhando uma ou mais características quanto aos seus itens de dados e procedimentos associados, cujo objetivo seja formatar a estrutura de um ou conjunto de objetos, usando a estrutura para instanciar um objeto.

O objeto, de acordo com Manzano e Figueiredo (2016 p.268), “como qualquer modulo que contem rotinas e estruturas de dados capazes de interagir com outros módulos similares, trocando mensagens”. Entende-se por rotinas e troca de

mensagens, se assemelha ao uso de passagem de parâmetros por referência. Na programação de computador, objeto pode ser uma pessoa, um local ou quaisquer outros elementos do mundo real que possam pertencer à classe de categorização.

O atributo na programação é descrito como uma das propriedades que definem um objeto ou entidade (MANZANO; FIGUEIREDO, 2016). Para Manzano e Figueiredo (2016), é correspondido em um banco de dados relacional, um dos campos de registros, o que vem de encontro às classes. O atributo pode ser comparado com uma variável, como características de um ser humano e suas ações. Na linguagem de programação, a visibilidade do conteúdo de um atributo pode ser pública, privada ou protegida (MANZANO; FIGUEIREDO, 2016). O autor ainda afirma que o uso de atributo privado ou protegido, ocorre o efeito de ocultar informações das características de uma classe ou objeto, da qual é chamada de encapsulamento.

O método na lógica de programação se associa a forma de como um atributo será alterado. Para Manzano e Figueiredo (2016 p.268), “método é uma característica que possibilita a alteração da funcionalidade de um atributo”, se trata de um controle lógico, chamado de operação.

A evolução da orientação a objetos começou a partir das bases de dúvidas dos programadores da qual, os mesmos se passavam por dificuldades em compreender a diferença entre classe e objeto, entretanto, foi de importância o desenvolvimento da orientação a objetos que visa determinar os funcionamentos do programa a partir da estrutura dividida em classe, objeto, atributo e método. A orientação a objetos como faz parte do desenvolvimento de um programa, essa estrutura é trabalhada com o auxílio de um compilador (MANZANO; FIGUEIREDO, 2016; AHO; SEHI; ULLMAN, 1995).

COMPILADORES

O compilador é uma ferramenta muito necessitada e necessária para se criar um programa, toda programação de computadores que forem feitas, passa por um processo de tradução entre a linguagem fonte e a linguagem objeto de forma que,

seja possível de entender a criação do programa através do desenvolvimento e do erro que o compilador relata ao programador.

A compilação, segundo Aho, Sethi e Ullman (1995, p. 01), é “um programa que lê um programa escrito numa linguagem, a linguagem fonte e o traduz num programa equivalente numa outra linguagem, a linguagem alvo”. A linguagem alvo é variada, onde pode ser linguagem de máquina ou linguagem de programação entre um microprocessador e um supercomputador.

De acordo com os autores, os compiladores são classificados como várias passagens simultâneas e múltiplas de forma que, para que pudessem carregar e executar, depurar ou otimizar em base de construção do código-fonte que consiste em realizar as tarefas e suas funções.

Na década de 50, os compiladores foram considerados difíceis de escrever e desde então, foi desenvolvido boas linguagens de implementação e ferramentas de software (AHO; SETHI; ULLMAN, 1995). Com esses avanços de programação, os autores afirmaram que há possibilidade de um compilador ser escrito num projeto de software.

Entretanto, a importância do compilador é o principal objetivo de tornar melhor as tarefas para o usuário, tais como detalhar os erros de compilação, dividir em fases e traduzir, e, portanto, é o conjunto de instruções para executar com os resultados esperados.

COMPILADORES: ANÁLISE E SÍNTESE

Na compilação existem duas partes sendo a análise e síntese (AHO; SETHI; ULLMAN, 1995). Segundo os autores, “a parte de análise divide o programa fonte nas partes constituintes e cria uma representação intermediária. A síntese constrói o programa alvo desejado a partir da representação intermediária.” (1995, p.01).

