

Fundamentos de Tecnologia da Computação

— *Syllabus* —

Prof. Abrantes Araújo Silva Filho

2024/2

LEIA COM ATENÇÃO!

Além do material oficial que está disponível no [Portal do Aluno da UVV](#)¹ (Universidade Vila Velha) e no [Portal da Disciplina](#)², este documento contém mais detalhes sobre a dinâmica da disciplina **Fundamentos de Tecnologia da Computação** (“Fundamentos”) para as turmas do semestre 2024/2. Leia-o com atenção e procure o professor para esclarecer qualquer dúvida. Este *syllabus*³ funcionará como a nossa “Constituição”, ou seja, será a base para todas as regras, políticas e funcionamento de nosso curso. Além do *syllabus* você também pode consultar a [ementa oficial da disciplina](#)⁴.

Sumário

1	Visão geral da disciplina	3
1.1	O que você aprenderá?	3
1.2	Origem desta disciplina	4
2	Pré-requisitos	5
3	Objetivos do curso e resultados esperados	6
4	Estrutura e recursos da disciplina	7
4.1	Professor	7
4.2	Monitores	8
4.3	Laboratório de computação e computador pessoal	8
4.4	Grau de dificuldade e esforço estimado	9
4.5	Datas e horários das aulas	11
4.6	Ambientes virtuais utilizados	11
4.7	Materiais necessários	11

¹<https://aluno.uvv.br>

²https://disciplinas.uvv.br/disciplinas/fundamentos_computacao

³Ementa, plano detalhado de ensino, programa de estudos.

⁴https://disciplinas.uvv.br/assets/disciplinas/fundcomp/2024_2/ementa_2024_2.pdf

5	Dinâmica da disciplina	11
5.1	Aulas	12
5.2	Monitorias	12
5.3	Tutorias	13
5.4	Laboratórios de programação	13
5.5	Diários de aprendizagem	13
5.6	Atividades no Autolab	14
5.7	<i>Problem Sets</i> (PSETs)	14
6	O que esperamos de você?	15
7	Avaliação da aprendizagem e notas	16
7.1	Peso das atividades nas notas bimestrais	16
7.2	Arredondamento de notas	16
7.3	Entregas em atraso	17
8	Integridade acadêmica	17
8.1	Política sobre trabalho colaborativo	19
8.2	Cláusula de arrependimento	20
9	Perguntas freqüentes	20

1 Visão geral da disciplina

Olá, seja muito bem-vindo!

Esta é a disciplina **Fundamentos de Tecnologia da Computação** (“Fundamentos”), uma **introdução aos desafios intelectuais da computação e da arte da programação**, para alunos da área de tecnologia (Ciência da Computação, Sistemas de Informação, Análise e Desenvolvimento de Sistemas, Engenharia da Computação) ou não (Medicina, Psicologia, Arquitetura, Direito e outros). Na verdade, qualquer pessoa interessada, de qualquer outra área ou curso, com ou sem experiência anterior em computação e/ou programação, tem condições de fazer a disciplina e aprender.

Não se preocupe se você não tiver nenhuma experiência prévia na área: estatísticas globais (Universidades de Harvard e Yale) mostram que 2/3 dos estudantes em disciplinas introdutórias como esta não tinham nenhuma experiência prévia na área. Se você souber utilizar um computador para atividades cotidianas, como mandar um e-mail, escrever um documento ou navegar por sites na Internet, já tem condições para fazer a disciplina.

Nota

No semestre 2024/2 esta disciplina está sendo ofertada para os cursos de [Ciência da Computação](#)⁵ e [Sistemas de Informação](#)⁶ da [Universidade de Vila Velha \(UVV\)](#)⁷. Se você é de algum outro curso da UVV e gostaria de fazer esta disciplina, entre em contato com o Professor Abrantes Araújo Silva Filho (abrantesasf@uvv.br) para verificar a possibilidade de participação.

1.1 O que você aprenderá?

Este curso ensinará como **projetar e implementar soluções para resolver problemas**, com ou sem programação, com ênfase em corretude, projeto e estilo. Pretendemos que você tenha uma **visão geral da computação** como está hoje e como será no futuro, e aprenda os conceitos fundamentais da computação (e resolução de problemas!), tais como:

- Pensamento computacional
- Abstração
- Algoritmos
- Estruturas de dados
- Bancos de dados
- Programação web
- Idéias fundamentais da computação
- ... e muito mais!

Esta disciplina será fundamental para que você aprenda a pensar **criticamente**, a pensar mais **metodicamente**, a pensar mais **computacionalmente**. Com o tempo você verá que computação não é sobre computadores ou sobre programação, é sobre conhecimento, problemas e sobre como resolver esses problemas de forma metódica, correta, algorítmica.

⁵<https://uvv.br/ensino-presencial/graduacao/ciencia-da-computacao>

⁶<https://uvv.br/ensino-presencial/graduacao/sistemas-de-informacao>

⁷<https://uvv.br>

Você trabalhará com diversos **Problem Sets** (PSETs)⁸ inspirados pelas áreas artísticas, humanas, ciências sociais e naturais, além das áreas STEM (*Science, Technology, Engineering e Mathematics*) e, mais do que ensinar você a programar em uma linguagem de programação específica (usaremos várias!), este curso vai ensinar os **fundamentos da computação** e os **fundamentos da programação** de forma que você mesmo consiga aprender por conta própria as linguagens do futuro.

