CENTRO UNIVERSITÁRIO CAMPOS DE ANDRADE

CIÊNCIA DA COMPUTAÇÃO

PROJETO INTEGRADOR

2020/1

SUMÁRIO

1. TEMA

O tema do projeto integrador – PI, baseia-se em uma situação-problema criada pelos professores orientadores do PI e estão descritas no Anexo I.

2. OBJETIVO

O propósito do PI é propiciar condições para que o acadêmico possa desenvolver de forma prática, os conhecimentos teóricos abordados em sala de aula, através de uma ação multidisciplinar entre as disciplinas do semestre corrente. Além disso, o PI foca em um estudo dirigido, no qual o acadêmico ao se deparar com uma situação problema, possa apresentar possíveis soluções.

3. DISCIPLINAS E SUAS APLICAÇÕES

Cabe aos professores(as), orientar e separar subtemas para as equipes participantes do projeto, além de auxiliar os acadêmicos no desenvolvimento.

Em anexo encontra-se a descrição do problema proposto para cada período, além das especificações dos professores(as) responsáveis pelo projeto integrador.

4. METODOLOGIA

Os(as) acadêmicos(as) serão separados(as) em grupos e trabalharão pelo menos 1 dia/aula com o projeto integrador em sala de aula e/ou laboratório em todas as disciplinas. Cabe ao grupo adotar a metodologia de pesquisa bibliográfica, indicadas pelos professores orientadores, para resolver o problema proposto em seu tema do projeto integrador.

Além da resolução técnica do problema do projeto, cada grupo apresentará seus projetos desenvolvidos para os demais alunos dos cursos no dia 24 de junho. A ordem das apresentações como também os grupos que irão apresentar serão deliberadas pelos professores responsáveis pelo projeto, podendo: selecionar grupos específicos para apresentar os melhores projetos ou utilizar a data para todos os grupos apresentarem seus projetos desenvolvidos.

5. AVALIAÇÃO

Fica aberto ao professor(a) o método avaliativo (apresentação ou entrega de trabalho impresso), bem como aumentar o valor do projeto em sua disciplina. A nota valerá de acordo com a classificação dos períodos na tabela abaixo. O cálculo é formado pela média aritmética das disciplinas que o(a) acadêmico(a) está matriculado.

CIÊNCIA DA COMPUTAÇÃO

Período	Projeto Integrador
1º	2.0 pontos
2º	2.0 pontos
3º	2.0 pontos
4º	2.0 pontos
5º	1.0 ponto
6º	1.0 ponto
7º	1.0 ponto
8º	1.0 ponto

Cada período seguirá as indicações e normativas contidas em anexo, sem fugir dos requisitos necessários para a conclusão do projeto. É deferido uma atribuição de nota parcial para a disciplina, se assim deseja o professor(a) orientador(a), ou seja, além da nota do projeto integrador, o professor poderá utilizar o trabalho para compor uma nota parcial da sua disciplina.

6. ENTREGA DAS NOTAS

Dos professores(as) orientadores(as):

- **Data: 23/06/2020**

Cada professor(as) preencherá a nota de cada acadêmico(a) numa planilha e enviará até a data acima para a coordenação.

Da coordenação:

- **Data: 30/06/2020**

Fica cargo da coordenação de curso receber todas as notas e realizar o cálculo da média aritmética baseado na quantidade de disciplinas que o(a) aluno(a) está **matriculado(a)** e enviar para todos os(as) professores(as) e representantes de turma até a data estipulada.

7. REGRAS E CONSIDERAÇÕES

I. Todos os(as) acadêmicos(as) devem participar do projeto, a não participação significa na perda da nota.

II. O material final entregue ou apresentado, devem ser repassados para a coordenação juntamente com as notas, por cada professor(a), numa planilha até a data estabelecida.

III. A atividade não entregue ou apresentada dentro do prazo estipulado pelo professor(a), como também plágios, serão desconsiderados, atribuindo **nota zero ao grupo**.

IV. Casos excepcionais serão analisados pelo colegiado de curso desde que o(a) aluno(a) apresente sua justificativa por escrito via e-mail para a coordenação do curso **até o dia 07/07/2020**.

Profº. Me. Edgar Cabral
Coordenador

ANEXO I

2º e 3º Período

O trabalho prático consiste em desenvolver algoritmos para solucionar problemas de Matemática e Física. Um programa deverá ser desenvolvido para cada problema proposto. Esses programas deverão ser criados utilizando conceitos vistos nas aulas de Linguagem de Programação I, Física Aplicada e Cálculo 1.

A seguir, encontram-se a descrição do problema, a forma de submissão do trabalho e os critérios de avaliação.

- Equipes de 2 ou 3 participantes;
- Os algoritmos devem ser desenvolvidos e executados na Linguagem C.
- O desenvolvimento do trabalho deverá ser escrito em 4 relatórios parciais de 1 a 2 páginas e um relatório completo de 5 a 7 páginas, contendo todo o desenvolvimento do trabalho e cálculos devem ser descritos em um relatório final de 5 a 7 páginas, no formato de artigo da Iniciação Científica.

Regras para entrega do trabalho

Data da Entrega:

- ✓ **16/03/2020 – Entrega do 1º relatório parcial por e-mail até 23h59.**
- ✓ **06/04/2020 – Entrega do 2º relatório parcial por e-mail até 23h59.**
- ✓ **11/05/2020 – Entrega do 3º relatório parcial por e-mail até 23h59.**
- ✓ **22/05/2020 – Apresentação do trabalho na aula de Física.**
- ✓ **01/06/2020 – Entrega do 4º relatório parcial por e-mail até 23h59.**
- ✓ **04/06/2020 – Apresentação do trabalho na aula de Cálculo 1.**
- ✓ **02/06/2020 – Apresentação do trabalho na aula de Ling. de Prog. I.**
- ✓ **15/06/2020 – Entrega do relatório final por e-mail até 23h59.**
- ✓ **24/06/2020 – Apresentação do melhor projeto integrador (evento com todos os alunos dos cursos de informática da Uniandrade).**

O código-fonte dos programas e o relatório deverão ser e entregues por e-mail para os endereços prof.aboim@gmail.com, allan_perna@hotmail.com e camilafranciscamelo@gmail.com. O nome completo dos participantes da equipe deve ser informado no corpo do e-mail.

