

Plano de desenvolvimento da disciplina

Instituto de Computação — Universidade de Campinas

Estrutura de Dados

Segundo Semestre de 2022 – MC202E

Prof. *Lehilton Lelis Chaves Pedrosa*

Monitores *Guilherme (PED)* e *Ieremies, Lucas, Thiago (PAD)*

Introdução

Este é o plano de desenvolvimento da disciplina e um guia de estudos. Leia-o com atenção e consulte este documento durante todo o semestre. Também, sempre acompanhe os avisos na página da disciplina.

Objetivos

Ao final do curso, @ estudante deverá ser capaz de:

- desenvolver e testar programas usando estruturas de dados conhecidas: listas, árvores, grafos;
- escolher as estruturas de dados mais adequadas e eficientes para problemas computacionais.

Pré-requisitos

Ao início do curso, @ estudante deverá ser capaz de:

- descrever problemas computacionais, computabilidade e limitações;
- descrever organização básica de computadores e de sistemas de software;
- utilizar estruturas de dados elementares: variável, lista, matriz
- escrever algoritmos iterativos e recursivos;
- implementar e testar programas escritos em Python ou em C.

Além disso:

- interpretar e descrever problemas e situações textualmente;
- desenvolver e executar um cronograma de estudo contínuo;
- consultar, organizar e resumir fontes de pesquisa diversas (bibliotecas e internet).

Atividades

Aula expositiva

As aulas ocorrerão presencialmente. O professor disponibilizará conteúdo em texto ou vídeo com antecedência, que deve ser estudado **antes e depois** das aulas correspondentes. Durante o horário das aulas haverá revisão e discussão dos conteúdos e exercícios e @s estudantes devem participar levantando questões, dúvidas, sugestões, ou possivelmente resolvendo os problemas solicitados. A qualquer momento, estudantes podem tirar dúvidas com o professor.

Listas de exercícios para fixação

Serão propostas diversas listas de exercícios para fixação para serem realizados durante e após as aulas. O conteúdo das listas é considerado parte integrante do curso. Há tanto questões teóricas quanto práticas. Implementar os programas solicitados nas questões serve como um exercício mais simples antes das tarefas de programação. Como as questões são abertas, não há gabarito. Esses exercícios para fixação não serão corrigidos e não é obrigatório realizar todos eles.

Recomenda-se tentar fazer esses exercícios tanto individualmente quanto em grupo. Se forem fazer em grupo, procurem não criar grupos com mais do que **quatro** estudantes, de forma que todos participem ativamente da resolução dos exercícios.

Tarefas

Haverá diversas tarefas individuais que servirão de avaliação da disciplina. O número previsto de tarefas é 10 e pode mudar a depender do andamento da turma. Cada tarefa é um conjunto de um ou mais exercícios de programação ou trabalhos mais elaborados, que deverão ser

implementados na linguagem de programação C. Todas as tarefas devem ser desenvolvidas em um repositório de controle de versões Git, utilizando a infraestrutura do IC <https://gitlab.ic.unicamp.br/>. Para criar uma conta, logue-se no sistema utilizando a senha fornecida pelo IC. Cada estudante terá um repositório de nome `raXXXXXX`, em que `XXXXXX` corresponde ao número de RA.

A sequência das atividades será a seguinte:

1. O enunciado da tarefa é publicado após ou simultaneamente os conteúdos correspondentes serem ministrados.
2. Um conjunto de arquivos auxiliares da tarefa será inserido automaticamente no repositório pessoal em um diretório de nome `tarefaNN`, onde `NN` é o número da tarefa. Esses arquivos auxiliares podem ser acessados bastando executar o comando `git pull` na cópia de trabalho pessoal.
3. @ estudante realiza os vários exercícios da tarefa, alterando ou incluindo novos arquivos no repositório correspondente. À medida em que cada pequeno passo for realizado, deve-se fazer um `git commit` com uma mensagem adequada. É **obrigatório** que todo o desenvolvimento da tarefa seja registrado utilizando o controle de versões, isso é, *não* escreva suas tarefas e copie de outros locais e faça commit mesmo dos programas não prontos, parcialmente escritos. *Adicione mensagens de commit adequadas descrevendo as modificações realizadas.* Tarefas que não tenham histórico de commits ou submetidas por outros meios serão desconsideradas.
4. A cada pequena parte implementada, deve-se testar o programa. Para isso, pode-se executar em um terminal `python3 testar.py`, o que executará um script de teste fornecido juntamente com os arquivos auxiliares. Além desses, pode ser necessário criar outros testes.
5. Terminada a hora de trabalho, @ estudante executa o comando `git push`. Isso enviará as alterações para o repositório remoto. Além disso, ativar um script de correção, que testará automaticamente a tarefa (utilizando os mesmos arquivos auxiliares e/ou outros testes). O resultado dessa correção ficará anotada na planilha de notas disponibilizada para cada aluno.