Nós começaremos com o **Snap!**⁹ e o **Scratch**¹⁰, linguagens visuais que utilizam peças de quebra-cabeça para escrever os programas, e que nos permitem explorar os primeiros conceitos fundamentais da computação e da programação.

Depois iremos para uma linguagem mais tradicional, onipresente, chamada **Linguagem C**¹¹, que foi uma precursora e base para diversas linguagens de programação mais modernas. Utilizando a linguagem C você aprenderá não só sobre variáveis, funções, estruturas de controle, estruturas de repetição e outras coisas mas, também, aprenderá sobre como os computadores funcionam por “baixo do capô”: processador, memória e tudo mais. Você perceberá que a linguagem C, sendo mais antiga, não tem tantas funcionalidades prontas como as linguagens mais modernas, como a Snap! ou Scratch.

Faremos então a transição para uma linguagem mais moderna, a **Python**¹², uma linguagem de alto nível que você entenderá com muita facilidade devido à base que você construiu com a linguagem C. Ao final do semestre você aprenderá a **SQL**¹³, para armazenar dados em bancos de dados.

Dependendo do cronograma, do ritmo de estudo, do comprometimento dos alunos e do grau de aprendizagem, talvez seja possível abordar tópicos extras adicionais:

- **HTML, CSS e JavaScript;**
- *Frameworks* web como o **Flask;**
- **Cibersegurança;** e
- Juntar todo o aprendizado em um **projeto final** a ser exibido para a UVV!

Prepare-se, pois esta disciplina é **extensa, árdua e difícil**, mas o resultado, se você se dedicar conforme descrito neste plano de ensino, será recompensador!

1.2 Origem desta disciplina

O conteúdo da disciplina é uma adaptação do curso “**CR6.100B: Introdução à Ciência da Computação**”¹⁴, do **Computação Raiz**¹⁵, que, por sua vez, é uma tradução e adaptação da disciplina “**CS50: Introduction to Computer Science**”¹⁶, da Universidade de Harvard, acrescida de idéias e conceitos de outras disciplinas (**BJC**¹⁷ e **CS10**¹⁸, da Universidade da Califórnia em Berkeley; **6.001**¹⁹ e **6.0001**²⁰ do MIT). Para deixar mais claro:

⁸PSET: é uma abreviatura para *Problem Set*, Conjunto de Problemas. Os PSETs são problemas altamente difíceis, que forçarão seu estudo e capacidade de resolução ao máximo.

⁹<https://snap.berkeley.edu>

¹⁰<https://scratch.mit.edu>

¹¹<https://cmprz.me/c>

¹²<https://www.python.org>

¹³<https://cmprz.me/wikisql>

¹⁴<https://www.computacaoraiz.com.br/cr6100b>

¹⁵<https://www.youtube.com/computacaoraiz>

¹⁶<https://cs50.harvard.edu/x>

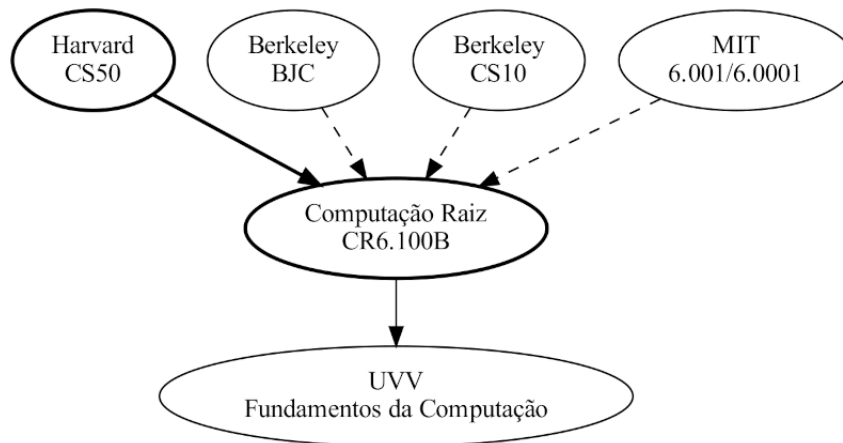
¹⁷<https://bjc.edc.org>

¹⁸<https://cs10.org>

¹⁹<https://cmprz.me/sicpocw>

²⁰<https://cmprz.me/mit60001>

Figura 1: Origens desta disciplina



A principal diferença entre a **CR6.100B** e a **Fundamentos de Tecnologia da Computação** é em relação ao tempo. Como a CR6.100B é uma disciplina online, tem um cronograma mais “lento”, no ritmo do próprio aluno, que pode durar vários e vários meses; já esta disciplina de Fundamentos deve se adequar ao calendário acadêmico da UVV e, por isso, nem todo o conteúdo da CR6.100B será visto: a Fundamentos da Computação faz uma seleção de conteúdos mais importantes para os alunos.

Conforme vimos, dependendo do cronograma, do ritmo de estudo, do comprometimento dos alunos e do grau de aprendizagem, a Fundamentos de Tecnologia da Computação pode conseguir cumprir uma maior ou menor quantidade de conteúdo da CR6.100B. Mas isso depende muito mais dos alunos do que da disciplina em si: se os alunos forem comprometidos, participarem de todas as atividades e estudarem com afinco, mais conteúdos podem ser vistos. O núcleo central do conteúdo (dos fundamentos até estruturas de dados) é sempre estudado, mas os conteúdos adicionais dependem do ritmo dos alunos.