De acordo com os autores, a fase da análise serve como base em verificar os componentes do programa-fonte, no caso de não seguir as regras e possuir erros, o compilador reporta ao programador. E de acordo com os autores, a análise é sempre realizada primeiro quando as ferramentas de software manipulam os programas fonte (AHO; SETHI; ULLMAN, 1995).

Além disso, a possibilidade de uma variável ser executada antes de ser definida depende das ferramentas de compilação, e para isso é utilizado os formatadores de texto, cujo foco principal é a entrada do fluxo de caracteres que inclui os comandos, parágrafos, estruturas matemáticas e figuras (AHO; SETHI; ULLMAN, 1995).

A fase da síntese é a construção do programa alvo em que, é gerado pela representação intermediária que é realizada pela análise. E segundo os autores, a síntese também percorre a estrutura conhecida como árvore, a partir de todos os nós possíveis em uma ordem de definição para elaborar e gerar uma linguagem de máquina.

Ambas as fases conhecidas como análise e síntese, para Aho, et al. (2008), é a divisão entre linguagem-fonte que pertence a análise e a linguagem alvo que pertence a síntese, dessa forma, trabalham juntas na elaboração de um programa.

COMPILADORES: LÉXICA, SINTÁTICA E SEMÂNTICA

O compilador na análise, segundo Aho, Sethi e Ullman (1995), é operar em três fases sendo análise léxica, análise sintática e análise semântica.

O analisador léxico ou scanner é a primeira fase do compilador que consiste em sequenciar o fluxo de caracteres que faz parte do programa fonte, com o objetivo de ler os caracteres da esquerda para direita, os agrupar em uma sequência denominada de lexemas e assim produz uma sequência de símbolos que são conhecidos como tokens (AHO et al., 2008). Os tokens, segundo Aho et al. (2008), podem ser identificadores, palavras ou operação que é atribuído a uma tabela de símbolos, durante essa fase, são ignorados os comentários e espaços em branco e assim é enviada para processamento pela análise sintática que é a segunda fase do compilador.

Segundo os autores, análise léxica, como controla o número de caracteres, também identifica a possibilidade de um token ser uma palavra reconhecida ou não pela linguagem e assim retorna uma mensagem de erro, dessa forma, reporta o usuário de corrigir os erros ao desenvolver o programa. Infere-se, portanto, que a

análise léxica tem como objetivo efetuar a validação na formação da palavra de uma linguagem.

O analisador sintático, também conhecido pelo nome de parser é a segunda fase do compilador, cujo objetivo é de agrupar partes do programa-fonte em frases gramaticais que são usadas para sintetizar a saída pelo compilador (AHO; SETHI; ULLMAN, 1995; PRICE; TOSCANI, 2001). Segundo os autores, o analisador sintático recebe uma cadeia de tokens que pertence ao analisador léxico.

Entretanto, de acordo com os autores, o analisador sintático tem como base, realizar a verificação na estrutura de um programa e relatar possíveis erros de sintaxe considerada inteligível e se recuperar dos erros. (AHO; SETHI; ULLMAN, 1995).

Para Aho, Sethi e Ullman (1995), se com frequência, o programa-fonte fosse executado corretamente e sem erros pelo compilador, seria um processo muito simplificado. Portanto, os autores afirmam que os programadores frequentemente erram uma sintaxe do programa-fonte. Dessa forma, a fim de evitar os erros e por precaução, o compilador acompanha o programador na identificação de sintaxe para localizar os erros que pode ser vários tipos de erros diferentes, tais como erros léxicos que erra um identificador sintático com expressão aritmética ou erros semânticos.

A terceira fase do compilador é a análise semântica, que consiste em assegurar que os componentes do programa fonte possam ter uma combinação de forma significativa e realiza uma verificação na estrutura hierárquica (AHO; SETHI; ULLMAN, 1995; PRICE; TOSCANI, 2001). O analisador semântico, segundo Aho et al. (2008), utiliza as informações da árvore de sintaxe com o objetivo de verificar a consistência semântica do código-fonte, definindo a linguagem.