As tarefas serão corrigidas pelo sistema de correção automática e por um monitor. Até o final do semestre, o professor irá verificar as correções e

poderá revisar a nota das tarefas para cima ou para baixo. Uma vez terminada a tarefa e após ela ter passado nos testes automáticos, deve-se **solicitar a correção ao monitor** clicando-se no botão correspondente da interface de notas (com exceção das tarefas que forem marcadas para correção automática apenas). Algumas tarefas determinadas (especificadas no enunciado), deverão ser *apresentadas pessoalmente* a um monitor presencialmente ou por meio de videoconferência de maneira sucinta nos horários de atendimento disponíveis. Além disso, @s estudantes são encorajad@s a mostrar e conversar sobre suas dúvidas e seus códigos com os monitores, via chat e videoconferência, sempre que desejarem nos canais de atendimento.

A cada tarefa será atribuído um conceito com os seguintes valores e significados:

- **A** (valor 10): a tarefa foi executada satisfatoriamente; deve-se prosseguir à próxima atividade;
- **B** (valor 8): os objetivos mínimos foram alcançados, mas há questões pontuais que podem ser melhoradas; deve-se prosseguir à próxima atividade;
- **C** (valor 6): nem todos os objetivos mínimos foram alcançados; a tarefa é considerada aprovada, mas recomenda-se corrigi-la e submeter novamente;
- **D** (valor 0): a tarefa não foi realizada, ou os objetivos não foram satisfeitos; é obrigatório refazer a tarefa.

Prazos

A disciplina adotará um método de avaliação contínua e individualizado. Assim, as tarefas serão divulgadas continuamente, de acordo com os conteúdos ministrados, mas cada um@ poderá levar mais tempo ou menos tempo para executá-las, de acordo com seu aprendizado sobre o conteúdo. As regras são as seguintes:

- Cada tarefa terá um prazo recomendado de pelo menos uma semana a partir da publicação. Se uma tarefa for entregue depois do prazo recomendado, será **descontado 15%** da nota correspondente.
- Todas as tarefas são **obrigatórias** e devem ser entregues e apresentadas até **11/12/2022**.
- Uma tarefa só será corrigida após as tarefas anteriores tiverem sido realizadas corretamente (tiverem recebido conceitos A, B ou C).

- A correção de novas tarefas também não será feita se alguma tarefa anterior receber conceito D após ser corrigida manualmente. Nesse caso, deve-se corrigir a tarefa e solicitar revisão a algum monitor via Discord.

É responsabilidade de cada um@ organizar e dividir o tempo para realizar cada atividade. É proibido acumular tarefas deliberadamente: estudantes com tarefas atrasadas que não tiverem nenhuma atividade no repositório por duas semanas terão a correção de novas tarefas bloqueadas até que apresentem justificativa ao professor (o professor indicará isso como um comentário no relatório da tarefa 0). Caso um estudante não mantenha atividade no repositório por três semanas e não apresente justificativa, será presumida a desistência da disciplina e a correção de suas tarefas será bloqueada permanentemente. Recomenda-se fortemente que a maior parte dos exercícios seja realizada durante as aulas de laboratório e que estudantes não deixem tarefas pendentes.

Avaliação

A nota da disciplina será a média ponderada de duas partes

1. A média aritmética M das notas de todas as tarefas, valendo de 0 a 10.
2. A nota de participação P do estudante atribuída pessoalmente pelo professor, valendo de 0 a 10.

Serão considerados aprovados os estudantes que tiverem pelo menos 75% de presença e que obtiverem conceito pelo menos C em todas as tarefas. Nesse caso, a nota de aproveitamento será $A=0,7M+0,3P$. Do contrário, a nota de aproveitamento do semestre será $A=\text{mínimo}\{4,M\}$.