Importante

Apesar da Fundamentos não ver todo o conteúdo da CR6.100B, não se preocupe: o conteúdo principal e essencial para uma disciplina introdutória como essa será totalmente visto e aprendido.

Recomenda-se que você, por conta própria, siga o programa de estudos da CR6.100B, paralelamente ou após esta disciplina de Fundamentos.

2 Pré-requisitos

🔒 Obrigatórios²¹:

- **Matemática:** familiaridade e conforto com, pelo menos, a matemática do ensino médio (aritmética, álgebra, funções e conjuntos). Caso sua base matemática seja fraca, veja o texto “[Preparação para graduação em áreas exatas: matemática além do básico](#)”²², com dicas para estudar novamente a matemática, do jeito certo; e

²¹São conhecimentos prévios e/ou recursos que você, obrigatoriamente, deve ter para que possa acompanhar adequadamente a disciplina e aprender o conteúdo. Caso você não possua algum dos pré-requisitos obrigatórios poderá ter dificuldade para acompanhar as aulas ou realizar todas as atividades propostas pelo professor (nesse caso converse **imediatamente** com o professor para explicar a situação e tentar encontrar uma solução).

²²<https://cmprz.me/estmat>

- **Informática básica:** entendimento sobre a estrutura de diretórios, pastas e arquivos do computador, utilização de *softwares* básicos como processadores de texto, planilhas eletrônicas e *email*, compactação e descompactação de arquivos, criação de arquivos em texto puro e em formatos como PDF, uso básico da *Internet*.

🔒 **Recomendados**²³:

- **Computador:** você deve ter um computador com acesso a Internet (câmera, microfone e caixas de som são recomendados) e com capacidade suficiente para instalar e executar os softwares que serão utilizados (um computador recente, com arquitetura Intel x86 de 64 bits, com Linux, Windows ou Mac, com 8 GiB de memória RAM e bastante espaço em disco). Verifique a Seção 4.3 (**Laboratório de computação e computador pessoal**) para maiores detalhes. Caso você não tenha um computador e/ou notebook, poderá utilizar os laboratórios de informática da UVV, no período vespertino, para estudar e realizar as atividades da disciplina; e
- **Inglês:** os livros, materiais e *softwares* utilizados na disciplina estarão em português ou em inglês (haverá bastante conteúdo em inglês, principalmente leitura de documentação técnica). Na área da computação é absolutamente fundamental o domínio de, pelo menos, a leitura de textos em inglês. **É sua obrigação** estudar e aprender inglês.

✓ **Não obrigatórios**²⁴:

- **Programação:** algum curso introdutório de programação, em qualquer linguagem.

3 Objetivos do curso e resultados esperados

- Entender o que é computação, ciência da computação e programação;
- Entender os componentes do pensamento computacional (decomposição, reconhecimento de padrões, abstração, algoritmos e estruturas de dados), aplicando-os na resolução de diversos problemas importantes;
- Entender como a estrutura de um computador (processador e memória) influenciam nos programas que você criar para resolver problemas;
- Pensar mais metodicamente;
- Representar e processar informação;
- Se comunicar de forma sucinta e precisa;
- Resolver problemas de modo correto e eficaz;
- Utilizar técnicas de gerenciamento de complexidade ao resolver problemas;
- Decompor um problema complexo em partes menores e achar a solução para essas partes;
- Reconhecer padrões entre os problemas;
- Operar em múltiplos níveis de abstração;

²³São conhecimentos e/ou recursos que facilitarão muito o seu estudo. Caso você não possua os requisitos recomendados, sugere-se que você faça um plano para obtê-los com o tempo.

²⁴São conhecimentos prévios e/ou recursos não obrigatórios mas que, se você tiver, podem facilitar seu estudo (você já terá uma base a partir da qual irá desenvolver novas habilidades). Se você não tiver nenhum dos requisitos recomendados, não se preocupe: você conseguirá acompanhar a disciplina.

- Programar imperativamente, com o uso de subprogramas;
- Separar os detalhes de projeto (design) dos detalhes de implementação;
- Inferir, a partir dos princípios, como os sistemas funcionam;
- Avaliar a corretude, design e estilo do código de um programa;
- Entender o básico sobre complexidade de algoritmos, e reconhecer alguns dos algoritmos clássicos da computação;
- Entender o básico sobre estruturas de dados, e reconhecer algumas das estruturas de dados clássicas da computação;
- Aprender por conta própria outras linguagens de programação;
- Ler documentação técnica e tirar conclusões a partir das especificações;
- Testar as soluções criadas, encontrando falhas e identificando casos extremos que o programa trata, ou não, com corretude;
- Identificar ameaças à segurança e privacidade;
- Descrever precisamente os sinais e sintomas de problemas, perguntando questões de modo claro e objetivo;
- Perguntar questões de modo claro;
- Utilizar o terminal Linux, especificamente a interface de linha de comando, para interagir com o sistema operacional;
- Identificar e quantificar *trade-offs* entre recursos, particularmente tempo e espaço;
- Utilizar um ambiente de desenvolvimento integrado; e
- Sentir-se seguro e confiante na resolução de problemas de modo computacional.

Importante

Em última análise, o curso pretende fornecer uma base sólida para estudos adicionais em ciência da computação (e áreas afins) e capacitá-lo a aplicar a computação a problemas em outros domínios.