Entretanto, segundo os autores, a importância da semântica é a verificação dos tipos em que o compilador checa se a operação é permitida pela especificação da linguagem-fonte, com o objetivo de o compilador relatar alguns erros contidos no programa-fonte e, além disso, possui o foco de compatibilidade entre operações.

Por fim, o conceito de um compilador na síntese tem por finalidade, verificar a estrutura do programa se há combinação entre seus funcionamentos e sintaxes, a fim de verificar também alguns erros do tipo. Dessa forma, a sintática trabalha na

construção do código objeto que é dividido em fases que consiste em gerar o código intermediário, otimização do código e a geração do código objeto enquanto a semântica verifica a compatibilidade. (AHO; SETHI; ULLMAN, 1995; PRICE; TOSCANI, 2001).

BANCO DE DADOS

Na Ciência da Computação, a implementação do banco de dados aumenta a produtividade de um software e ocorre a constante evolução, devido ao fato de ser uma característica essencial para armazenagem de dados (HEUSSER, 2009). O banco de dados é definido como uma coleção de dados gravados da qual pode representar aspectos do mundo real que são usados por usuários para fins específicos (ALVES, 2014; ELMASRI; NAVATHE, 2011; RAMAKRISHNAN; GEHRKE, 2008).

Entretanto, através do banco de dados ocorre o compartilhamento de dados que faz parte da estrutura de um software e devido a sua complexidade em questão de arquivos internos em uma estrutura de software, é necessário minimizar os problemas relacionado com o objetivo de atender diferentes sistemas, para isso usa-se o Sistema de Gerência de Banco de Dados (SGBD) (HEUSSER, 2009).

De acordo com o autor, os programas antigamente continham um conjunto de funcionalidades como as operações de interface, transformações de cálculos e dados, armazenamento de dados, bem como as tarefas de comunicação e vários sistemas e programas.

Para Alves (2014), o SGBD consiste em ferramentas e programas que permitem a criação do banco de dados, podendo realizar manutenção do mesmo e é também um software designado a construção, definição e manipulação de dados.

Dessa forma, o SGBD proporciona muitas vantagens e a manutenção de programa se torna mais simples, devido a separação de funções armazenadas em dados para ser compreendido pelos usuários (HEUSSER, 2009). O autor também afirma que, “a produtividade dos programadores também aumenta já que os programas ficam menores, pois usam funções já construídas” (2009, p.23). Essas funções para o autor ficam armazenadas dentro do banco de dados.

Baseado nas pesquisas, o SGBD é mais fácil de usar do que armazenar dados em arquivos e os gerenciar a partir do código escrito, essa comparação é grande já que quando se trata de código muito complexo, o SGBD entra em ação, minimizando as dificuldades do usuário em relação aos códigos fontes, objetos abstratos e dentre outros. O objetivo do SGBD é também, proteger os usuários dos efeitos de falhas do sistema podendo assegurar que os dados serão restaurados para um estado consistente após o sistema se reiniciar devido a uma falha.

LIBRAS E SUA LINGUAGEM NA TECNOLOGIA DA INFORMAÇÃO

A importância da linguagem é conceder a um indivíduo, a possibilidade de formular os seus conhecimentos, consciência e ideia, sendo um privilégio do sistema linguístico com o intuito de construir subjetividade, se apropriando da cultura da qual está inserida e podendo assim, construir relações sociais.

De acordo com Capovilla (2004), quando não há uma linguagem definitiva na vida do ser humano, ocorre sérias consequências na vida social, causando impactos negativos. Entretanto, para o autor, é muito importante haver a comunicação do ser humano com a sociedade, dessa forma, auxilia com os conhecimentos, podendo possibilitar o compartilhamento linguístico e experiências com a sociedade.

Entretanto, no Brasil, a Língua Brasileira de Sinais (Libras) é reconhecida como segunda língua oficial do Brasil e tem proporcionado melhorias no desenvolvimento educacional dos surdos (CAPOVILLA, 2004). No dia 24 de abril de 2002, foi oficializada a lei 10.436 que se refere a Libras como um meio legal de comunicação entre a comunidade surda (BRASIL, 2002).