Especificação do Trabalho Prático

Problema 1

A empresa aérea *PhysicsAviation* transporta passageiros somente entre algumas cidades do Brasil. Essa empresa possui dois tipos de aviões:

- *VectorComponent007* que só executa trajetórias retilíneas paralelas à linha do equador e ao meridiano de Greenwich, ou seja, trajetórias horizontais e verticais, nas direções norte/sul e leste/oeste.
- *ResultantVector011* que executa qualquer trajetória retilínea entre as cidades.

Por exemplo, um dos aviões voa para o norte, de Brasília até Belém, a 1630 km de distância, levando 2h e 10 min nesse percurso. De lá, segue para oeste, chegando a Manaus, distante 1290 km de Belém, após 1h e 50 min de voo. Qual é o vetor deslocamento total do avião? Em outras palavras, quais são as componentes vertical e horizontal do deslocamento total do avião? Qual o valor do módulo deste vetor?

Você é funcionário da *PhysicsAviation* e precisa criar uma programa que forneça algumas informações para o piloto. O piloto irá informar ao programa o tipo de avião com o qual vai viajar e o número do plano de voo que contém as informações referentes as cidades pelas quais tem que passar, origem e destino. Os planos de voo, referente a cada tipo de avião, estão apresentados nas tabelas a seguir:

Tipo de avião	Número do Plano de voo	Origem	Destino intermediário	Destino Final
VectorComponent007	PA232	Curitiba	Rio de Janeiro	Manaus
	PA457	Porto Alegre	Florianópolis	Rio de Janeiro
	PA949	Brasília	São Paulo	Curitiba
	PA1080	Belém do Pará	Manaus	Goiânia

Tipo de avião	Número do Plano de voo	Origem	Destino Final
ResultantVector011	PA360	São Paulo	Rio de Janeiro
	PA888	Brasília	Manaus

Utilize o *Google Maps* para estimar a distância entre as cidade, mas lembre-se que você deve conhecer principalmente a distância horizontal e vertical entre as cidades. Por exemplo, se o avião sai de Curitiba e vai para o Rio de Janeiro, o piloto da *PhysicsAviation* executará a seguinte trajetória aproximada com o avião Vector Component007:

1º) 280 km ao norte (linha reta paralela ao meridiano de Greenwich).

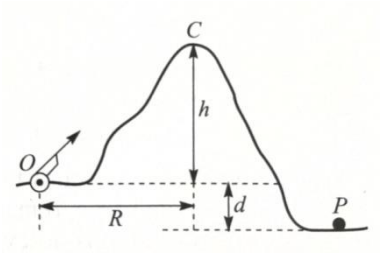
2º) 750 km ao leste (linha reta paralela ao equador).

Se o piloto estiver viajando com o avião *VectorComponent007*, o programa deve informar a componente horizontal (paralela ao equador) e vertical (paralela ao meridiano de Greenwich) do vetor deslocamento entre as cidades. Ou seja, entre a origem e o destino intermediário, entre o destino intermediário e o destino final, e entre a origem e o destino final. Além disso, o programa deve informar o módulo do vetor deslocamento entre a origem e o destino final.

Se o piloto estiver viajando com o avião *ResultantVector011* programa deve informar o módulo do vetor deslocamento entre a origem e o destino. Além disso, o programa deve informar qual o menor ângulo formado entre a linha reta que liga as cidade e uma linha paralela à linha do Equador.

Problema 2

Um canhão lança um projétil por cima de uma montanha de altura h, de forma a passar quase tangenciando o cume C no ponto mais alto de sua trajetória. A distância horizontal entre o canhão e o cume é R. Atrás da montanha há uma depressão de profundidade d (veja a figura abaixo). Determine a distância horizontal entre o ponto de lançamento O e o ponto P onde o projétil atinge o solo em função de R, d e h.



Uma empresa de armamento militar desenvolveu uma bomba que não pode ser rastreada se estiver próxima ao solo. O objetivo deste armamento é destruir grupos inimigos que se encontram próximos de montanhas. Assim, a base de lançamento da bomba pode ser construída do outro lado da montanha e executar um ataque surpresa, desde que a bomba não se afaste muito das rochas, por isso a bomba deve passar quase tangenciando o cume.

Os soldados do grupo de ataque tem contato com engenheiros cartográficos que conhecem as medidas da topografia da região. O geógrafo vai informar os valores de R, d e h e os soldados, com essas três informações, devem saber o local no qual a base de lançamento deve ser instalada. Crie um programa que calcule a distância horizontal entre o ponto de lançamento e o ponto onde o grupo inimigo se encontra.

Problema 3

A empresa de tecnologia *StudentsPro* trabalha desenvolvendo aplicativos e *softwares* que ajudam os alunos de curso superior. A empresa é dividida em 4 grandes áreas do conhecimento: Ciências Humanas e suas Tecnologias, Ciências da Natureza e suas Tecnologias, Linguagens, Códigos e suas Tecnologias e Matemática e suas Tecnologias. A última subdivisão, Matemática e suas Tecnologias receberam a encomenda de um *software*

que vai acompanhar a disciplina e será vendido acompanhando a disciplina de Cálculo. Como o contrato envolve valores exponenciais de reais é necessário entregar uma prévia deste *software* para o comprador.

A empresa de tecnologia sabe da importância do conteúdo de derivadas e resolveu utilizar isso a seu favor. Imagine que você é o desenvolvedor da *StudentsPro*, e recebeu do seu gerente um *software* com as seguintes instruções:

i. Crie um *software* que receba uma função de uma variável $f(x)$ e entregue a derivada desta função $f'(x)$.
ii. Neste mesmo *software* devem ser plotados juntos ou não a função $f(x)$ e a derivada desta mesma função pertencente a um domínio qualquer.
iii. Junto da plotagem da função deve utilizar algum recurso que permita diferenciar qual é a função é qual é a derivada da função.
iv. O *software* deve ser capaz de resolver derivadas que utilizem pelo menos dois casos especiais de derivadas.

Requisitos

a) Para cada problema proposto, identifique os conceitos (físicos e/ou matemáticos) envolvidos;
b) Para cada problema proposto, determine quais são as fórmulas, equações, ou funções necessárias para solucionar o problema;
c) Crie um algoritmo (programa) para solucionar cada um dos problemas propostos;
d) Defina a entrada, processamento e saída de cada algoritmo;
e) Crie a representação gráfica (fluxograma) de cada algoritmo;
f) Os programas devem exibir informações e mensagens para que o usuário realize a correta entrada de dados. Os resultados devem ser exibidos na saída de forma clara e organizada.
g) O relatório final deve ser organizado conforme a seguinte estrutura:
    - Resumo
    - Introdução
    - Desenvolvimento
        - Problema <X>
            - Descrição
            - Conceitos e Cálculos
            - Algoritmo
    - Resultados
        - Problema <X>
    - Conclusão
    - Referências

A seção Desenvolvimento deve conter uma subseção para cada problema proposto. Para cada problema devem ser apresentados uma descrição resumida do problema, os conceitos e cálculos envolvidos e o algoritmo desenvolvido.