4 Estrutura e recursos da disciplina

4.1 Professor

Meu nome é **Abrantes Araújo Silva Filho** e serei o professor responsável por esta disciplina. Sou graduado em **medicina** pela **UFES** (1999), mestre em **epidemiologia e métodos quantitativos em saúde** pela **FIOCRUZ** (2002) e graduado em **ciência da computação** pela **FAESA** (2021). Informações de contato:

 www.linkedin.com/in/abrantes-filho

 github.com/abrantesasf, github.com/computacaoraiz, github.com/uvv-computacao

 www.abrantes.pro.br, www.computacaoraiz.com.br

 abrantesasf@uvv.br

 **Canal Computação Raiz** em <https://www.youtube.com/computacaoraiz>

 (27) 9-9991-4393, para entrar em contato via **Signal**²⁵ (eu não uso WhatsApp!)

²⁵Eu não uso o WhatsApp ou o Telegram por motivos de privacidade. Se quiser me mandar uma mensagem, você terá que usar o Signal: <https://signal.org>

Atenção

Eu recebo muitos e-mails e muitas mensagens via Signal por dia e, por isso, é normal que eu não consiga responder prontamente sua mensagem. Dependendo do período acadêmico, um atraso de alguns até vários dias é normal. Se sua mensagem é para tirar dúvidas sobre a disciplina ou a matéria, você pode entrar em contato também com os **monitores**²⁶. Se você precisa realmente falar comigo de modo urgente, o mais fácil é você **me procurar na UVV e conversar pessoalmente**.

4.2 Monitores

A disciplina contará com a participação de alguns monitores para auxiliar o estudo teórico e orientar os alunos nos laboratórios de programação. Você pode conhecer os monitores deste semestre visitando a página com as **informações sobre os monitores**²⁷.

Os monitores são alunos que já fizeram (e foram aprovados com louvor) na disciplina, e que estão **voluntariamente** dedicando tempo e esforço para ajudar os novos alunos. Por favor entenda que os monitores, por mais capacitados que estejam, podem não saber alguns aspectos mais avançados da matéria. Contamos com sua compreensão: os monitores são alunos, assim como vocês. O papel dos monitores é variado e inclui, entre outros:

- Controlar a frequência dos alunos nas monitorias e nos laboratórios de programação;
- Dar aulas de revisão do conteúdo visto com o professor;
- Dar aulas com conteúdo extra importante;
- Ajudar nos exercícios teóricos e atividades de programação a serem realizados pelos alunos;
- Receber, conferir e visar os diários de aprendizagem;
- Avaliar os códigos dos programas dos alunos, quanto a correção, estilo e design;
- Fornecer feedback nas atividades teórica e de programação;
- Utilizar ferramentas de detecção de cola ou plágio nas atividades de programação; e
- Auxiliar o professor no cálculo das notas dos alunos.

Os alunos terão até 15 horas semanais de atividades orientadas pelos monitores, incluindo monitorias teóricas e laboratórios de programação.










4.3 Laboratório de computação e computador pessoal

A estrutura de laboratórios de computação da UVV é excelente, possuindo máquinas com *hardware* e *softwares* padronizados que facilitam a realização das atividades. Recomendamos que você faça as tarefas de programação durante o período vespertino, no Laboratório 12 (que fica reservado no período vespertino para nossa disciplina). Ao utilizar o laboratório você terá ajuda direta dos monitores e de seus próprios colegas de turma.

Entretanto, por limitações de tempo ou outro problema, pode ser necessário que você utilize seu próprio computador pessoal. Veja aqui as configurações mínimas necessárias para que você consiga realizar todas as tarefas em seu próprio computador:

²⁶https://disciplinas.uvv.br/disciplinas/fundamentos_computacao/pessoal/#monitores



²⁷Ver a nota anterior.

-  **Processador:** qualquer computador relativamente recente, com processador Intel ou AMD, preferencialmente com arquitetura Intel x86 de 64 bits. Outras arquiteturas, como a ARM (MacBook Air M1, M2 ou M3, por exemplo), podem não funcionar em alguns raros casos específicos (entre em contato com os monitores ou o professor em caso de dúvidas);
-  **Memória:** recomenda-se pelo menos 8 GiB de memória RAM (16 GiB de RAM é desejável);
-  **Disco rígido:** discos SSD com, no mínimo, 240 GiB, são altamente recomendados. Nesta disciplina temos duas opções de ferramentas de programação: **ferramentas online**²⁸ ou o uso de uma **máquina virtual Linux**²⁹. Recomendamos o uso das ferramentas online de programação, que já estão configuradas e prontas e, nesse caso, você não precisa de nenhum espaço adicional no disco. Se você é mais avançado ou tem curiosidade de aprender Linux, pode optar pelo uso da máquina virtual e, nesse caso, seu disco ter pelo menos 50 GiB de espaço livre (70 GiB durante a instalação);
-  **Sistema operacional:** Linux, Windows ou Mac (Linux é altamente recomendado, e é o sistema operacional que você utilizará se optar pelas ferramentas online de programação ou pelo uso da máquina virtual);
-  **Monitor:** no mínimo um monitor HD (Full HD é recomendável). Seu monitor deve ser grande o suficiente para que você possa trabalhar com conforto. Se você tiver dois monitores poderá realizar as tarefas em um monitor e usar o outro para ler o conteúdo ou consultar a documentação técnica;
-  **Internet:** você deve ter uma conexão razoavelmente rápida à Internet para acessar o ambiente online e enviar as tarefas de programação;
-  **Som:** você precisará de caixas de som ou *headphones* para assistir aos vídeos indicados pelo professor;
-  **Microfone:** não é obrigatório mas pode ser necessário caso alguma aula extra seja feita de modo online;
-  **Câmera:** não é obrigatória mas uma *webcam* pode ser utilizada caso alguma aula extra seja feita de modo online.