Os fundamentos para o desenvolvimento da Libras é através da aprendizagem, tradução e interpretação, dando início a fluência da linguagem de sinais (SILVA; KAUCHAKJE; GESUELI, 2004; LODI; HARRISON; CAMPOS, 2010).

Portanto, as dificuldades quanto à educação dos surdos pode representar a ineficácia da escrita e leitura (SILVA; KAUCHAKJE; GESUELI, 2004; LODI; HARRISON; CAMPOS, 2010), pois existe muita diferença entre a Língua Brasileira de Sinais (Libras) e Língua Portuguesa de acordo com Capovilla (2004) que realizou a pesquisa.

Dessa forma, as soluções computacionais para a aprendizagem em questão educacional da comunidade surda na maioria buscam ensinar Libras e língua portuguesa, mesmo que a comunidade surda siga as regras da Libras, a língua portuguesa não deixa de ser opcional e essencial para obter conhecimentos gramaticais.

Para Silva, Kauchakje e Gesueli sobre as ferramentas computacionais para o uso na comunidade surda:

Outros programas de computadores são utilizados como pratica discursivas no contexto da surdez, em que, com os avanços tecnológicos, as ferramentas para produção escrita (editores de texto, páginas web e de histórias em quadrinho) e para a comunicação a distância (bate-papo, icq e correio eletrônico) fundam novas dimensões para “o uso da linguagem e novas condições de produção de discurso”, o que integra outros elementos em que hoje, é denominado leitura-escrita (2003, p.213).

Há estudos que tratam do acesso da comunidade surda às tecnologias da informação e comunicação, que propõem a adoção do SignWriting como um sistema de escrita para tradução. Para Silva, Kauchakje e Gesueli, o uso do SignWriting para traduzir termos técnicos do português para a língua de sinais facilitará o manuseio de aplicativos e softwares (2003, p. 204).

O SignWriting foi desenvolvido por Valerie Sutton em 1975, nos Estados Unidos e representa graficamente as características visual da Língua de Sinais, até então considerada ágrafa (SILVA; KAUCHAKJE; GESUELI, 2003; LODI; HARRISON; CAMPOS, 2010). O sistema expressa graficamente os sinais em base dos movimentos, nas formas das mãos, nas marcas não manuais e nos pontos de articulações (SILVA; KAUCHAKJE; GESUELI, 2003).

Para Silva, Kauchackje e Gesueli, “mesmo que cresça os membros da comunidade surda, o uso da SignWriting ainda não foi adotado como um sistema de ensino pelas instituições no Brasil” (2003, p.206), inclusive pela dificuldade de sua caligrafia, é importante o uso como forma de melhorar o ensino e a aprendizagem. Além disso, é importante proporcionar vantagem com o uso de software para a aprendizagem educacional, utilizando a tradução da escrita brasileira Libras e dicionário, onde poderia haver informações sobre o significado das palavras, os sinais e as informações sobre a determinada palavra.

CONSIDERAÇÕES FINAIS

Diante do Trabalho, foi feito o estudo sobre algoritmo e suas características fundamentais para ser um bom programador, abordando sobre os quatro principais fundamentos para fazer um bom programa de computador.

Aborda também sobre a importância da programação orientada a objeto e a sua principal estrutura que são classe, objeto, atributo e método, sendo os principais componentes da programação, tornando-se um conjunto de definições que visa determinar o funcionamento de um software. Além disso, é muito importante o uso dos compiladores na programação, foi realizado a pesquisa abordando a possibilidade de otimizar o código fonte para objeto, dando início a um sistema básico de desenvolvimento e a ligação com o banco de dados de como poderia ser usado para armazenar as informações, minimizando as dificuldades dos usuários de programação em desenvolver projetos. Foi pesquisado também, sobre o uso do Banco de Dados e a sua importância dentro da programação, podendo garantir a segurança de informações e, além disso, o sistema de gerencia de banco de dados proporciona vantagem utilizando o armazenamento de dados.