A seção Resultados deve conter uma subseção para cada problema proposto. Para cada problema deve ser apresentado ao menos um exemplo de execução do programa correspondente. Mostre a entrada e saída produzida pelo programa.

Critérios de avaliação

Os trabalhos serão avaliados de acordo com os seguintes critérios objetivos:

Critério	**Nota***
Conteúdo dos relatórios e apresentações.	De 0 a 0.6
Atendimento aos requisitos	De 0 a 0.2
Corretude dos requisitos	De 0 a 0.2
Trabalho entregue fora do prazo.	Metade da nota por dia de atraso

*** O total de pontos do Projeto Integrador no 2º bimestre será 2.0 pontos para o 1º e 2º período.**

Alguns alunos serão convocados durante a defesa para uma entrevista para explicar detalhes do trabalho. A nota individual do aluno dependerá desta entrevista.

Os trabalhos serão avaliados segundo critérios subjetivos definidos pelos professores, variando a nota dentro da faixa de notas definida pelo critério anterior. Alguns dos critérios subjetivos avaliados serão:

- Legibilidade (nome dos identificadores, formatação do código, uso de comentários quando necessário);
- Modularização (criação de funções bem definidas e independentes);
- Eficiência (desempenho e uso de recursos);
- Uso de estruturas de dados adequadas.

Referências

1. HALLIDAY, D.; RESNICK, R.; WALKER, J. Fundamentos de Física: mecânica. 10.ed. Rio de Janeiro, RJ: LTC, 2008.
2. STEWART, James. Cálculo. v. 1. 5ª edição. São Paulo: Cengage Learning.
3. SWOKOWSKI, E. W. Cálculo com geometria analítica. 2 ed. São Paulo: Makron Books, 1995.

4. Damas, Luís. **Linguagem C**; tradução João Araújo Ribeiro, Orlando Bernardo Filho. - 10.ed. - [Reimpr.]. - Rio de Janeiro : LTC, 2016.
5. Manzano, José Augusto N. G. **Linguagem C** : acompanhada de uma xícara de café / José Augusto N. G. Manzano. - São Paulo : Érica, 2015. 480 p.

4º e 5º Período

O projeto integrador consiste na criação de um sistema computacional para as disciplinas de Banco de Dados 2, Probabilidade e Estatística e Pesquisa e Ordenação.

A seguir, encontram-se a descrição do problema, a forma de submissão do trabalho e os critérios de avaliação.

- Equipes de 3 a 5 participantes;
- A entrega do trabalho deve conter o código do sistema e um Relatório final.

Regras para entrega do trabalho

Datas de Entregas/Defesas:

- ✓ **30/03/2020 – Avaliação parcial do trabalho na aula de Banco de Dados.**
- ✓ **31/03/2020 – Avaliação parcial do trabalho na aula de Pesquisa e Ordenação.**
- ✓ **15/06/2020 – Entrega final e Defesa individual do trabalho na aula Banco de Dados para todos os professores.**
- ✓ **24/06/2020 – Apresentação dos melhores projetos integradores de cada período no auditório.**

O trabalho deve ser e entregue por e-mail (em 15/06) para os endereços: prof.aboim@gmail.com, jj.gdias@gmail.com e camile.bordini@gmail.com. O nome completo dos participantes da equipe devem ser informados no corpo do e-mail.

Especificação do Trabalho Prático

Descrição do problema

A análise de dados históricos assim como a pesquisa eleitoral são grandes oportunidades para que candidatos prevejam suas possibilidades de serem eleitos para determinado cargo em uma certa eleição.

Com a proximidade das eleições municipais de 2020, você e sua equipe foram contratados para desenvolver um sistema que auxilie os candidatos a analisarem suas situações.

O trabalho é composto por duas frentes: o levantamento de dados históricos de eleições passadas e na simulação de uma pesquisa eleitoral para as eleições municipais de 2020.

1. HISTÓRICO

1.1. Levantamento

Realizar o levantamento de dados eleitorais dos anos de 2016, 2012 e 2008 de candidatos a cargo de prefeito e vereadores, para o município de Curitiba-PR no site do TSE / TRE.

Os dados a serem levantados são:

- Ano da eleição (2016, 2012 ou 2008);
- Número, nome e partido/coligação do candidato;
- Número de votos recebidos por: faixa etária, sexo, grau de escolaridade e zona eleitoral.

1.2. Ordenação

A partir desses dados, o sistema deve ser capaz de mostrar em uma tela ordenada as seguintes informações (para uma das eleições escolhida pelo usuário):

- Por prefeito (número de votos recebidos)
- Por vereador (número de votos recebidos)
- Por zona eleitoral (número de votos recebidos, para prefeito e vereador)

** OBS: Sempre mostrar junto ao nome do candidato e o número de votos recebidos mais alguma informação relevante para a análise, como o seu partido político por exemplo.*

Para a ordenação dos dados históricos: o sistema deve, internamente, realizar a ordenação do que foi solicitado pelo usuário utlizando TODOS os algoritmos de ordenação estudados em sala de aula até o 1º Bimestre:

- **BubbleSort**
- **Selection Sort**
- **Insertion Sort**
- **QuickSort**
- **MergeSort**

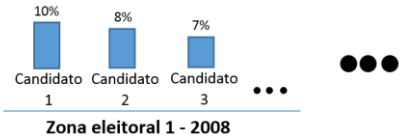
Abaixo da tela deve mostrar o tempo de execução que cada algoritmo levou. Para isso você deve colocar um contador de tempo ao início e ao fim de CADA método de ordenação, para que uma comparação final possa ser feita pelo usuário entre os métodos.

Como todos os algoritmos produzem a mesma “resposta”, obviamente não é necessário exibir em tela as respostas de todos, você pode escolher apenas um dos algoritmos para exibir os nomes dos candidados ordenados por número de votos recebidos.