Se você tiver um *notebook* pode utilizá-lo durante as aulas com os monitores no laboratório de programação. A vantagem de usar seu próprio *notebook* é que você terá o mesmo ambiente, tanto na UVV quanto em casa, e poderá seguir os exercícios e tarefas de modo contínuo. A desvantagem de usar seu próprio *notebook* é que você terá que andar com ele pela cidade, correndo o risco de algum dano acidental (queda) ou de ser furtado.

4.4 Grau de dificuldade e esforço estimado

Tenha sempre em mente que esta disciplina é extensa e precisa de **dedicação, estudo e leitura diária** para que você consiga aprender tudo o que é necessário. Em especial atente-se para o seguinte:

-  **Nível:** as primeiras três ou quatro semanas são de nível introdutório; depois teremos várias semanas de nível intermediário até intermediário/avançado; as semanas finais são de nível avançado;
-  **Língua:** as aulas, livros e materiais da disciplina estarão em **português** e/ou em **inglês**. Os *softwares* utilizados estão, em sua maioria, em inglês (alguns oferecem tradução para português, mas não conte com isso: **é sua obrigação aprender inglês**);

²⁸<https://cs50.dev>

²⁹<https://cmprz.me/lingprog4>

- 🗓️ **Esforço total:** 10–15 horas de estudo por semana, **além da carga horária das aulas presenciais**, dependendo de seu conhecimento e habilidades prévias. Prepare-se para dedicar, em média, 2–3 horas diárias para leitura e realização das atividades programadas;
- 📖 **Leitura:** parte do esforço nessa disciplina é voltado para atividades de leitura. Prepare-se para ler bastante conteúdo online e, também, bastante documentação técnica em inglês. Mantenha as leituras em dia, conforme o cronograma de estudos. Acumular leituras é “fatal” para o aprendizado; e
- 🔧 **Trabalhos e atividades:** a maior parte do esforço nessa matéria é representada pelos **Diários de Aprendizagem** (tarefas semanais discursivas obrigatórias) e trabalhos de programação (em especial os **PSETs** são atividades que consumirão bastante tempo: não deixe para começar os PSETs faltando um ou dois dias para o prazo de entrega).

Você já percebeu que esta disciplina precisa de um grau de **dedicação de tempo e esforço** consideráveis. Mesmo assim, se você estiver gastando mais tempo do que esforço total indicado acima, pode ser um sinal de que você precisa de ajuda para entender melhor o material. Entre imediatamente em contato com o professor, não deixe acumular dúvidas.

Eu espero que você, como estudante universitário³⁰, esteja buscando aprender o máximo possível e fazendo todo esforço para se dedicar nas atividades propostas e aprender por conta própria de acordo com as orientações do professor. Lembre-se: a única maneira de aprender é estudar! Leia os textos indicados abaixo para saber o que faz a diferença no momento de estudar:

- [Como estudar? Dicas ao estudante iniciante](#)³¹
- [O problema é o tamanho do lápis](#)³²
- [Você gasta sua borracha?](#)³³

Eu também espero que você seja um **estudante** e não um **aluno**. E que você tenha **brio**, seja **bravo**, **persistente** e **resiliente** para encarar o desafio! Assista os pequenos vídeos indicados a seguir para saber a diferença entre aluno e estudante, e para saber a necessidade de brio para estudar:

- ▶ [Aluno ou estudante?](#)³⁴ Do professor Pierluigi Piazzi; e
- ▶ [Você tem brio?](#)³⁵ Do professor Clovis de Barros Filho (desconsidere os palavrões)

Para saber como **vencer a procrastinação e manter o estudo em dia**, assista ao seguinte vídeo (ative as legendas em português se necessário):

- ▶ [Why we procrastinate?](#)³⁶, de Vik Nithy.

Cuidado

Uma das principais causas de falha e de reprovação nesta disciplina é deixar acumular o estudo. Faça de tudo para **estudar diariamente** e manter sempre o **estudo e tarefas em dia**. Muito importante: **não deixe acumular conteúdo!**

³⁰O estudo em nível universitário é **muito diferente** do estudo em nível de ensino médio. Na universidade você terá que ler muito e aprender muita coisa por conta própria. Não espere que o professor vá passar todo o conteúdo mastigado e resumido para você decorar para a prova, essa vida acabou!

³¹<https://cmprz.me/comoestudar/>

³²<https://cmprz.me/lapis/>

³³<https://cmprz.me/borracha/>

³⁴<https://cmprz.me/piazzi1>

³⁵<https://cmprz.me/brio>

³⁶<https://cmprz.me/procrast>

4.5 Datas e horários das aulas

O período 2024/2 se inicia no dia 29/jul/2024 e termina no dia 30/nov/2024. Entre os dias 01/dez/2024 e 15/dez/2024 serão realizadas as avaliações de segunda chamada e as avaliações finais de recuperação. As notas finais serão liberadas no dia 16/dez/2024.

Cada turma tem um dia específico de aula teórica, consulte a data de sua turma no [site da disciplina](#). Além das datas de aula teórica a disciplina contará com as seguintes atividades extras opcionais:

- 📅 **Laboratórios de programação:** todas as segundas, terças e quartas, no Laboratório 12, de 13:00h até 16:00h; e
- 📅 **Monitorias:** todas as quintas e sextas, de 13:00h até 16:00h.

