ATIVIDADE INTEGRADORA

ORIENTAÇÕES/DOCENTE

Disciplina:	Docente:
Análise e Projeto de Algoritmos	Walter Schmidt Marinho

1ª Parte

Introdutória

Tema:	Criação das classes de código do sistema
Objetivo (desta disciplina):	• Dominar a linguagem de programação Java; • Ser capaz de desenvolver aplicativos gráficos e que interajam com o usuário; • Dominar um ambiente de desenvolvimento de *software*, e ferramentas afins, como *debugger* e *unit testing*; • Empregar ferramentas de medição e análise do desempenho de algoritmos e estruturas de dados; • Dominar a teoria de algoritmos e estruturas de dados para ser capaz de decidir quais os melhores a serem empregados na resolução de problemas específicos.
Conteúdos abordados (desta disciplina):	1. O ambiente de desenvolvimento; 2. Programação com testes e *debugging*; 3. Algoritmos de ordenação: seleção, bolha e inserção; 4. Análise de desempenho: *benchmarking* e análise assintótica; 5. Recursividade e *mergesort*; 6. Algoritmos de ordenação avançados: *heapsort* e *quicksort*; 7. Tipos abstratos de dados (TDA) e especificações; 8. Mutabilidade de objetos, igualdade e código de espalhamento; 9. Filas prioritárias; 10. Tabelas de espalhamento e mapas; 11. Busca com retrocesso.

Competências e Habilidades (desta disciplina)	• Compreender os fatos essenciais, os conceitos, os princípios e as teorias relacionadas à Ciência da Computação para o desenvolvimento de *software* e *hardware* e suas aplicações; • Reconhecer a importância do pensamento computacional no cotidiano e sua aplicação em circunstâncias apropriadas e em domínios diversos; • Identificar e analisar requisitos e especificações para problemas específicos e planejar estratégias para suas soluções; • Empregar metodologias que visem garantir critérios de qualidade ao longo de todas as etapas de desenvolvimento de uma solução computacional; • Aplicar temas e princípios recorrentes, como abstração, complexidade, princípio de localidade de referência (*caching*), compartilhamento de recursos, segurança, concorrência, evolução de sistemas, entre outros, e reconhecer que esses temas e princípios são fundamentais à área de Ciência da Computação; • Escolher e aplicar boas práticas e técnicas que conduzam ao raciocínio rigoroso no planejamento, na execução e no acompanhamento, na medição e gerenciamento geral da qualidade de sistemas computacionais; • Aplicar os princípios de interação humano-computador para avaliar e construir uma grande variedade de produtos incluindo interface do usuário, páginas *web*, sistemas multimídia e sistemas móveis.

2ª Parte

Descrição da Atividade

Contextualização

Baseado no estudo de caso (*startup*) que está sendo desenvolvido na disciplina de extensão, serão desenvolvidas as atividades do Projeto integrador. Para cada disciplina do módulo, serão solicitadas tarefas na qual deverão ser cumpridas.

Deverão ser levantados os requisitos funcionais e não funcionais do sistema, sendo que os requisitos funcionais serão representados pelo diagrama de caso de uso e especificação de caso de uso.

Da mesma maneira, com os conceitos de banco de dados adquiridos, é possível manipular informações na base de dados, utilizando os comandos SQL.

Sendo assim, o D.E.R. (Diagrama de Entidade-Relacionamento), deverá ser utilizado para implementação das classes de código no sistema, fazendo a persistência no banco.

Problematização

De acordo com o cenário das *Startups* que está sendo desenvolvido na disciplina de Extensão, agora, você irá conectar o aplicativo com um Banco de Dados para a gravação das informações no aplicativo.

Descrição da Atividade e Forma de Entrega

Considerando as tabelas da Base de Dados, que foram levantadas no D.E.R. (Diagrama de Entidade-Relacionamento), construa as classes de código em uma linguagem de programação orientada a objetos, para fazer a persistência dos dados.

Material de Apoio (vídeos, textos, *links* etc.)

A seguir, veja um exemplo de uma classe de Conexão do Java no Banco MySQL na Figura 1.

Figura 1 - Classe de conexão com o MySQL

```
public Connection conectar(){

    try{
        Class.forName("com.mysql.cj.jdbc.Driver");

        String serverName = "127.0.0.1:3306";
        String mydatabase ="unifil";
        String url = "jdbc:mysql://" + serverName + "/" + mydatabase + "?autoReconnect=true&useSSL=false";

        String username = "root";
        String password = "professor";

        //conn = (Connection) DriverManager.getConnection(url, username, password);
        conn = (Connection) DriverManager.getConnection("jdbc:mysql://localhost:3306/unifil?autoReconnect=true&useSSL=false","root","professor");



        System.out.println("Conexão realizada com sucesso!! - OK");

    }catch(ClassNotFoundException ex){
        Logger.getLogger("erro na ClassForName " + Conexao.class.getName()).log(Level.SEVERE, null, ex);
    }catch (SQLException sql){
        System.out.println("erro no banco " + sql);
    }
    return conn;
}
```

Fonte: o autor (2021)

A Figura 1, retrata o método conectar que retorna uma conexão aberta com o banco de dados. Vale lembrar que esse retorno pode ser utilizado em outra classe ou método para melhor organização do projeto. Dentro do *try*, o método se conecta ao banco por meio da *string* de conexão com os dados do caminho, nome do banco, usuário e senha. No final existe uma mensagem somente para dizer se a conexão foi realmente realizada.

Este é o primeiro passo, após construir a classe de conexão, você deverá implementar os outros métodos que deverão tratar a manipulação dos dados no banco: (inserir, alterar, buscar e excluir).

Critérios de Avaliação

1. Construção da classe de conexão;
2. Classes contendo as instruções S.Q.L. para manipulação das informações (Crud);
3. Interface gráfica (*web*, *desktop* ou mobile) para gerenciar as informações do sistema.

3ª Parte

Bibliografia

CORMEN, Thomas H. *et al*. **Algoritmos**: teoria e prática. Rio de Janeiro: Campus, 2012. 926 p.

DEITEL, Harvey M.; DEITEL, Paul J. **Java**: como programar. 8. ed. São Paulo: Pearson, 2010.

GOODRICH, Michael T.; TAMASSIA, Roberto. **Estruturas de dados e algoritmos em Java**. 4. ed. Porto Alegre: Bookman, 2007. 600 p.