Planos de Aula e Critérios de Avaliação

EA876 - 1o. Semestre de 2020  
Tiago Fernandes Tavares

[Contato](#_91dx8zoilt4p)

[Objetivos do Curso](#_2pzsbkqwzs77)

[Aulas](#_2xlz1ke6euq8)

[Atividades propostas](#_bswbmkj2g1iv)

[Exercícios computacionais](#_4u6dpsqle0qt)

[Trabalhos em dupla](#_1kcv9bcfw6pg)

[Testes em aula](#_4b9eumxbhhow)

[Prova final](#_hln5k8ug3oml)

[Outras atividades](#_5p4k6c3pvd1o)

[Quanto ao plágio](#_7p6d6a8zge8y)

[Notas e critério de aprovação](#_r1hbsqqaesd2)

[Bibliografia](#_dznlo6zk0i4)

# Contato

* Para encontrar o **Tiago**, você pode tentar bater diretamente na porta da sala 311 (bloco A, terceiro andar), ou procurar pelos corredores da FEEC ou nos laboratórios (DSPCom, no bloco G2, ou no NICS, no prédio da Reitoria 3). A maneira mais eficaz, porém, é mandar um e-mail ([tavares@dca.fee.unicamp.br](mailto:tavares@dca.fee.unicamp.br)) explicando cuidadosamente sua dúvida ou seu problema, que será respondido com calma e tranquilidade, geralmente em menos de 48h.
* Para encontrar o **Lucas**, você pode enviar um e-mail ( lucasbertoloto@outlook.com ) ou buscar diretamente os **horários de monitoria** (Segunda-feira e Sexta-feira, das 12:30 às 13:55). Pode encontrá-lo também pelos corredores da FEEC.
* O material de aula está todo linkado do Google Classroom da disciplina. Acesse pelo site: <http://www.ggte.unicamp.br>, e então navegue para o ambiente de aprendizagem Google Classroom. Talvez você tenha que usar o código **2xu62vz.**

# Objetivos do Curso

Ao fim deste curso, os alunos serão capazes de:

* Resolver problemas de busca por padrões em grandes documentos de texto;
* Entender a importância da pilha na execução de programas e no processo de compilação;
* Resolver problemas de paralelização de programas usando programação multi-processo e multi-thread;
* Programar (*develop*) e entregar (*deploy*) programas em C usando Makefile, testes tipo entrada-saída e comentários adequados;
* Redigir textos científicos curtos, corretos e efetivos que demonstram a solução de problemas computacionais.

# Aulas

* Sala FE-11, quartas e sextas das 14:00 às 16:00.
* Evitar atrasos!
* Trazer o material que precisa para acompanhar as aulas.
* Ao sair da sala, deixamos preparada para as próximas pessoas que forem usar!

# Atividades propostas

Neste curso, teremos diversas atividades. Algumas deles serão avaliadas com nota e terão peso na nota final do aluno, como segue:

## Exercícios computacionais

Tratam-se de exercícios de programação em C com o objetivo de exercitar os conceitos vistos nas aulas teóricas. Todos os exercícios foram planejados para serem feitos em menos de 4 horas. Os exercícios computacionais devem ser entregues através do GitHub. Para isso, siga as instruções que estão na folha específica.

**Quanto ao prazo:** o prazo para entrega do trabalho é, irrevogavelmente, às 23h59 do dia marcado como prazo final. Não serão aceitos trabalhos com atraso. Porém, todas as tarefas já estão disponíveis desde o primeiro dia de aula, então podem ser feitas com adiantamento. Além disso, todos os testes são abertos, então não deveria haver nenhuma surpresa na solução dos exercícios.

**Quanto às notas:** as notas serão dadas, em princípio, proporcionais ao número de testes que a solução enviada passar. Ou seja, se existirem 5 testes e o programa passou em 3 deles, então a nota daquele programa será (3/5) = 6. Além disso, os programas serão vistoriados manualmente para analisar requisitos não-funcionais (por exemplo: verificar se a divisão de trabalho entre as diversas threads do programa está correta). Também, não serão aceitas soluções do tipo “if (entrada) then (saída)”, ou seja, que são talhadas a funcionar somente com os testes que forem passados.

## Trabalhos em dupla

Este curso tem essencialmente duas metades: uma tradicionalmente chamada de *compiladores* e outra chamada de *sistemas operacionais*. Embora não consigamos chegar a fazer um compilador nem um sistema operacional, vamos aprender técnicas que são bastante relacionadas a esses dois temas. Ao fim de cada etapa, faremos um trabalho que consiste em resolver um problema computacional cuja resposta não é fechada e, após, analisar os resultados encontrados. Trata-se de um trabalho mais analítico que os exercícios computacionais. Os trabalhos serão entregues em um arquivo .zip contendo tanto o relatório quanto o código correspondente, incluindo instruções para compilar e executar o programa.

**Quanto ao prazo:** o prazo para entrega do trabalho (texto e código) é, irrevogavelmente, às 7h30 da manhã do dia marcado como prazo final. Não serão aceitos trabalhos com atraso, nem textos sem código computacional acompanhando, nem códigos computacionais sem texto relacionado. O enunciado do trabalho está disponível desde o primeiro dia de aula, e será melhor discutido quando faltar por volta de um mês para a entrega final.