Apesar da participação nas atividades de laboratório de programação e de monitoria não serem obrigatórias, é importante que você tente se programar para participar.

4.6 Ambientes virtuais utilizados

- **Portal do Aluno da UVV:** é o ambiente oficial para a comunicação entre o professor e os alunos (através do blog de mensagens) e para a divulgação do boletim de notas; e
- **Disciplinas UVV:** é o ambiente adicional de auxílio aos alunos, contando com o cronograma previsto das aulas, materiais para download, roteiro de estudos, tarefas de casa, diários de aprendizagem e muitos outros recursos.

Atenção: se você estiver sem acesso ao Portal do Aluno da UVV, entre em contato com a coordenação do curso.

4.7 Materiais necessários

- ✍️ **Material escolar:** você precisará de caderno, lápis escuro (de preferência 4B ou 6B), borracha e apontador para fazer anotações nas aulas (as anotações de aula são fundamentais para seu aprendizado).
- 🦆 **Pato de borracha:** é interessante que você tenha um pato de borracha (daqueles amarelinhos para bebês mesmo) para praticar uma técnica chamada de *Rubber Duck Debugging*³⁷.
- 📖 **Livro texto de referência:** utilizaremos como livro texto de referência o livro online da CR6.100B.
- 📖 **Outros livros recomendados:** se necessários, serão informados pelo professor.
- ✚ **Outros:** se necessário, serão detalhados pelo professor.

5 Dinâmica da disciplina

Esta disciplina tem carga horária total de 60 horas semestrais (40h presenciais e 20h online) e está dividida em aulas, monitorias, tutorias, diários de aprendizagem e atividades no Autolab (incluindo exercícios, laboratórios e PSETs).

³⁷Ver também: <https://cs50.noticeable.news/posts/rubber-duck-debugging-in-cs-50-ide>.

5.1 Aulas

Cada turma terá 1 (uma) aula presencial por semana, com 1:40h de duração, conforme o calendário da UVV (consulte a seção específica de sua turma para ver os dias/horários de sua turma). A presença nas aulas é obrigatória e terá peso na nota da disciplina.

Perigo

Conforme normas da UVV você pode acumular até 10 horas de faltas, o que corresponde a perder 5 dias de aula. Caso você ultrapasse as 10 horas (ou 5 dias) de faltas você será **automaticamente reprovado** na disciplina, sem possibilidade de recurso. O professor tem liberdade de escolher a política de verificação de presença que preferir (por exemplo: fazer uma única chamada durante a aula, fazer uma chamada no início e outra no final, utilizar listas em papel ou o sistema online).

5.2 Monitorias

Cada turma terá, além das horas de aula normais, até 3 horas semanais de monitoria, que podem ser feitas na quinta ou na sexta (você deve escolher o dia de sua monitoria através do questionário inicial da disciplina). As monitorias são realizadas no período vespertino, de 13:00h até 16:00h. A presença nas monitorias não é obrigatória mas é **ALTAMENTE RECOMENDADA** porque você pode conseguir até **1 PONTO EXTRA** na nota do semestre (0,5 por bimestre), e você estará **REVISANDO O CONTEÚDO E APRENDENDO** com os monitores. Os monitores farão o seguinte durante as monitorias:

- Aulas de revisão do conteúdo da aula;
- Aulas com conteúdos extras importantes;
- Ajudarão nos exercícios, laboratórios e PSETs no Autolab;
- Estarão à disposição para esclarecimento de dúvidas;
- Receberão e farão a conferência dos diários de aprendizagem;
- Avaliarão os códigos que vocês enviarem no Autolab, quanto à correção, design e estilo;
- Utilizarão ferramentas de inteligência artificial para a detecção de cola ou plágio nas atividades do Autolab;
- Fornecerão feedback nas atividades do Autolab; e
- Farão a correção dos diários de aprendizagem.

Importante

A **participação nas monitorias é importante para seu aprendizado!** Apesar de introdutória esta não é uma disciplina fácil e você terá que dedicar um tempo considerável de estudo e realização de atividades. Você terá muitas dúvidas que devem ser sanadas com os monitores. Como a participação nas monitorias **não é obrigatória** mas é importante para seu aprendizado, faça um esforço para participar. Você não será reprovado se não comparecer às monitorias, mas perderá os pontos extras e toda a revisão e discussão de conteúdo com os monitores. Um erro comum que os alunos cometem é deixar para participar das monitorias apenas um ou dois dias antes da prova... se esse é seu caso, não perca nem seu tempo: o conteúdo da matéria é muito grande e você não conseguirá aprender em dois dias antes da prova.

5.3 Tutorias

Além de poder participar das monitorias, juntamente com seus colegas, você pode marcar tutorias com o professor. As tutorias são pequenas reuniões (presenciais ou online), de modo individual ou em pequenos grupos, com o professor, para esclarecimento de dúvidas ou busca de ajuda. A participação nas tutorias é opcional.

As tutorias duram, em geral, 20 minutos e devem ser agendadas previamente. O calendário e a forma de agendamento serão divulgados pelo professor após o início das aulas.