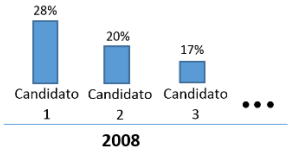
1.3. Estatística

A partir destes dados, a equipe deve apresentar um Relatório com os seguintes gráficos estatísticos:

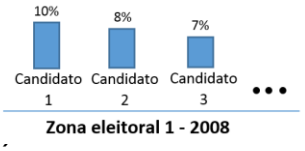
a) Percentual de votos em ordem decrescente de cada candidato à prefeitura no primeiro turno e no segundo turno (se houve), em cada zona eleitoral, por ano (2016, 2012, e 2008). Ilustração:



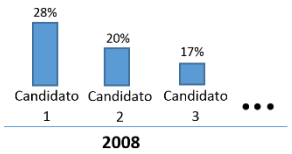
b) Percentual de votos em ordem decrescente de cada candidato a prefeitura em Curitiba no primeiro turno e segundo turno (se houve), por ano (2016, 2012, e 2008), para Curitiba. Ilustração:



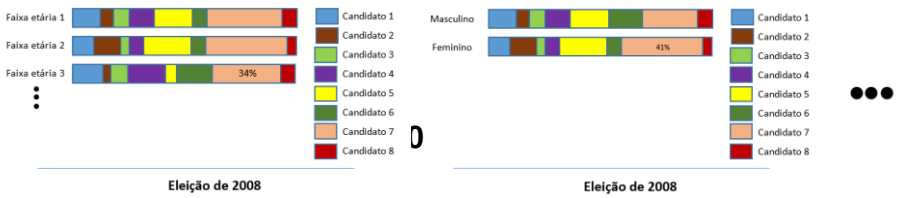
c) Percentual de votos em ordem decrescente para os **10** vereadores mais votados a câmara de vereadores de Curitiba, em cada zona eleitoral, por ano (2016, 2012, e 2008). Ilustração:



em decrescente para os **20** vereadores mais votados a câmara de vereadores de Curitiba, por ano (2016, 2012, e 2008). Ilustração:



e) Percentual de votos de cada candidato à prefeitura em Curitiba no primeiro turno e segundo turno (se houve), por ano (2016, 2012, e 2008) e por faixa etária, sexo e grau de escolaridade para Curitiba. Ilustração considerando que sejam oito candidatos:



2.1. Pesquisa

Pesquisar os pré-candidatos a prefeito e vereador e seus respectivos partidos em Curitiba para as eleições de 2020. Caso não haja nomes suficientes de vereadores para determinado

partido, a equipe pode criar algumas opções com base em eleições passadas, desde que não passe de 10 candidatos por partido.

Todos esses dados devem ser cadastrados no banco de dados.

2.2. Simulação

Realizar a “simulação” de uma pesquisa eleitoral realizada em Curitiba.

Os dados a serem levantados pela pesquisa são:

Nome, Idade, sexo, escolaridade, bairro(*), religião, cor, renda, prefeito(**), vereador(**), partido(**), data(***).

(*) A partir do bairro, o sistema deve buscar qual a zona eleitoral daquele eleitor (pesquisar a respeito).

(**) O campo de “partido” só é habilitado para preenchimento se pelo menos um dos campos “prefeito” ou “vereador” não for preenchido.

(***) Devem ter 3 datas de pesquisas realizadas.

OBS: a quantidade de “entrevistados” pesquisados devem ser de 20 pessoas por zona eleitoral.

2.3. Ordenação

A partir desses dados, o sistema deve ser capaz de mostrar em uma tela ordenada as seguintes informações:

- Por prefeito (número de votos recebidos)
- Por vereador (número de votos recebidos)
- Por zona eleitoral (número de votos recebidos, para prefeito e vereador)

** OBS: Sempre mostrar junto ao nome do candidato e o número de votos recebidos mais alguma informação relevante para a análise, como o seu partido político por exemplo.*

Para a ordenação dos dados da pesquisa: O sistema deve, internamente, realizar a ordenação do que foi solicitado pelo usuário utlizando TODOS os algoritmos de ordenação estudados em sala de aula na disciplina, ou seja, todos os anteriores (**BubbleSort, Selection Sort, Insertion Sort, QuickSort, MergeSort**) e mais:

- **HeapSort**
- **Countingsort**

- **Radixsort**
- **Bucketsort**

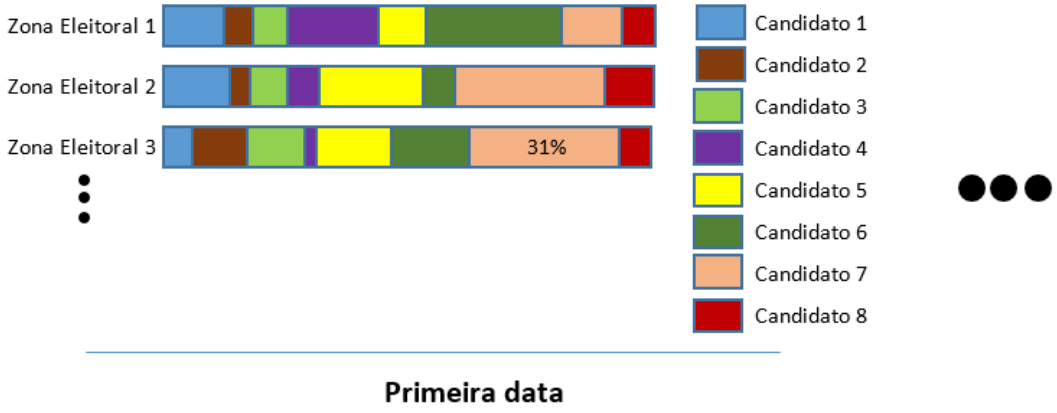
Abaixo da tela deve mostrar o tempo de execução que cada algoritmo levou. Para isso você deve colocar um contador de tempo ao início e ao fim de CADA método de ordenação, para que uma comparação final possa ser feita pelo usuário entre os métodos.

Como todos os algoritmos produzem a mesma "resposta", obviamente não é necessário exibir em tela as respostas de todos, você pode escolher apenas um dos algoritmos para exibir os nomes dos candidados ordenados por número de votos recebidos.