**Quanto a notas:** o trabalho será avaliado pelo relatório apresentado e pelo programa, que deverá permitir replicar os resultados comentados no trabalho.

## Testes em aula

Durante algumas das aulas (como está marcado no calendário!), teremos testes curtos. O teste deve ser feito *individualmente*. A duração do teste deverá ser de aproximadamente 5 minutos. O teste será corrigido binariamente (acertou/errou). Os testes que estiverem errados serão devolvidos na aula seguinte, e a pessoa testada poderá fazê-lo novamente, valendo 50% da nota total (se for devolvido corretamente) ou 75% (se for devolvido correto e, adicionalmente, apontando o erro que foi cometido anteriormente).

**Quanto ao prazo:** um teste com erro deverá ser devolvido sempre na aula seguinte à devolução. Ou seja: se um teste for feito na quarta, e estiver errado, ele será devolvido à pessoa testada na sexta; a pessoa testada, por sua vez, deverá devolver o teste até a quarta feira seguinte. A não-devolução do teste com erro implica em assumir que a pessoa testada avaliou que o trabalho investido no teste não vale a nota relacionada.

**Quanto a notas:** todos os testes receberão nota 0, 0,5, 0,75 ou 1, dependendo do *status* do teste.

## Prova final

No último dia de aula, teremos uma prova final (individual, escrita e em horário de aula). Trata-se de uma prova conceitual, que irá avaliar conceitos (ou seja, evitando perguntas que envolvem fazer contas ou lembrar de especificidades do curso). Deve ser uma prova que poderia ser feita com tranquilidade por qualquer aluno do curso.

## Outras atividades

A qualquer momento do curso, qualquer participante pode realizar projetos extras, em assuntos que sejam de seu interesse mas que, porventura, não tenham sido cobertos pelas atividades inicialmente propostas. Se esse for o seu caso, você pode discutir com o professor sobre como seria esse projeto (objetivos, requisitos, forma de entrega, etc.), de forma que ele possa contar para aprendizado extra.

## Quanto ao plágio

Esse é um assunto muito chato, sobre o qual precisamos conversar claramente. O motivo da existência dessa grande variedade de atividades é permitir exercitar diversas habilidades pessoais e diversos aspectos do conteúdo (teoria, prática, análise, etc.). O objetivo não é, de forma alguma, tornar a vida do aluno mais difícil, e sim fornecer *feedback* quanto ao seu próprio aprendizado. Plágio, nestas atividades, significa fraudar as condições propostas para a realização das atividades, por exemplo (mas não restringindo-nos a): copiando o relatório de outra dupla e trocando o nome, copiando o programa de um colega e trocar o nome das variáveis, fazer em grupo atividades propostas para serem individuais, buscar brechas/loopholes em enunciados ou outros documentos da disciplina para obter vantagens indevidas, etc. Uma situação bastante chata que a FEEC vem passando é que muitos casos de plágio vêm sendo reportados, e isso é um grande problema, porque o plágio generalizado causa distúrbio na confiabilidade do processo de ensino, tanto no sentido de garantir que todos os alunos aprendam assuntos relevantes quanto no sentido de o diploma ser um certificado deste aprendizado. Por isso, precisamos ter ciência de que casos de plágio em todos os documentos oficiais (em código, texto, provas, testes, listas de presença, etc.) implicarão em, *no mínimo*: (a) nota zero na disciplina e (b) advertência enviada à DAC.

# Notas e critério de aprovação

As notas serão divididas da seguinte forma:

C - Média das notas em exercícios **C**omputacionais, excluindo o de menor nota

G - Média das notas dos trabalhos em **G**rupo

T - Média das notas dos **T**estes feitos em aula, excluindo o de menor nota

P - Nota na **P**rova final

A - Notas em atividades **E**xtras

A **N**ota final será calculada por:

N = (4C + 4G + T + P + 2A) / 10

Um aluno não fica de exame se, simultaneamente:

* N >= 5,0
* As notas de todos os trabalhos computacionais, excluindo o de menor nota, forem, individualmente, maiores ou iguais a 3
* As notas dos dois trabalhos em grupo forem, cada uma, maiores ou iguais a 3

Caso o aluno fique de exame, deverá vir fazer o exame (que é uma prova individual, escrita e no horário de aula, sobre a matéria do semestre todo). Sua **M**édia final será dada por:

M = (N + E) / 2, onde E é a nota do **E**xame. **Conforme exigência do MEC, a presença em 75% das aulas é necessária para aprovação.**

# Bibliografia

Tudo o que abordaremos no curso serão problemas bastante conhecidos na computação, então buscas no Google serão sempre bem-vindas. Não seguiremos nenhum livro em especial, mas alguns deles podem ser úteis para consulta:

* Ricarte, I., ``Introdução à Compilação''. Ed. Elsevier.
* Tannenbaum, A. S., ``Sistemas Operacionais Modernos''. Ed. Pearson.
* Jargas, A. M, ``Expressões Regulares -- Guia de Consulta Rápida'' (<http://aurelio.net/regex/guia/>)
* Niemann, T, ``Lex and Yacc tutorial '' (<http://epaperpress.com/lexandyacc/>)

Porém, a maior parte do material do curso (slides, códigos, e, mais importante: o link para os exercícios computacionais) será disponibilizada no GitHub:

[**https://github.com/EAxxx/EA876**](https://github.com/EAxxx/EA876)