5.4 Laboratórios de programação

Cada turma terá, além das horas de aula normais, até 9 horas semanais de atividades de laboratório de programação, que podem ser feitas na segunda, terça ou quarta-feira (você pode participar em quantos dias quiser). Os laboratórios de programação são realizados no período vespertino, de 13:00h até 16:00h, e são orientados e organizados pelos monitores. A presença nos laboratórios não é obrigatória mas é **ALTAMENTE RECOMENDADA** porque você pode conseguir até **1 PONTO EXTRA** na nota do semestre (0,5 por bimestre), e você estará **APRENDENDO A PROGRAMAR** e tendo auxílio direto nas atividades do Autolab. Os monitores farão o seguinte durante os laboratórios de programação:

- Ajudarão nos exercícios, laboratórios e PSETs do Autolab;
- Estarão à disposição para esclarecimento de dúvidas;
- Auxiliarão nas tarefas de programação; e
- Explicarão conteúdo adicional, se necessário.

Importante

A **participação nos laboratórios** é importante para seu aprendizado! As tarefas no Autolab exigirão um grau de esforço considerável, e você terá muitas dúvidas que devem ser sanadas com os monitores. Faça um esforço para participar. Você não será reprovado se não comparecer aos laboratórios de programação, mas perderá os pontos extras e toda a revisão e discussão de conteúdo com os monitores. Um erro comum que os alunos cometem é deixar para participar dos laboratórios apenas um ou dois dias antes do prazo de entrega de algum PSET... não faça isso, os PSETs são difíceis e devem ser feitos com bastante antecedência.

5.5 Diários de aprendizagem

São exercícios discursivos sobre o conteúdo da matéria. Praticamente em todas as semanas você receberá um arquivo PDF com as questões discursivas referentes ao conteúdo da semana. Esse arquivo deve ser **impresso** e respondido de forma **manuscrita**, e entregue para seu monitor ou professor nas datas especificadas no cronograma da disciplina. Os diários também fazem parte das notas bimestrais da disciplina, e são **atividades obrigatórias** (valem 10% da nota de cada bimestre).

Importante

Os diários de aprendizagem são uma ferramenta poderosa de estudo e auto-avaliação. Eles serão vistoriados pelos monitores e/ou professor, mas não serão corrigidos individualmente. É sua responsabilidade fazer os exercícios do diário e, em caso de dúvidas, buscar ajuda com os monitores.

Vamos repetir: todas as dúvidas e dificuldades nos exercícios dos diários de aprendizagem devem ser discutidas com os monitores nos dias de monitoria. Verifique no calendário de sua turma as datas de entrega dos diários, para não perder pontos por atraso. A nota dos diários será calculada de acordo com a proporção de questões que foram respondidas com uma tentativa séria (ou seja, mesmo que você tenha errado alguma questão, se você tentou seriamente responder, essa questão contará positivamente para o cálculo da nota).

5.6 Atividades no Autolab

Durante todo o curso você fará diversas atividades de programação para a resolução de problemas nos mais variados campos do conhecimento humano. Apesar de você programar as soluções para os problemas, o foco não é em aprender uma ou outra linguagem de programação mas, sim, aprender computação. Essas atividades no Autolab são divididas em três grupos:

- **Exercícios:** são tarefas básicas e relativamente fáceis de programação, geralmente para ajudar a fixar um conceito ou uma técnica;
- **Laboratórios (labs):** são tarefas mais complexas de programação, que envolverão mais raciocínio e compreensão mais profunda da matéria; e
- **PSETs:** são problemas extremamente difíceis que desafiarão toda sua capacidade de resolução e pensamento computacional. Prepare-se para gastar várias horas por semana na resolução dos PSETs.

Todas as atividades do Autolab **são obrigatórias** e valerão um peso considerável da nota da disciplina. Além disso a entrega de cada atividade é rigorosamente controlada: entregas em atrasos podem ter a pontuação severamente prejudicada e/ou zerada. Verifique no calendário de sua turma as datas de entrega de cada atividade no Autolab.

Cuidado

Todas as atividades de programação enviadas para o Autolab são analisadas para a detecção de plágio e/ou cola e/ou cópia de código da Internet. A verificação é feita através de dois sistemas altamente sofisticados:

- **MOSS: Measure of Software Similarity**, da Universidade de Stanford; e
- **COMPARE50: Similarity in Code**, da Universidade de Harvard.

Esses sistemas são extremamente sofisticados e detectam plágio e/ou cópia de código mesmo que você tente disfarçar ou modificar o programa (trocar nomes de variáveis, mudar a indentação, reescrever comentários, etc.). Por favor, **não tente a sorte**: você não vencerá os softwares, receberá nota 0 (zero) na atividade e poderá ser encaminhado para a coordenação para as sanções disciplinares previstas nas regras da UVV.

5.7 Problem Sets (PSETs)

Um *Problem Set* (PSET) é uma lista com um número relativamente pequeno de **questões difíceis**, feitas para testar sua compreensão global do conteúdo estudado durante toda a semana.

Ao contrário das atividades rápidas e trabalhos semanais, que são relativamente fáceis e diretos (pois têm o objetivo de auxiliá-lo no estudo do conteúdo de uma semana), os PSETs são projetados para forçá-lo a pensar profundamente no que você aprendeu durante

a semana; para conseguir responder às questões dos PSETs você precisará ter desenvolvido uma compreensão profunda da matéria, muito além do nível de simples “decoreba”.

