

Plano de Ensino

1) Identificação

Disciplina: INE5605 - Desenvolvimento de Sistemas Orientados a Objetos I
Turma(s): 02238A
Carga horária: 108 horas-aula Teóricas: 48 Práticas: 60
Período: 1º semestre de 2025

2) Cursos

- Sistemas de Informação (238)

3) Requisitos

- Sistemas de Informação (238)
 - INE5603 - Introdução à Programação Orientada a Objetos
 - INE5603 - Introdução à Programação Orientada a Objetos

4) Professores

- Andre Beims Brascher (andre.brascher@ufsc.br)
- Douglas Hiura Longo (douglas.hiura@ufsc.br)

5) Ementa

Implementação de pequenos projetos com programação orientada a objetos. Sistemas de Tipo; Sistemas de Tratamento de Exceções.

6) Objetivos

Geral: Capacitar os estudantes a desenvolver sistemas utilizando técnicas da programação orientada a objetos e arcabouços básicos de software

Específicos:

- Compreender os principais conceitos sobre orientação a objetos em sistemas de informação;
- Aprender técnicas de reuso de software;
- Dominar a utilização de arcabouços básicos de software; e
- Saber implementar sistemas empregando os conceitos da orientação a objetos

7) Conteúdo Programático

- 7.1) Introdução ao desenvolvimento de sistemas reusáveis de software [6 horas-aula]
- 7.2) Conceitos e mecanismos da programação orientada a objetos [24 horas-aula]
 - Objetos e classes
 - Associação, agregação e composição
 - Herança e polimorfismo
 - Classes abstratas
 - Diagramas de classes
- 7.3) Técnicas de uso comum em sistemas orientados a objetos [36 horas-aula]
 - Interface gráfica com o usuário
 - Tratamento de exceções
 - Listas e dicionários
 - Persistência de dados e objetos (serialização)
- 7.4) Práticas de Desenvolvimento de Software [42 horas-aula]
 - Introdução a práticas/técnicas de desenvolvimento orientado a objetos
 - Arquitetura em camadas e padrões de projeto
 - Construção de sistemas de software que demonstrem as características básicas da orientação a objetos

8) Metodologia

As aulas serão desenvolvidas da seguinte forma:

- exposição de conteúdo teórico;
 - prática de programação na forma de exercícios assistidos em laboratório;
 - prática de programação na forma de exercícios automatizados na plataforma Moodle;
 - desenvolvimento (parcial/total) de pequenos projetos de software com ênfase em tópicos da disciplina;
 - desenvolvimento de projetos de software refletindo a extensão dos tópicos abordados na disciplina;
- A ferramenta de EAD Moodle será utilizada como repositório oficial de material de aula. A disciplina no Moodle também detalhará o cronograma deste plano de ensino, servindo para marcar as datas exatas das avaliações e documentar possíveis alterações de cronograma advindas de necessidades identificadas no decorrer do semestre. Todos os slides a serem utilizadas durante a disciplina serão disponibilizadas no Moodle sob a Licença 2.5 Brasil Creative Commons Atribuição-Uso Não-Comercial-Compartilhamento.
- A disciplina pode contar com apoio de **estagiário de docência**, responsável pelas seguintes atividades: revisão dos roteiros das aulas práticas, acompanhamento dos alunos na execução dos trabalhos práticos em horário extraclasse; acompanhamento dos alunos na execução dos experimentos durante as aulas práticas, auxílio aos alunos para dirimir dúvidas das aulas teóricas, e; auxílio aos professores na aplicação das avaliações.

9) Avaliação

A Média final (MF) é calculada obedecendo a seguinte equação:

$$MF = P * 0,35 + T1 * 0,30 + T2 * 0,25 + ME * 0,10$$

onde:

P - Prova sem consulta, de conteúdo teórico e/ou prático

T1 e T2 - Trabalhos práticos de programação em duplas entregues e apresentados.

ME - Média das notas dos exercícios práticos de programação.

Detalhamento das avaliações:

A **Prova (P)** será realizada presencialmente em sala de aula ou em laboratório (utilizando o Moodle Provas). No caso de provas realizadas no Moodle Provas, com avaliação automatizada, serão cobrados: a sintaxe e o uso da linguagem de programação, sendo que erros de sintaxe que impeçam a compilação do código podem resultar em nota zero para o aluno.

Os **Trabalhos (T1 e T2)** serão realizados em duplas (2 alunos) e relacionados à prática do desenvolvimento software orientado a objetos. É de responsabilidade do aluno entregar o trabalho na forma correta. Arquivos corrompidos, ilegíveis ou em formato diferente do especificado não serão considerados.