Nota de participação

Para calcular a nota de participação serão levadas em conta tanto a participação nas aulas online quanto a realização das demais atividades. Isso será monitorado considerando atividade regular no repositório git bem como a presença e participação nas aulas, no chat de discussão e no atendimento. Para garantir a nota máxima de participação, deve-se:

- Manter atividade regular no repositório git, realizando inclusões e alterações de pelo menos **200** linhas de código em arquivos .c ou .h no repositório em cada semana (o número de linhas considerado pode ser

menor em determinadas semanas a critério do professor). Vocês podem incluir no repositório trechos de códigos das **listas de exercícios para fixação** realizados em grupo, mas é obrigatório fazer um comentário indicando as pessoas com quem colaborou.

- Participar continuamente das aulas e/ou dos canais de chat, compartilhando dúvidas, respondendo perguntas ou fazendo comentários pertinentes sobre a disciplina.
- Apresentar uma tarefa (a partir da tarefa 5) ao professor aumentará a nota de participação em 2 pontos. Não será uma correção ou atribuição de nota da tarefa. O objetivo é dar uma oportunidade para conversar e receber feedback diretamente do professor sobre alguma atividade mais elaborada realizada. O professor estará disponível na aula de laboratório apenas alguns dias a serem especificados e atenderá na medida do possível e dentro do horário.

Exame

Aquel@ estudante com pelo menos 75% de presença e $A \geq 2,5$ que não tiver conseguido completar todas as tarefas até o prazo poderá realizar exame. Como exame, o professor solicitará a realização de um subconjunto das tarefas e atribuirá nota E . As tarefas do exame deverão ser submetidas no sistema e apresentadas ao professor por vídeoconferência até o dia 20/12/2022 em horário acordado com o professor (o laboratório ficará disponível no horário de aula para os estudantes que não dispuserem de computador pessoal). Para estudantes que fizerem o exame, a nota final será **mínimo**{ $5, (A+E)/2$ }.

Fraude

Em caso de fraude (plágio, atestado falso, assinar lista por colegas, abandonar aula após assinar, usar bibliotecas não permitidas, copiar quaisquer trechos da internet sem autorização expressa, mostrar ou distribuir laboratório de programação individual, cola independentemente de origem, consulta a material proibido, tentar burlar testes automáticos, etc.), os envolvidos serão reprovados com nota 0 e será registrada a ocorrência no histórico escolar. Fique atento:

- Cada um@ é responsável por manter seguros os arquivos de seu repositório. Não compartilhe sua senha nem deixe cópias de arquivos em computadores compartilhados. **Todas** as tarefas são individuais.

- É proibido utilizar ajuda de terceiros (colegas, amigos, professores particulares, etc.) para realizar as tarefas sem autorização expressa do professor da disciplina.
- É permitido compartilhar e discutir sobre trechos de código apenas dos exercícios para fixação. Sempre que utilizar trechos de código de outras pessoas ou de outras fontes, deve-se indicar isso explicitamente.
- É permitido tirar dúvidas e discutir a respeito de tarefas de programação com colegas e monitores. Mas toda discussão deve ser feita em canais abertos no Discord e não é permitido compartilhar algoritmos para resolver a tarefa (em português, código ou pseudocódigo) nem ideias prontas de solução etc.
- O material disponibilizado e os trabalhos submetidos nas avaliações são para uso exclusivo da disciplina. Não compartilhem seus trabalhos mesmo após o término da disciplina. Se desejar publicar algum trabalho que realizou, converse antes com o professor.
- Cláusula de arrependimento: se você cometer qualquer tipo de fraude, mas reconhecê-la **antes** de ser comunicad@, então o professor irá relaxar as ações acima por ações locais.

Conteúdo e bibliografia

O professor não seguirá nenhum livro específico. O conteúdo abordado é coberto por capítulos dos livros listados a seguir.

1. T. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein. Algoritmos - Teoria e Prática. Campus, 2002.
2. R. Sedgewick, Algorithms in C. Addison-Wesley, 1990.
3. D. E. Knuth, The Art of Computer Programming, Vol I. Addison-Wesley, 1978.
4. A. V. Aho, J. E. Hopcroft, J. Ullmann. Data Structures and Algorithms. Addison-Wesley, 1983.
5. W. Celes, R. Cerqueira, J. L. Rangel. Introdução a Estruturas de Dados. Campus, 2004.
6. M. J. Folk e B. Zoellick. File Structures. Addison-Wesley, 1992.
7. F. Lorenzi, P. N. de Mattos, T. P. de Carvalho. Estruturas de Dados. Thomson, 2007.
8. S. L. Pereira. Estruturas de Dados Fundamentais. Érica, 1996.
9. E. M. Reingold e W. J. Hanson, Data Structures. Little-Brown, 1983.
10. J. L. Szwarcfiter e L. Markenzon. Estruturas de Dados e Seus Algoritmos. Editora LTC, 1994.
11. N. Wirth, Algorithms + Data Structures = Programs. Prentice-Hall, 1976.

