Universidade Federal de Santa Catarina
Centro Tecnológico
Departamento de Informática e Estatística

Plano de Ensino

1) Identificação

Disciplina: INE5410 - Programação Concorrente
Turma(s): 03208A, 03208B
Carga horária: 72 horas-aula Teóricas: 36 Práticas: 36
Período: 1º semestre de 2020

2) Cursos

- Ciências da Computação (208)

3) Requisitos

- Ciências da Computação (208)
  - INE5404 - Programação Orientada a Objetos II

4) Professores

- Márcio Bastos Castro (marcio.castro@ufsc.br)
- Odorico Machado Mendizabal (odorico.mendizabal@ufsc.br)

5) Ementa

Multiprogramação. Multitarefa. Execução concorrente. Recursos compartilhados e exclusão mútua. Regiões críticas. Coordenação de processos e threads. Semáforos. Monitores. Troca de mensagem. Programação concorrente orientada a objeto. Deadlock. Modelos de computação concorrente.

6) Objetivos

Geral: Compreender os princípios gerais da programação concorrente.

Específicos:

- Entender a importância atual do conhecimento da computação concorrente.
- Conhecer os principais conceitos, problemas e ferramentas da programação concorrente.
- Exercitar a elaboração de programas concorrentes.

7) Conteúdo Programático

7.1) Introdução [08 horas-aula]

7.2) Fundamentos de Programação Concorrente [24 horas-aula]
- Processos
- Threads
- Exclusão Mútua
- Semáforos
- Deadlocks

7.3) Tecnologias para Programação Concorrente [30 horas-aula]
- APIs para o Desenvolvimento de Aplicações Paralelas
- Concorrência em Linguagens Orientadas a Objetos

7.4) Modelagem de Programas Concorrentes [10 horas-aula]

8) Metodologia

7.1. Visão Geral

Os principais conceitos sobre programação concorrente serão abordados através de aulas expositivas. Para incentivar o auto-estudo, o professor poderá selecionar um ou mais tópicos da disciplina a ser(em) coberto(s) através de estudo(s) dirigido(s) ou atividade(s) de sala de aula invertida. Ao longo do curso, serão propostos exercícios a serem resolvidos pelos alunos para a fixação dos conteúdos ministrados.

As atividades pedagógicas não presenciais serão realizadas através de duas modalidades de comunicação: **síncrona** e **assíncrona**. Na **modalidade síncrona**, o professor e os alunos estarão conectados **simultaneamente** a uma ferramenta de webconferência ou ao *Moodle* em dia/horário estabelecido na grade horária da disciplina e

conforme cronograma disponível no *Moodle*. Na **modalidade assíncrona**, o professor poderá estipular um prazo para que os alunos realizem uma atividade e ficará disponível para atendimento através dos recursos de tecnologias da informação que permitam a interação aluno-professor de maneira assíncrona.

7.2. Recursos Didáticos

Poderão ser utilizados os seguintes recursos de tecnologias da informação e de comunicação nas atividades pedagógicas:

· **Atividades síncronas**: *Google Meet* (ou eventualmente, *Conferência Web - RNP*) e *Moodle*;
· **Atividades assíncronas**: *Moodle*, *Youtube* e *Github Classroom*.

As aulas expositivas serão realizadas de maneira assíncrona através de vídeos disponibilizados pelo professor no *Youtube*. Periodicamente, serão marcados encontros síncronos para discussão mais aprofundada sobre os assuntos tratados nos vídeos. O conteúdo dos encontros síncronos será gravado e disponibilizado também no *Youtube*.

7.3. Estagiário de Docência

Além do professor, responsável pela disciplina, está prevista a participação de um estagiário de docência, que irá trabalhar juntamente com o professor no desenvolvimento e avaliação das atividades.

9) Avaliação

8.1. Instrumentos de Avaliação

Os alunos serão avaliados através dos seguintes instrumentos de avaliação:

· **Prova regular (P)**: Prova individual a ser realizada em horário estabelecido na grade horária da disciplina de forma **síncrona**;
· **Trabalhos de implementação (T)**: Trabalhos práticos de programação de forma **assíncrona** com prazos para entrega definidos pelo professor.
· **Atividades de fixação (AF)**: Atividades de fixação de conteúdo de forma **assíncrona** com prazos para entrega definidos pelo professor. Estas atividades compreendem o desenvolvimento de código, questionários sobre partes do conteúdo, estudos dirigidos ou atividades de sala de aula invertida.

Os seguintes critérios serão observados para fins de avaliação individual dos alunos:

· Compreensão dos conteúdos discutidos;
· Clareza e qualidade das soluções das atividades práticas;
· Participação efetiva nas aulas síncronas e nas atividades práticas;
· Cumprimento de prazos de entrega das atividades e trabalhos práticos.