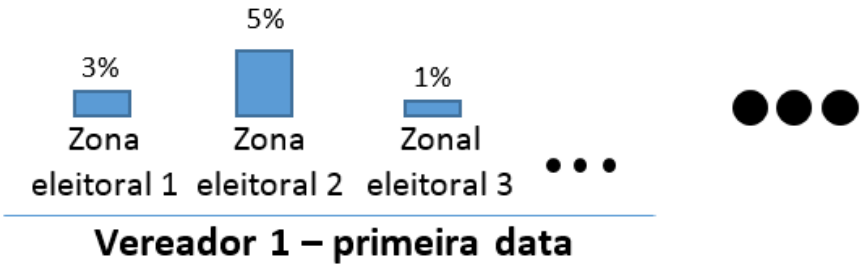
2.4. Estatística

Além disso, a equipe deve apresentar alguns gráfico estatísticos sobre esses dados:

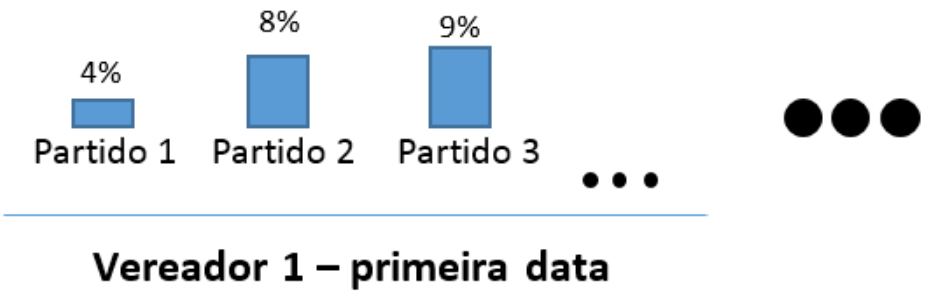
a) Percentual de cada candidato a prefeito, por zona eleitoral, por data de pesquisa. Ilustração considerando que sejam oito candidatos:



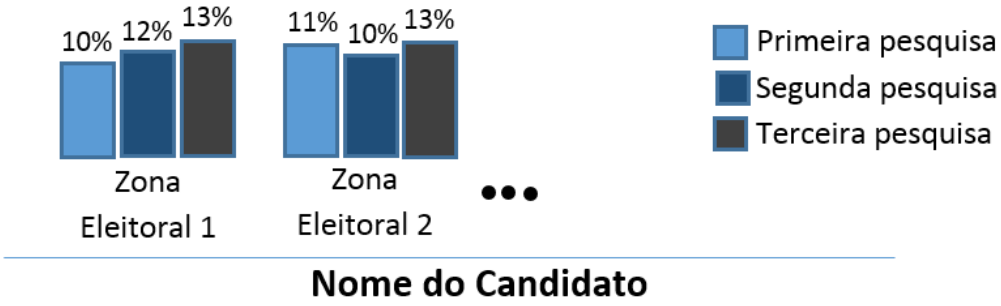
b) Percentual de cada candidato a vereador, por zona eleitoral, por data de pesquisa. Ilustração:



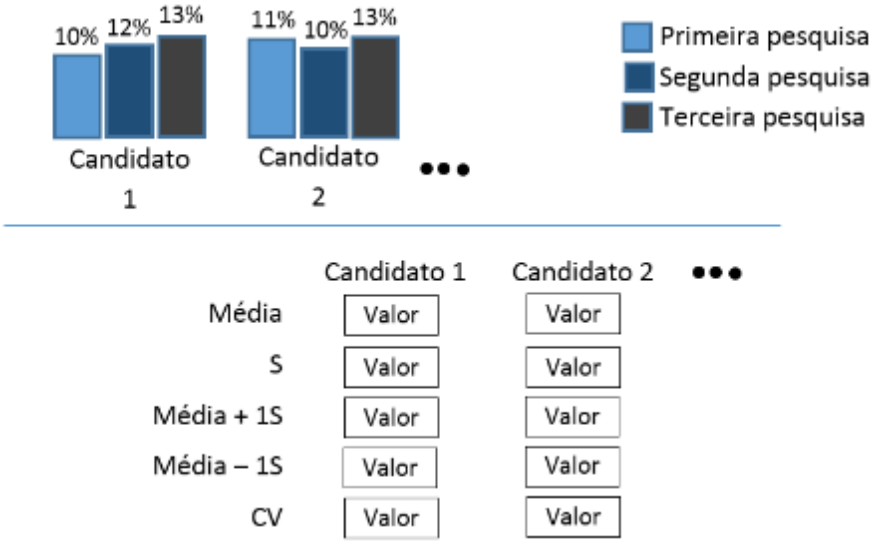
c) Percentual de cada partido (somando os votos de todos os seus candidatos a vereadores), por zona eleitoral, por data de pesquisa. Ilustração:



d) Percentual de cada candidato a prefeito, por zona eleitoral, por data de pesquisa. Ilustração:



e) Percentual de cada candidato a prefeito, para Curitiba, por data de pesquisa. Neste gráfico, mostre para cada candidato o percentual médio, médio +1S, médio –1S, coeficiente de variação e regressão linear simples considerando o gráfico com a data que a equipe atribuiu para cada pesquisa e a data real da eleição do primeiro turno de 2020 (data para estimar o percentual).



(*) Observação: alguns parâmetros poderão ser ajustados pelos professores durante o decorrer das disciplinas. Caso isso ocorra, será avisado à turma, sem prejuízo para a interpretação geral do projeto.

Entrega

Código do sistema

Banco de Dados: Deverá ser desenvolvido o MER, com chaves (PK / FK), índices, cardinalidades entre as tabelas; rotinas de criação, importação, exportação, alteração, exclusão e consultas diversas com utilização de queries, views, procedures e trigges e; geração de LOG FILE de todas as movimentações no sistema para segurança, auditoria e integridade das informações.

Pesquisa e Ordenação: deverá ser implemetado na linguagem de programação escolhida todos os algoritmos de ordenação vistos em sala de aula:

- Até a 1º avaliação parcial: BubbleSort, Selection Sort, Insertion Sort, QuickSort e MergeSort.
- Até a entrega/defesa final: Todos os métodos anterioes e mais: HeapSort, Countingsort, Radixsort e Bucketsort.

Mostrando o tempo de execução de cada algoritmo mais abaixo da tela dos nomes ordenados.

Probabilidade e Estatística: arquivo em PowerPoint com o print das linhas onde foram feitos os cálculos do **2.4 e**. Informe no slide o que significam as variáveis envolvidas no cálculo. Por exemplo:

Relatório

O relatório deve ser estruturado conforme regras institucionais, com no mínimo, as seguintes sessões:

- Resumo
- Introdução
- Descrição do problema
- Desenvolvimento
- Resultados
- Conclusão/ Propostas futuras
- Referências

Critérios de avaliação

*** 1º Bimestre: para cada disciplina o valor do trabalho será de:**

Banco de Dados: 5.0 pontos.