12. A. M. Tenenbaum. Estruturas de Dados Usando C. Makron Books, 1995.
13. N. Ziviani. Projeto de Algoritmos. Thomson, 2004.

As principais referências são os livros [1] e [2] (particularmente para conteúdos de árvores rubro-negra, árvores B, etc). O livro [3] é um livro clássico em computação e aborda tanto estruturas elementares quanto estruturas mais avançadas (como generalizações de árvores e listas). Alguns exercícios propostos foram retirados ou inspirados pelas demais referências (em particular, [4], [10], [11] e [12]). Embora estejam listadas as edições clássicas dos livros, as edições mais novas também podem ser consultadas sem prejuízo do conteúdo a ser visto; há alguns exemplares disponíveis na biblioteca. Além dos livros, pesquisar com cuidado na internet também é um bom jeito de aprender. Procure os verbetes correspondentes às estruturas vistas na Wikipédia. Você pode visualizar vários dos algoritmos estudados no site em <http://visualgo.net/>.

Material didático

A página da disciplina conterà um material de apoio para estudo, que inclui um tutorial para programar em C, slides apresentados em sala de aula e exercícios para prática de programação. Os slides da disciplina foram construídos pelo prof. Rafael e por mim e servem apenas para revisar a discussão em sala e não formam uma referência completa do conteúdo. Para o estudo, é recomendado que @ estudante, principalmente, pratique programação resolvendo e implementando os exercícios propostos e, além disso, busque e estude exemplos de código-fonte nos capítulos correspondentes dos livros-textos. A página também inclui alguns tutoriais para instalar o ambiente, configurar um editor, utilizar git e depurar programas. Esses tutoriais foram gentilmente criados por monitores de diversos semestres.

Aulas e atendimento

As aulas expositivas com o professor ocorrerão às quartas e quintas das 19 às 21h na sala XXX. As aulas de laboratório ocorrerão às terças das 21 às 23h na sala 300 no IC 3. Nessas aulas, os monitores irão esclarecer o enunciado das tarefas e tirar outras dúvidas, além de fornecer o atendimento.

Em datas determinadas, as aulas expositivas poderão ocorrer às terças na sala de laboratório.

O atendimento e comunicação entre estudantes, professor e monitores serão feitos via [Discord](#). Cada estudante receberá um link por e-mail com o convite para se cadastrar no servidor. Você pode utilizar uma conta de Discord pessoal existente, mas deve alterar seu apelido visível no servidor da disciplina para seu nome completo e número de RA (e.g., [Fulano da Silva \(123456\)](#)). Se preferir, também pode criar uma conta nova com e-mail institucional.

É recomendável utilizar o Discord a partir do computador (utilizando o programa para desktop ou o site), mas também é possível baixar e instalar o aplicativo de celular disponível.

Todos podem escrever e ler as mensagens nas salas de atendimento e vocês são **fortemente encorajados** tanto a fazer perguntas quanto a responder dúvidas de colegas. Vocês também podem discutir quaisquer outros assuntos relacionados à disciplina que acharem interessante. O professor e os monitores acompanharão as conversas e tentarão responder sempre que possível. Pelo menos um monitor ficará disponível no chat nos horários de atendimento marcados. A tabela com os horários de atendimento estará disponível no Discord e pode sofrer alterações de acordo com a demanda de estudantes e disponibilidade dos monitores. Observações importantes:

1. Observe os horários de atendimento com atenção e siga as regras descritas no servidor do Discord.
 2. Para correção de tarefas que peçam apresentação, compareça a um atendimento (presencial ou virtual) com um monitor PED.
 3. O professor responderá dúvidas gerais durante as aulas após a parte inicial com a exposição do conteúdo. Para conversar pessoalmente e individualmente com o professor, você deve solicitar um horário ao professor via Discord com antecedência, que poderá atendê-lo ou na sala do professor (sala 12), ou via videoconferência, a depender da disponibilidade.
-