No caso de impossibilidade de realização da prova por motivo justificável, devidamente comprovado, o aluno deverá solicitar ao professor, no prazo de 72 horas, a autorização para substituí-la, através de uma prova substitutiva única (**PS**), cujo conteúdo avaliado será equivalente ao conteúdo total da disciplina. A **PS** será realizada posteriormente em data definida no cronograma da disciplina. **A nota da prova perdida será substituída pela nota obtida na PS**.

A avaliação dos Ts será sempre **individual**, mesmo em caso de Ts realizados em grupos, e será feita mediante: (i) entrega de relatório descritivo com detalhamento da solução proposta e resultados observados **ou** (ii) apresentação do trabalho e respostas aos questionamentos do professor de forma síncrona. A entrega e acompanhamento dos trabalhos será realizada pelo *Github Classroom*. A forma e os critérios de avaliação dos Ts serão definidos em seus enunciados.

A avaliação das AFs será sempre **individual**, mesmo em caso de AFs realizadas em grupos. De modo geral, a entrega das soluções das AFs será feita através do *Moodle*. A forma e os critérios de avaliação das AFs serão definidos em seus enunciados.

8.2. Cálculo da Frequência e Média Final (MF)

A frequência será contabilizada pelo número de atividades avaliativas (P, Ts e AFs) realizadas. O número de AFs (*n*) pode variar de acordo com as atividades, mas serão realizadas pelo menos 12 AFs.

Os alunos com frequência inferior à 75% serão considerados reprovados por frequência insuficiente (**FI**). Os alunos com frequência suficiente (**FS**) terão sua média final (**MF**) calculada da seguinte forma:

· **MF** = 0,3 * P + 0,4 * (T1 + T2) + 0,3 * (AF1 + AF2 + ... + AFn)

8.3. Prova de Recuperação (REC)

Conforme parágrafo 2º do artigo 70 da Resolução 17/CUn/97, o aluno com frequência suficiente (FS) e média final no período (**MF**) entre 3,0 e 5,5 terá direito a uma nova avaliação ao final do semestre (**REC**), sendo a nota final (**NF**) calculada conforme parágrafo 3º do artigo 71 desta resolução, ou seja: **NF = (MF + REC) / 2**.

10) Cronograma

O cronograma da disciplina será disponibilizado no *Moodle*. O conteúdo programático será dividido em 4 módulos e o cronograma indicará a quantidade de semanas necessárias para cobrir o conteúdo de cada módulo. Além disso, o cronograma indicará as datas em que ocorrerão as atividades síncronas.

As datas aproximadas para a realização das provas são:

· **Prova**: Penúltima semana do calendário acadêmico;
· **Prova Substitutiva Única**: Última semana do calendário acadêmico;
· **Prova de Recuperação**: Última semana do calendário acadêmico.

11) Bibliografia Básica

- TANENBAUM, Andrew S. Sistemas operacionais modernos. 3. ed. Rio de Janeiro: Pearson Prentice Hall, 2010. xiii, 653 p. ISBN 9788576052371.
- SILBERSCHATZ, Abraham; GALVIN, Peter B.; GAGNE, Greg. Fundamentos de sistemas operacionais: princípios básicos. Rio de Janeiro: LTC, c2011. xvi, 432 p. ISBN 9788521622055.
- OLIVEIRA, Rômulo Silva de; CARISSINI, Alexandre da Silva; TOSCANI, Simão Sirineo. Sistemas operacionais. 4. ed. Porto Alegre: Bookman, 2010. xii, 374 p. (Livros didáticos ; 11). ISBN 9788577805211.

12) Bibliografia Complementar

- PACHECO, Peter. An Introduction to Parallel Programming. 1. ed. Burlington: Morgan Kaufmann, 2011. 392 p. ISBN 0123742609.
- KERNIGHAN, Brian W.; RITCHIE, Dennis M. C, a linguagem de programação. 4. ed. Porto Alegre: EDISA; Rio de Janeiro: Campus, 1988. 208p ISBN 8570014104
- DEITEL, Paul J.; DEITEL, Harvey M. C: como programar. 6. ed. São Paulo: Pearson Prentice Hall, 2011. xxvii, 818 p. ISBN 9788576059349.
- LEA, Douglas. Concurrent programming in Java: design principles and patterns. 2nd ed. Reading: Addison Wesley, c2000. x, 411 p. ISBN 0-201-31009-0.
- DEITEL, Harvey M.; DEITEL, Paul J. Java como programar. 8. ed. São Paulo: Pearson Prentice Hall, 2010. xxix, 1144 p. ISBN 9788576055631