A apresentação dos trabalhos (T1 e T2) será realizada através da gravação de um vídeo, entregue junto com o código. Neste vídeo, deve ser demonstrado o sistema em execução e cada membro da equipe deve apresentar a parte que desenvolveu do sistema. A nota de cada membro da equipe será individual, dependendo da sua participação individual no desenvolvimento do trabalho (demonstrado através de commits no git) e na sua apresentação. Os trabalhos devem ser entregues utilizando-se a plataforma Moodle, não sendo aceitas outras formas de entrega/envio. **Não serão aceitos trabalhos fora do prazo estabelecido.**

A **Média dos Exercícios (ME)** será resultado da média das notas dos exercícios realizados individualmente pelos alunos e entregues para o professor nas datas planejadas. Serão realizados de 07 a 10 exercícios práticos de programação durante o semestre. Os exercícios poderão ser automatizados utilizando-se a plataforma EAD Moodle.

Conforme Resolução Nº 17/CUn/97 Art. 70 § 4º, ao aluno que não comparecer às avaliações ou não apresentar trabalhos no prazo estabelecido será atribuída nota 0 (zero). O aluno, que por motivo de força maior e plenamente justificado, deixar de realizar avaliações previstas no plano de ensino, deverá formalizar pedido de avaliação à Chefia do departamento INE, dentro do prazo de 3 (três) dias úteis. Cessado o motivo que impediu a realização da avaliação, o aluno, se autorizado pelo INE, deverá fazê-la conforme agendado pelo professor. Conforme Resolução Nº 17/CUn/97, será obrigatória a frequência às atividades correspondentes a disciplina, ficando nela reprovado o aluno que não comparecer, no mínimo, a 75% das mesmas. O professor registrará a frequência, para cada aula no Moodle. Cabe ao aluno acompanhar, junto ao professor e no Moodle, o registro da sua frequência às aulas.

Somente será atribuída presença em uma aula ao aluno que dela participar pelo período majoritário da sua duração.

Nesta disciplina **NÃO** será realizada Prova de Recuperação ao final do semestre.

Não é prevista atividade de recuperação para esta turma, nos termos previstos no art. 70, parágrafo 2º, da Resolução 17/CUn/97, uma vez que cumpre pelo menos um dos seguintes requisitos:

- ter pelo menos 50% de carga prática;
- ter pelo menos 50% do peso da média final originado de trabalho prático;
- ter a inadequação da aplicação de avaliação de recuperação reconhecida pelo colegiado do curso, a partir da avaliação de solicitação fundamentada de dispensa de avaliação de recuperação, encaminhada pelo(s) professor(es) autor(es) do respectivo plano de ensino, para disciplinas com carga prática prevista no programa da disciplina, com nota de trabalho prático considerada no cálculo da média final e que não tenham cumprido um dos requisitos anteriores.

10) Cronograma

As datas PREVISTAS são as seguintes:

Semana 1: Apresentação da Disciplina e Introdução

Semanas 2 e 3: Classes e Objetos

Semanas 4 e 5: Associação, Agregação, Composição

Semanas 6 e 7: Classes Abstratas e Polimorfismo

Semanas 8 e 9: Estruturando Sistemas

Semana 10: Tratamento de Exceções

Semanas 11 e 12: Elaboração e apresentação do Trabalho T1

Semana 13: Prova

Semanas 14 e 15: Interface gráfica com o usuário

Semana 16: Persistência de dados e objetos

Semanas 17 e 18: Elaboração e apresentação do Trabalho T2

* (Semanas sem contar feriados)

O cronograma atualizado estará disponível na ferramenta Moodle.

11) Bibliografia Básica

- HALL, Tim; STACEY, J. P. Python 3 for absolute beginners. Apress, 2010. Disponível online (utilizar VPN UFSC): <http://dx.doi.org/10.1007/978-1-4302-1633-9>
- BORGES, Luiz Eduardo. Python: para desenvolvedores. São Paulo: Novatec, 2014. Disponível online: https://ark4n.files.wordpress.com/2010/01/python_para_desenvolvedores_2ed.pdf

12) Bibliografia Complementar

- ALCHIN, Marty. Pro Python. New York: Apress, 2010. Disponível online (utilizar VPN UFSC): <http://dx.doi.org/10.1007/978-1-4302-2758-8>
- Design Patterns – Elements of Reusable Object-Oriented Software. E. Gamma, R. Helm, R. Johnson, J. Vlissides. Addison-Wesley, 1995.
- WAZLAWICK, Raul S. Introdução a Algoritmos e Programação com Python. São Paulo: Elsevier, 2017.
- SILVA, Ricardo Pereira e. UML 2 – Modelagem Orientada a Objetos. Editora VisualBooks, 2007.
- WAZLAWICK, R. Análise e Projeto de Sistemas de Informação Orientados a Objetos. Editora Campus, 2004.