E por fim, especificamente a Libras em como poderia haver uma solução computacional para aprendizagem em questão de educação proposto por escolas ou instituições de forma que a comunidade surda aprenda a escrita brasileira, através do uso de dicionário e tradutor, enriquecendo suas gramáticas e as possibilidades de aprendizagem em relação a língua de sinais.

Dentre os demais fatores, foi pesquisado a possibilidade de haver um projeto de programação que visa solucionar o problema dentro da educação, podendo adotar o método de compilação juntamente com o banco de dados que armazenam informações, para assim, realizar a tradução da língua portuguesa para Libras e a utilização de dicionário. O estudo do projeto é baseado em pesquisas que ao longo do tempo tem um histórico referente as dificuldades da comunidade surda e as principais características da programação de computadores que pode auxiliar usando a lógica de programação.

REFERÊNCIAS

- AHO, A. V. et al., **Compiladores: princípios, técnicas e ferramentas**. 2. ed., São Paulo: Pearson, 2008.
- AHO, A. V.; SETHI, R; ULLMAN, J. D. **Compiladores: princípios, técnicas e ferramentas**. Tradução Daniel de Ariosto Pinto. Rio de Janeiro: LTC, 1995.
- ALVES, W. P., **Banco de dados**. 1a. ed., São Paulo: Erica, 2014.
- BRASIL. Lei no 10.436, de 24 de abril de 2002, que dispõe sobre a Língua Brasileira de Sinais – Libras. Brasília: Casa Civil, 2002. Disponível em: <http://www.planalto.gov.br/ccivil_03/_ato2004-2006/2005/decreto/d5626.htm>. Acesso em: 10 jun. 2018.
- CAPOVILLA, F. C., **Neuropsicologia e Aprendizagem: uma abordagem multidisciplinar**. 2a. Ed. São Paulo: Memnon, 2004.
- CORMEN, T. H., et al., **Algoritmos: teoria e prática**. Rio de Janeiro: Campus, **2002**.
- ELMASRI; NAVATHE, **Sistemas de banco de dados**, 6a.ed., São Paulo: Pearson, 2011.
- HEUSSER, C. A., **Projeto de banco de dados**, 6a. Ed., Porto Alegre: Bookman, 2009.
- LAFORE, R. **Aprenda em 24 horas Estrutura de Dados e Algoritmos**. Rio de Janeiro: Campus, 1999.
- LODI, A. C. B; HARRISON, K. M. P; CAMPOS, S. R. L. **Leitura e escrita: no contexto da diversidade**, Porto Alegre: Mediação, 2010.
- MANZANO. J. A. OLIVEIRA. J. F. de. **Algoritmos: lógica para desenvolvimento de programação de computadores**. Ed.28. – São Paulo: Érica 2016.
- MENDES, D. R., **Programação Java com Ênfase em Orientação a Objetos**, 1a. Ed., São Paulo: Novatec, 2009.
- PRICE, A. M. A.; TOSCANI, S. S. **Implementação de linguagens de programação: compiladores**, 2.ed. Rio Grande do Sul: Sagra Luzzatto, 2001.
- RAMAKRISHNAN, R., GEHRKE, J. **Sistemas de Gerenciamento de Bancos de Dados**. AMGH, 01/2008
- SALIBA, W.L.C., **Técnicas de programação: uma abordagem estruturada**. São Paulo: Makron-Books, 1992.

SANTOS, R. **Introdução a programação orientada a objetos usando Java**, Rio de janeiro: Campus, 2013.

SILVA, I. R; KAUCHAKJE, S; GESUELI, Z. M. **Cidadania, Surdez e Linguagem: Desafios e realidades**, São Paulo: Plexus, 2003.

SZWARCFITER, J.L; MARKENZON, L. **Estruturas de Dados e Seus Algoritmos**. 2.ed. Rio de Janeiro: LTC, 1994.