UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO – UNIRIO.
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA – CCET
ESCOLA DE INFORMÁTICA APLICADA – EIA

Programa de disciplina
PM – Programação Modular

CURSO: Bacharelado em Sistemas de Informação

DEPARTAMENTO: Informática Aplicada

DISCIPLINA: Programação Modular

CÓDIGO: TIN0121 TIPO: Obrigatória

CARGA HORÁRIA: 60 horas Nº DE CRÉDITOS: 4 créditos PERÍODO: 6º

PROFESSOR(ES): Paulo Sérgio Medeiros dos Santos

EMENTA:
. Estilos de programação. Processo de desenvolvimento de programas modulares. Princípios de programação modular. Módulos, interfaces, acoplamento. Ferramentas para programação modular. Tipos abstratos de dados. Conceitos de orientação a objetos. Princípios de projeto de sistemas. Padrões de projeto. Tratamento de exceções. Revisões e Inspeções. Técnicas e estratégias de teste de software. Testes de unidade. Testes de integração. Integração de programas. Gerência de Configuração de Software.

PRÉ-REQUISITOS:
Projeto e Construção de Sistemas

OBJETIVOS DA DISCIPLINA:
.• Capacitar o aluno a desenvolver programas em equipe utilizando recursos de sistemas de controle de versões.
• Capacitar o aluno a desenvolver programas modulares de qualidade por meio da aplicação de métodos e técnicas de projeto de software.
• Capacitar o aluno a verificar a qualidade dos programas desenvolvidos por meio de testes e revisões.

METODOLOGIA:
- **Educação online:** A abordagem didático-pedagógica baseia-se na Educação Online, partindo da compreensão de que o conhecimento é socialmente construído, uma "obra aberta" – nesse sentido, os alunos serão desafiados a se apropriar dos conteúdos da disciplina visando a construir novos conhecimentos sobre "Programação Modular".

- **Abordagem Teórico-Prática**: haverá planos de estudos teóricos dirigidos os quais serão apresentados e disponibilizados periodicamente (1 ou 2 semanas) aos alunos. Um conjunto de exercícios também será disponibilizado para auxiliar os estudos. O conteúdo teórico apropriado pelos alunos será aplicado em atividades práticas (construção de um sistema real), onde a turma será organizada como uma equipe de desenvolvimento.

- **Pedagogia de Projetos (aprender-fazendo)**: cada aluno será responsável por uma parte do projeto a ser desenvolvido na disciplina, tal como tipicamente ocorre em em

uma equipe de desenvolvimento em empresas de software. O professor buscará trabalhar como líder de equipe atuando de forma a direcionar os esforços, apoiar em questionamentos sobre o projeto, apontando aspectos teóricos importantes (e.g., modularidade em programas) e promover a facilitação da colaboração entre os alunos. Para isto uma "sequência" de atividades está planejada, a qual guiará o processo de desenvolvimento.

- **Aula Invertida:** os encontros síncronos serão realizados de acordo com a necessidade (provavelmente de forma semanal). Como antes destes encontros os planos de estudos já estarão disponibilizados, é esperado que os alunos realizem os estudos assincronamente ("em casa") de tal forma que os encontros fiquem focados em discussões e dinâmicas relacionadas ao conteúdo estudado e ao projeto desenvolvido.

- **Aprender-ensinando:** em alguns momentos da disciplina os alunos serão solicitados a apresentarem parte dos seus projetos, fundamentando as suas decisões tomadas com base no conteúdo teórico estudado.

- **Aprendizagem Colaborativa:** a atividade de desenvolvimento de software é inerentemente colaborativa. Porém, para facilitar o desenvolvimento das atividades, o sistema a ser desenvolvido será estruturado em módulos que poderão ser desenvolvidos individualmente. Ainda assim, os alunos deverão ser capazes de colaborar de tal forma a permitir a integração correta dos módulos.

- **Metacognição**: na medida em que os alunos aprenderão a utilizar diversas ferramentas de desenvolvimento, os alunos irão manter um "diário da disciplina" em que deverão registrar, ao longo do aprendizado de cada ferramenta, as suas reflexões sobre os caminhos e conteúdos que direcionaram a sua aprendizagem (certezas provisórias) e o que imaginam como próximos passos (novos dilemas). Ao final da disciplina, cada aluno entrega um pequeno texto (relatório de 4 a 6 páginas) descrevendo o seu percurso de aprendizagem, apontando dicas e informações importantes para os novos alunos do próximo período.

CONTEÚDO PROGRAMÁTICO:

- Gerência de configuração em projetos de software
  - Sistemas de controle de versão
  - Sistemas de controle de requisição
  - Sistemas de construção (build)
  - Integração contínua
- Aspectos de Qualidade de Programas
  - Características de um programa bem construído
  - Nomes
  - Funções (nome, tamanho, nível de abstração, assinatura)
  - Princípio da eliminação de replicação de código
  - Redução de complexidade na implementação de funções
  - Comentários

- Tratamento de Exceções
- Projeto de software
  - Princípios de projeto de software
  - Projeto arquitetural
  - Padrões arquiteturais
  - Padrões de projeto
  - Projeto de pacotes
  - Projeto de classes
  - Refactoring
- Manutenção de software
- Testes
  - Importância e papel dos testes
  - Testes de unidade
  - Testes de integração
  - Testes de sistema
  - Testes unitários automatizados
  - Test Driven Development
  - Outros tipos de testes
  - Técnicas para elaboração de testes
  - Testes Funcionais x Testes Estruturais
  - Avaliação da cobertura dos testes
  - Registro de falhas
- Revisões e inspeções

CRONOGRAMA:

SEMANA 1	Gerência de configuração em projetos de software
SEMANA 2	Gerência de configuração em projetos de software
SEMANA 3	Aspectos de Qualidade de Programas
SEMANA 4	Aspectos de Qualidade de Programas
SEMANA 5	Aspectos de Qualidade de Programas
SEMANA 6	Projeto de software
SEMANA 7	Projeto de software
SEMANA 8	Projeto de software
SEMANA 9	Projeto de software
SEMANA 10	Manutenção e teste
SEMANA 11	Manutenção e teste
SEMANA 12	Manutenção e teste
SEMANA 13	Manutenção e teste
SEMANA 14	Apresentação de trabalhos, divulgação de notas e definição de melhorias no projeto para alunos com nota entre 4,0 e 7,0.
SEMANA 15	Entrega da versão final dos trabalhos.

EXAMES E AVALIAÇÕES:
10% Exercícios resolvidos
50% Projeto entregue na disciplina
20% Participação em discussões sobre o projeto e tomadas de decisão pelo "time" (turma)
20% Relatório sobre o seu caminho de aprendizagem (4 a 6 páginas), entregue ao final do período.

FERRAMENTAS DIGITAIS UTILIZADAS:
Google Classroom
Google Meeting
Aulas gravadas
Mentimeter

BIBLIOGRAFIA

Martin, R. **Clean Code: A Handbook of Agile Software Craftsmanship.** Prentice hall, 2008.
Larman, Craig. **Utilizando UML e Padrões – Uma introdução à análise e ao projeto orientados a objetos** – 3ª Edição. Bookman. 1999.
McConnell, Steve. Code Complete 2. Microsoft Press, 2004.
Wiegers, K.E., **Peer Reviews in Software – A Practical Guide**. Addison-Wesley, 1ª Edição, 2002.

Assinatura do professor: