Instituto de Computação/Unicamp – Estruturas de Dados - MC202DEF
Prof. Lehilton Lelis Chaves Pedrosa

Programa

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: https://www.ic.unicamp.br/~lehilton/mc202def.

Objetivos

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

- escrever algoritmos iterativos e recursivos;
- implementar e testar programas escritos em Python ou em C.

No primeiro dia de aula, haverá uma avaliação diagnóstica. A avaliação não terá nota atribuída e servirá para verificar os requisitos. Se você não estiver presente na primeira aula, procure fazê-la e entregá-la ao professor posteriormente.

Atividades

Exercício pré-aula: Em unidades específicas, serão sugeridas leituras ou pesquisas, que deverão ser realizadas pelo aluno como pré-requisitos para as aula; essas atividades terão o formato de questões de motivação do conteúdo das aulas.

Aula expositiva: Durante as aulas, @s alun@s devem participar levantando dúvidas, sugestões, ou possivelmente resolvendo os problemas solicitados. As aulas correspondem a cerca de 20% do tempo total dedicado a disciplina; a qualquer momento @ alun@ pode tirar dúvidas com o professor. **Durante a aula expositiva não será permitido o uso de equipamentos eletrônicos (celular, notebook etc.).**

Exercícios de fixação: Após as aulas, serão propostos diversos exercícios de fixação para serem feitos em casa. O conteúdo das listas é considerado parte integrante do curso e @s alun@s devem resolvê-las para realizar as avaliações. Há tanto questões teóricas quanto práticas. Implementar os programas solicitados nas questões serve como um exercício mais simples antes do laboratório de programação. Recomenda-se tentar fazer as atividades individualmente e, depois, discutir em grupo. Como as questões são abertas, não há gabarito; as dúvidas devem ser **anotadas** e levadas a monitoria nos horários de atendimentos, ou ao professor, durante as aulas.

Exercícios da unidade (avaliação): Serão realizadas quatro aulas de exercícios, com duração de 1h a 1h30 em datas a serem divulgadas na página da disciplina. Nessas aulas, @s @lunos deverão resolver e entregar listas com os exercícios propostos em dupla, cada lista valendo de 0 a 10 pontos (E1 a E4). A dinâmica será:

1. Poderão ser consultadas notas de aula ou resolução dos exercícios propostos desde que essas sejam **anotações manuscritas e pessoais** (i.e., escritas em papel a mão pel@ própri@ alun@). Não será permitida consulta a qualquer outro material: eletrônico, anotações de terceir@s, etc.
2. Amb@s devem participar discutindo e resolvendo as questões. Não é permitido discutir com demais. **Não é permitido repetir a dupla**; alun@s que repetirem duplas terão a nota correspondente zerada no final do semestre.
3. Cada dupla receberá uma única folha de questões. Tod@s deverão preencher os campos de identificação nos locais adequados. As respostas devem ser escritas sem rasura nos quadros indicados. Respostas que ultrapassarem os quadros ou estiverem em folhas de rascunhos serão lidas a critério do corretor.

Laboratório de programação (avaliação): Haverá 12 trabalhos de programação individuais, a serem submetidos no SUSY, com prazos de uma semana (incluindo eventuais falhas do sistema) divulgados na página, valendo de 0 a 10 pontos (L1 a L12). Os critérios de correção serão especificados em cada tarefa e a nota será a soma de duas partes:

- 7 pontos proporcionais aos acertos de **casos de teste fechados** e correção do algoritmo;
- 3 pontos referentes a critérios específicos do exercício e a qualidade de código.

A dinâmica de correção será:

1. alun@ entrega o trabalho até o prazo, valendo até 10 pontos;
2. monitor divulga a correção e o feedback do trabalho;
3. após o prazo, @ alun@ que desejar, poderá ressubmeter o trabalho uma única vez até o dia **XXX** desde que tenha realizado as correções sugeridas no feedback; nesse caso, a nota será será truncada a no máximo 7 pontos.

Observação: O programa deverá ser submetido nas tarefas do SUSY de código ENTREGA-01, ENTREGA-02, etc. As **tarefas ENTREGA-XX têm limite de uma única submissão**. Antes de entregar, o programa poderá ser testado nas tarefas do SUSY de código DRAFT-01, DRAFT-02, etc., que não têm limite no número de submissões; note que os programas submetidos nas **tarefas DRAFT-XX não serão corrigidas**.

Avaliação

Média: Serão calculadas as médias aritméticas de exercícios de unidade (E) e de laboratórios de programação (L). Depois, será calculada a média do semestre como M := (E+L)/2. Será aprovad@ com nota de aproveitamento A := M @ alun@ que satisfizer todos os critérios abaixo:

1. 75% de frequência;
2. E ≥ 6; L ≥ 6;
3. nota maior ou igual a 6 em pelo menos dois laboratórios de cada um dos grupos a seguir: {L1, L2, L3}, {L4, L5, L6}, {L7, L8, L9} e {L10, L11, L12}.

Do contrário, a nota de aproveitamento do semestre será A := *mínimo {4, M}*.

Observação: A frequência será controlada por listas de presença ou pela participação nas avaliações. É obrigatória a presença nas aulas teóricas e de laboratório. Não poderá assinar a lista @ alun@ que chegar com 20 min de atraso ou sair com 20 min de antecedência; em caso de atestado, envie uma cópia por e-mail ao professor no prazo de 15 dias e mantenha o documento original. Alun@s que, justificadamente, não puderem comparecer ao menos a 75% das aulas devem conversar com o professor no início do semestre. **Importante:** Em qualquer momento, se tiver dúvidas ou tiver dificuldade em cumprir os critérios, procure o professor (o quanto antes!).