Pesquisa e Ordenação: 3.0 pontos

Probabilidade e Estatística: 1.0 ponto

*** 2º Bimestre: a pontuação do Projeto Integrador será de 2.0 pontos para o 4º período e 5º período. Para cada disciplina, a nota no bimestre terá adicionalmente:**

Banco de Dados: 3.0 pontos.

Pesquisa e Ordenação: 2.0 pontos

Probabilidade e Estatística: 1.0 ponto

Alguns alunos serão convocados durante a defesa para uma entrevista, na qual deverão explicar o trabalho desenvolvido. A nota individual do aluno dependerá desta entrevista, segundo critérios subjetivos considerados pelos professores.

Referências

WIRTH, Niklaus. **Algoritmos e estruturas de dados**. Rio de Janeiro: LTC, 1999.

PREISS, Bruno R. **Estrutura de Dados e algoritmos: Padrões de projetos orientados a objetos com java**. Rio de Janeiro: CAMPUS, 2001.

CRESPO, A. A. **Estatística Fácil.** São Paulo: Editora Saraiva, 2000.

AMADEU, Claudia Vicci. **Banco de Dados.** São Paulo: Pearson Education do Brasil, 2014.

ELMASRI, Ramez; NAVATHE, Shamkant B. **Sistemas de Banco de Dados**. 6. ed. São Paulo: Pearson, 2011.

COUGO, Paulo. **Modelagem conceitual e projeto de banco de dados.** Rio de Janeiro: Campus: 1997.

DATE, C.J. **Introdução a sistemas de banco de dados.** Rio de Janeiro: Campus, 2000.

MACHADO, Felipe; ABREU, Maurício. **Projeto de banco de dados**. São Paulo: Érica: 2001.

SILBERSCHATZ, Abraham et al. ***Sistema de banco de dados.*** *São Paulo: Makron Books, 1999.*

6º e 7º Período

O projeto integrador consiste na execução e implementação (você escolhe a linguagem) de um algoritmo para o problema de encontrar uma árvore geradora mínima (*minimum spanning tree)* sobre determinados grafos e de extração de vértices em grafos de acordo com o grau.

Ao final destas execuções, teremos grafos que servirão de base para a criação de um autômato finito determinístico (AFD) para reconhecer cadeias de caracteres específicas. Este autômato deverá ser desenhado no JFLAP8 e também ser implementado em alguma linguagem de programação. As equipes formadas poderão ter até 4 participantes.

A seguir, encontram-se a descrição do problema, a forma de submissão do trabalho e os critérios de avaliação por disciplina.

OBS: no dia 24/06/2020 (quarta-feira) serão apresentados os melhores projetos de cada período no auditório.

1. Especificação do Trabalho Prático

Descrição do problema

Estamos em 2570, vivemos uma era altamente tecnológica: carros voadores, dobras espaciais, exoesqueleto, vacinas que permitem o ser humano viver 300 anos etc. Grande parte deste avanço deve ser agradecido aos alunos da maior instituição de ensino mundial: a Uniandrade (que tornou Harvard e MIT suas filiais secundárias). Mas como nada é perfeito, a ganância tomou conta de alguns alunos (faltaram as aulas de “ética espacial”) fazendo com que os mesmos explorassem a Zona X7 com a nave Interestelar.

Esta área abriga seres conhecidos como Zergs, cuja única função é destruir toda forma de vida em outros planetas. Os Zergs conseguiram chegar ao nosso planeta escondido na Interestelar e infectar inicialmente 47% da população mundial.

Depois de 48 horas que um ser humano é infectado, o mesmo se torna um zumbi cego com um único objetivo: se alimentar de qualquer ser vivo que emita calor.

Agora temos duas notícias, uma boa e uma ruim. A notícia boa é que existe a vacina para a cura, porém, se for injetada em alguém não contaminado, a pessoa morre. A única coisa que precisamos fazer é coletar uma amostra de sangue das pessoas e verificar se uma das cadeias de proteína existe.

Cadeia 1: $\{b(ab)^n b \mid n \geq 0\}$

Cadeia 2: $\{ba^n ba \mid n \geq 0\}$

A notícia ruim é que nosso autômato de reconhecimento foi destruído pelos ataques Zergs e desta forma teremos que recriá-lo. Alguns passos deverão ser executados até chegarmos na implementação do autômato.

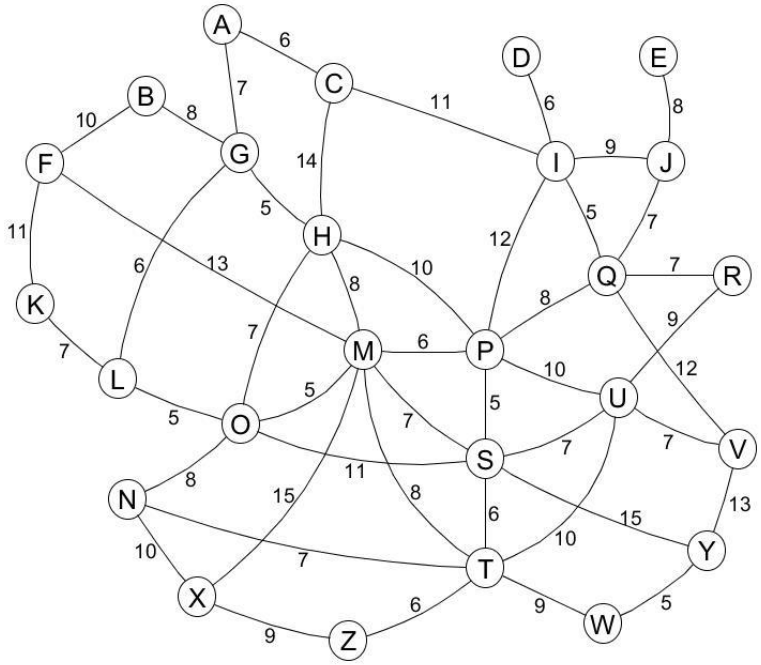
Vocês serão divididos em vários grupos (até 4 pessoas). Alguns grupos irão ser responsáveis por identificar a cadeia 1, enquanto outros grupos ficarão com a cadeia 2.