Exame: Poderá realizar exame, com nota E, entre 0 e 10, no dia 11/7/2019, @ alun@ com 75% de frequência e nota A ≥ 2,5. Se @ alun@ fizer o exame, a nota final será *mínimo {5, (A+E)/2}*, senão a nota final será *A*.

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, mostrar ou distribuir laboratório de programação, cola ou consulta a material proibido etc.), os envolvidos serão reprovados com nota 0.

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. Tenembaum. 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 http://visualgo.net/.

Material didático

Será disponibilizado na página da disciplina material de apoio para estudo, que inclui um tutorial para programar em C, uma máquina virtual com um ambiente de desenvolvimento completo e configurado, slides apresentados em sala de aula e exercícios para prática de programação. Parte do material foi construído e cedido gentilmente pelo prof. Rafael. Os slides servem apenas para revisar a discussão em sala e não formam uma referência completa do conteúdo. Para o estudo, é recomendado que @ alun@, 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.

Observação: Nos trabalhos de programação, podem ser copiados trechos de códigos disponibilizados nos slides. Revise-os atentamente e observe que esses trechos têm função didática, então são incompletos e podem conter erros. Cópia de trechos de código de qualquer outro material sem autorização expressa do professor será considerada fraude.

Rotina de estudo

Importante: Note que não há provas final e de meio de semestre. Portanto é fundamental criar uma rotina de estudos contínua. Algumas sugestões são úteis para o bom desenvolvimento da disciplina:

1. Faça os exercícios de motivação antes da aula correspondente e utilize a aula confirmar ou sanar as dúvidas. Participe das aulas ativamente e não deixe passar qualquer dúvida; toda pergunta é relevante!
2. Planeje-se e separe algumas horas em alguns dias por semana para resolver e **implementar os exercícios sugeridos**, mesmo os não obrigatórios. Não é obrigatório resolver todos os exercícios de fixação, mas é fundamental praticar programação constantemente e disciplinadamente. @s alun@s também são encorajados a se reunir e estudar em grupo (sempre depois de tentar resolver os exercícios individualmente).
3. Faça primeiro os exercícios mais simples e depois os mais desafiadores. Em média, deve-se investir 10min por cada item. Não gaste todo o tempo em um conjunto pequeno de problemas: anote as principais dificuldades, discuta com colegas e leve-as ao monitor no horário de atendimento.
4. Veja o resultado das avaliações! Elas servem para que você identifique os problemas (dificuldades com entendimento do problema, algoritmo, sintaxe ou legibilidade). Tente refazer as tarefas e, se não puder identificar o que está errado ou não conseguir corrigir o problema, anote a dúvida e leve-a ao monitor ou ao professor (mas evite pedir aumento de nota, ou comparar notas de colegas).

Horário e local

As aulas teóricas serão ministradas das 21 às 23h na sala XXX às terças-feiras e das 19 às 21h na sala XXX às quintas-feiras. As práticas de laboratório serão ministradas nas salas XXX e XXX às sextas-feiras das 21 às 23h.

Atendimento

Lista de e-mails:
Haverá uma lista de e-mails para discussão das turmas EF e GH. Os alunos podem usar a lista para tirar dúvidas sobre o conteúdo e sobre a disciplina, ou para compartilhar conteúdo relevante a estrutura de dados. Nessa lista **não** é permitido enviar ou indicar soluções para os exercícios de avaliação (i.e., não envie o código ou trechos de código de seu laboratório mesmo que em formato de pseudocódigo, etc.), mas é permitido discutir e tirar dúvidas sobre o enunciado.

Monitoria:
@s alun@s são fortemente encorajados a procurar monitoria, tirando dúvidas sobre o conteúdo e exercícios. Haverá atendimento de monitoria na sala XXX, nos horários abaixo.

Monitores:	Quinta-feira	Sexta-feira
13h às 14h	XXX	XXX
18h às 19h	XXX	XXX

Além disso, se não for possível sanar as dúvidas no horário de atendimento, @s alun@s podem contatar os monitores por e-mail monitores-mc202@googlegroups.com. Sempre que for compartilhar trechos de código do laboratório, certifique-se de que apenas os monitores recebam o código.

Organização do curso

O curso tem quatro unidades. Na unidade i, será feita uma revisão de conceitos básicos juntamente com uma introdução à linguagem de programação C. Cada unidade será finalizada com uma avaliação teórica dos assuntos da unidade. A ordem dos assuntos abaixo é uma tentativa e serve para que @ alun@ se planeje e estude para as aulas. Ordem e conteúdo podem variar a depender do andamento da turma. **Sempre consulte também os avisos na página da disciplina!**

i – Conceitos básicos e linguagem de programação C
- Avaliação Diagnóstica
- Vetores e entrada e saída
- Matrizes
- Strings e operadores lógicos
- Registros e tipo abstrato de dados
- Ponteiros e alocação dinâmica
- Passagem por referência
- Recursão
- Análise de Algoritmos
- Exercícios de unidade

ii – Conjuntos dinâmicos
- Listas Ligadas
- Variações de Lista
- Pilha e Fila
- Aplicações de Pilha
- Árvores Binárias
- Árvores de Busca
- Árvores Balanceadas
- Hashing
- Exercícios de unidade

iii – Técnicas de programação
- Backtracking
- Fila de Prioridade e Heap binário
- Ordenação e Heapsort
- Mergesort e Quicksort
- Exercícios de unidade

iv – Estruturas de dados avançadas
- Grafos - Representação
- Grafos - Percursos
- Grafos - Algoritmos
- Árvores B
- Escolhendo uma estrutura de dados
- Exercícios de unidade