Atividades

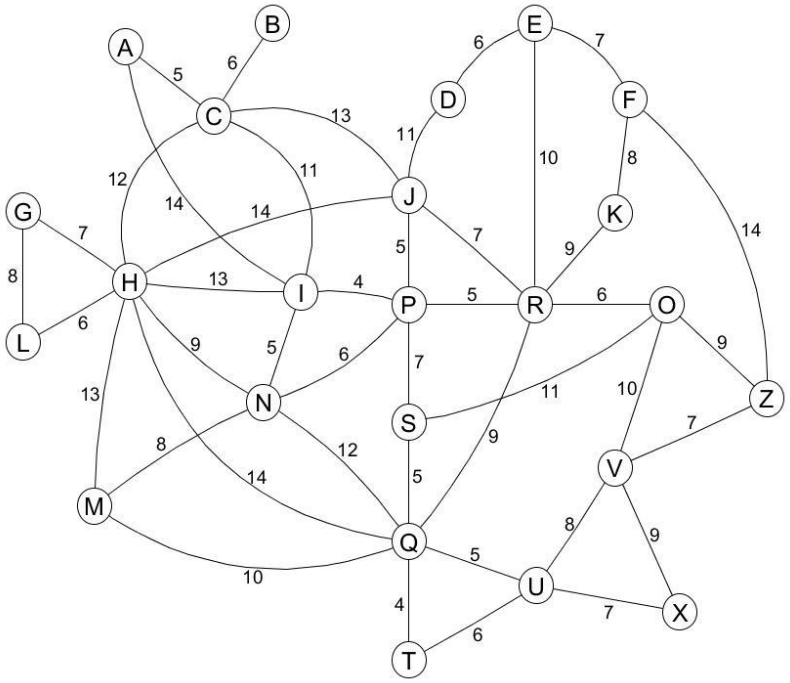
1. Recriar os dois grafos de proteínas no software Gephi, de acordo com a cadeia escolhida pela equipe.

Cadeia 1: grafos 1 e 2

Grafo 1: 26 vértices, 46 arestas

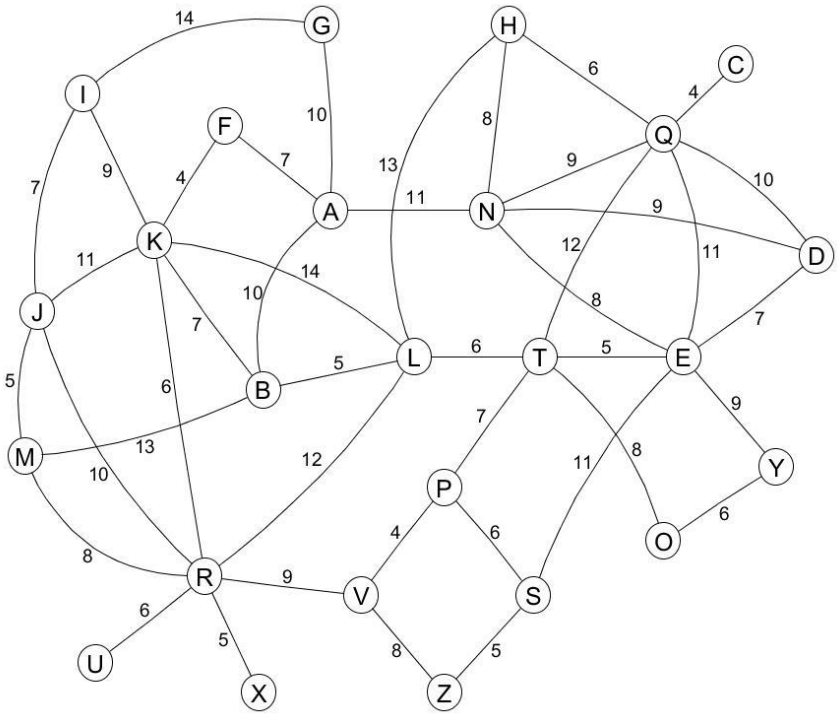


Grafo 2: 24 vértices, 44 arestas

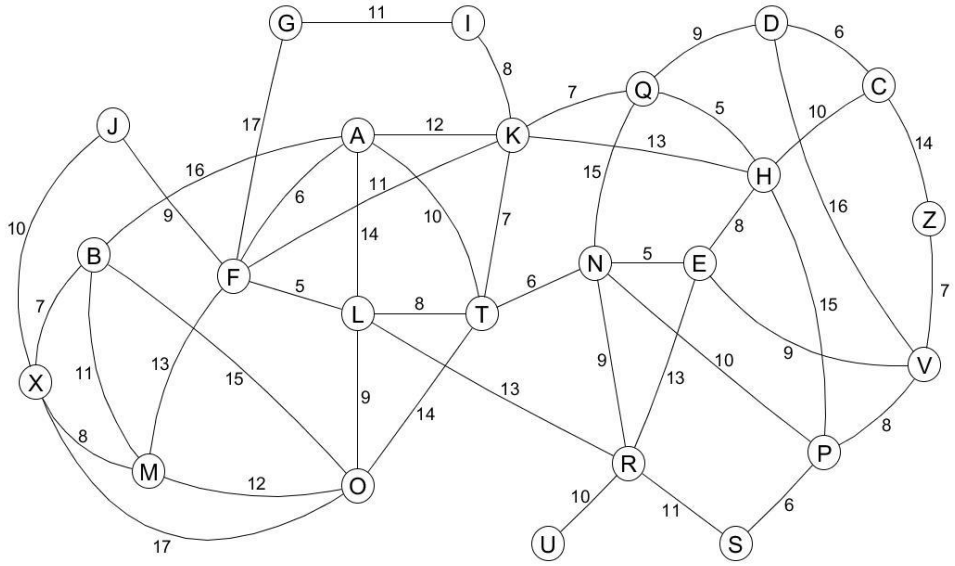


Cadeia 2: grafos 3 e 4

Grafo 3: 25 vértices, 43 arestas



Grafo 4: 24 vértices, 46 arestas



2. Implementar em alguma linguagem de programação à escolha da equipe um dos algoritmos de árvore geradora mínima (Prim, Kruskal). A representação do grafo em meio computacional também fica a critério da equipe (Matriz de Adjacência, Lista de Adjacência, …).
3. Após o retorno da árvore geradora mínima do algoritmo para ambos os grafos - (1 e 2) ou (3 e 4) – a equipe deve implementar um algoritmo de eliminação de vértices de grau 1, sendo:

    3.1. Para quem ficou com a cadeia 1 (grafos 1 e 2): vocês deverão executar o algoritmo de eliminação de vértices com grau 1:

    - No **grafo 1**: <u>cinco vezes</u>
    - No **grafo 2**: <u>quatro vezes</u>

    Observe que somente após cada execução, é que os graus de todos os vértices restantes devem ser atualizados.

    Após todas as execuções, unir o resultado do grafo 1 com o grafo 2. A regra para a união é que: vértices com o mesmo rótulo devem ser somados/mesclados, e arestas paralelas devem ser mantidas. A partir deste ponto já temos a ideia de como será o autômato para ler a cadeia 1.

3.2. Para quem ficou com a cadeia 2 (grafos 3 e 4): vocês deverão executar o algoritmo de eliminação de vértices com grau 1:

- No **grafo 3**: cinco vezes
- No **grafo 4**: quatro vezes

Observe que somente após cada execução, é que os graus de todos os vértices restantes devem ser atualizados.

Após todas as execuções, unir o resultado do grafo 3 com o grafo 4. A regra para a união é que: vértices com o mesmo rótulo devem ser somados/mesclados, e arestas paralelas NÃO devem ser mantidas. Além disso, para resolver 100% o exercício, um loop deverá ser adicionado em algum estado do autômato. A partir deste ponto já temos a ideia de como será o autômato para ler a cadeia 2.

4. Uma vez que foi descoberto o autômato, você deverá desenhá-lo no JFLAP e implementá-lo em alguma linguagem de programação.

2. Regras para entrega do trabalho

✓ **Disciplina Linguagens Formais**

- Entrega: enviar o desenho do autômato e o código fonte no máximo até o dia **22/06 (segunda-feira)** às 23:59:59 para o e-mail fdr.miranda@gmail.com
- Apresentação: apresentar no dia **23/06 (terça-feira)** para toda a turma o desenho do autômato, o código fonte e realizar alguns testes sobre algumas cadeias de proteínas. Eu irei levar também alguns testes válidos e inválidos. Seu autômato precisa acertar, caso contrário, milhões irão morrer.

✓ **Disciplina Teoria dos Grafos**

Fazem parte da entrega:

1. Um Relatório acadêmico (nas normas da instituição) contendo:
   - A descrição das atividades realizadas pela equipe no trabalho;
   - O desenho dos grafos originais no Gephi;
   - O desenho das árvores geradoras mínimas resultantes após a aplicação do primeiro algoritmo;
   - O desenho final da árvore após as extrações dos vértices de graus 1;

- As partes principais dos algoritmos utilizados na primeira parte do trabalho (árvore geradora mínima e extração de vértices em grafos);

2. O código fonte da primeira parte do trabalho.

As avaliações serão compostas por:

- Avaliação parcial (1º bimestre) do andamento do trabalho (Relatório e código) durante a aula de Grafos no dia **27/03 (sexta-feira)**.
- Entrega final (2º bimestre): enviar (Relatório e código) até o dia **18/06 (quinta-feira)** às 23:59:59 para o e-mail camile.bordini@gmail.com
- Defesa individual (2º bimestre): no dia **19/06 (sexta-feira)**.

3. Critérios de avaliação

A pontuação do Projeto Integrador será de 1.0 ponto na nota geral, podendo estender alguns pontos na nota de trabalhos bimestrais. Para as disciplinas, a pontuação respectiva no bimestre será:

- Linguagens formais:
  2º Bimestre: 1.0 ponto na nota geral + 1.5 no trabalho bimestral
- Teoria dos Grafos:
  1º Bimestre: 2.0 pontos no trabalho bimestral
  2º Bimestre: 1.0 ponto na nota geral + 2.0 no trabalho bimestral

Os trabalhos serão avaliados de acordo com os seguintes critérios**:**

- **Linguagens formais**
  - Mostrar andamento do projeto durante o 2º bimestre: 0,5. As datas serão ainda definidas em conjunto com a turma.
  - Entrega do desenho + código fonte via e-mail: 0,2 ponto.
  - Explicar o desenho em sala de aula: 0,2 ponto.
  - Realizar alguns testes em sala de aula: 0,6 ponto.
  - Explicar o código em sala de aula 1,0 ponto.
  - OBS: Serão realizadas perguntas para algum integrante do grupo (na hora eu decido). Para cada erro, será descontado pontos na nota geral de todos. Se algum integrante faltar o mesmo recebe 10% da nota geral.

- **Teoria dos Grafos** (2º Bimestre):
    - Qualidade da escrita do Relatório: 0,7 ponto.
    - Código executando sem erros: 0,8 ponto.
    - Explicar o código em sala de aula: 0,5 ponto.
    - OBS: Serão realizadas perguntas para integrantes do grupo. Se algum integrante faltar o mesmo recebe 10% da nota geral.

4. Referências

- BOAVENTURA P. O. N. **Grafos**: Teoria, Modelos, Algoritmos. 3. ed. São Paulo: Blucher, 2003
- CLAUDIO, D, M.; DIVERIO, T. A; TOSCANI, L. V**. Fundamentos de matemática computacional**. Porto Alegre: Sagra, 1987.
- FORBELLONE, A. L. V.; EBERSPACHER, N. F. **Lógica de Programação: A construção de algoritmos e estrutura de dados**. 3.ed. São Paulo: Pearson Prentice Hall, 2005.
- GEPHI. **Gephi: makes graphs handy**. Disponível em: <https://gephi.org/>.
- GERSTING, J. L. **Fundamentos matemáticos para a ciência da computação**. 4. ed. Rio de Janeiro: LTC, 2001
- GRAHAM, R. L; KNUTH, D. E; PATASHNIK, O. **Matemática concreta: fundamentos para a ciência da computação**. 2.ed. Rio de Janeiro: LTC, 1995
- GUEDES, S. **Lógica de Programação Algorítmica**. São Paulo: Pearson, 2014.
- NETTO, Paulo Oswaldo Boaventura. **Grafos:** Teoria modelos e algoritmos**.** 3 ed. São Paulo: Blucher, 2003.
- PUGA, S.; RISSETTI, G. **Lógica de programação e estrutura de dados:** com aplicações em JAVA. 2 ed. São Paulo: Pearson Prentice Hall, 2009.
- SIMOES, Pereira. J.M.S. **Grafos e Redes:** teoria e algoritmos básicos. Rio de Janeiro: Inter ciência, 2013.
- STEIN, C.; DRYSDALE, R. L.; BOGART, K. **Matemática discreta para Ciência da Computação.** São Paulo: Pearson Prentice Hall, 2013.
- SZWARCFITER J. L. **Grafos e Algoritmos Computacionais**. 2. ed. Rio de Janeiro: Ed. Editora Campus, 1986.
- THOMAS, H. CORMEN et al. **Algoritmos:** teoria e prática. Rio de Janeiro: Campus, 2002.