Deixe-me alertá-lo mais uma vez: os **PSETs são difíceis**, todas as questões são difíceis e algumas são incrivelmente difíceis. Por esse motivo você terá uma ou duas semanas para realizar cada PSET (e não fique chateado se não conseguir responder completamente uma ou mais questões de cada PSET, eles foram feitos para desafiá-lo; mas é importante que, mesmo que você não consiga responder alguma questão, você se esforce e tente ao máximo encontrar a solução).

Para ter uma idéia melhor do que esperar de cada PSET, e porque o uso de PSETs é importante, dê uma olhada no que os alunos do *Massachusetts Institute of Technology* (MIT) acham do uso de PSETs em suas disciplinas:

- *Harbinger of Doom, Despair, and Knowledge: PSETS* (“Arautos da Desgraça, Desespero e Conhecimento: PSETS”);
- *A love letter to psets* (“Uma carta de amor aos psets”); e
- *The Process of Psetting* (“O Processo de Psetting”).

O grau de dificuldade de cada PSET não atingirá o grau exigido dos alunos do MIT (ainda!), mas mesmo assim serão muito mais difíceis do que você pode estar acostumado (mas afinal... você quer aprender computação e programação, não é?).

A resolução dos PSETs é **fundamental para seu aprendizado**. Trabalhe diariamente em cada PSET: é melhor fazer várias pequenas partes todos os dias do que tentar fazer tudo de uma vez, de última hora.

Devido ao grau de dificuldade, recomenda-se **trabalhar em pequenos grupos** na resolução de cada PSET, desde que você siga ao pé da letra todas as regras de **integridade acadêmica** do curso (Seção 8), em especial as **políticas sobre trabalho colaborativo** (Seção 8.1).

A questão da integridade acadêmica é muito séria e eu levo isso ao pé da letra. Por favor, não tente entregar o trabalho de um colega (no todo ou até mesmo alguma pequena parte) como seu: existem ferramentas online que verificam plágio em questão de segundos e, além disso, eu realmente leio as respostas de todos os PSETs. E pior ainda, você está se enganando...

A melhor opção para fazer o PSET é participar dos laboratórios de programação vespertinos, onde o monitor estará à sua disposição para auxiliá-lo. Cada PSET terá instruções específicas que indicam como as respostas devem ser enviadas no Autolab.

6 O que esperamos de você?

Nós esperamos que você:

- Esteja presente em todas as aulas (prestando atenção e participando ativamente das discussões);
- Esteja presente em 1 tarde de monitoria por semana (na quinta ou na sexta). A presença não é obrigatória, mas é fundamental para seu aprendizado e vale até 1 ponto extra na nota semestral (0,5 pontos extra por bimestre);
- Faça e entregue todos os diários de aprendizagem para análise do monitor e/ou professor;
- Faça todos os exercícios e laboratórios no Autolab;
- Faça todos os PSETs; e
- Siga todas as normas de integridade acadêmica da disciplina.

7 Avaliação da aprendizagem e notas

As notas serão baseadas em uma escala de 10 pontos (de 0 até 10), e serão calculadas a cada bimestre (ou seja, cada bimestre vale até 10 pontos). Ao final do semestre é calculada a média das notas bimestrais e você será aprovado ou não de acordo com os seguinte critérios:

- nota semestral $\geq 7,0$: você está aprovado;
- $3,0 \leq$ nota semestral $< 7,0$: você está de recuperação e deverá fazer a prova final de recuperação; e
- nota semestral $< 3,0$: você está reprovado (não poderá nem fazer a prova de recuperação).

Caso você vá para prova final de recuperação, sua nota final será calculada como a média entre a nota semestral e a nota da prova de recuperação. Você será aprovado ou não de acordo com os seguinte critérios:

- média pós-recuperação $\geq 5,0$: você está aprovado; e
- média pós-recuperação $< 5,0$: você está reprovado.

7.1 Peso das atividades nas notas bimestrais

Você fará diversas atividades durante o curso e cada atividade terá um peso diferente na composição da nota bimestral. Os pesos estão discriminados abaixo:

Figura 2: Peso de cada atividade nas notas

Atividade	Peso	
Diários de Aprendizagem	10%	
Prova Bimestral	30%	
Autolab	Exercícios	5%
	Labs	10%
	PSETs	45%

Além das notas conforme as atividades acima, cada bimestre terá até 1,0 (um) ponto extra calculado proporcionalmente à participação nas monitorias e laboratórios de programação. Outras notas extras (participação no Inova Week ou outra atividade proporcionada pela UVV serão considerados posteriormente).

7.2 Arredondamento de notas

Todas as notas que forem lançadas no Portal do Aluno já estarão arredondadas de acordo com os [Critérios de Arredondamento de Notas](#)³⁸. Por favor leia esse documento com atenção! Nenhuma nota será arredonda fora desses critérios.

³⁸<https://disciplinas.uvv.br/assets/docs/arredondamento.pdf>

Perigo!

Se sua nota semestral for menor ou igual a 6,94 você IRÁ PARA A PROVA FINAL DE RECUPERAÇÃO. Por favor, leia o documento com os critérios de arredondamento.

7.3 Entregas em atraso

Verifique no cronograma específico de sua turma as datas de entrega das diversas atividades. Atividades manuscritas devem ser entregues diretamente ao professor e/ou aos monitores, até o limite indicado no cronograma.

As atividades no Autolab também possuem datas específicas de entrega, do seguinte modo:

- Data de entrega: é a data oficial de entrega, é o prazo que você deve obedecer;
- Data final: corresponde ao prazo máximo, após a data de entrega, no qual o Autolab ainda receberá as atividades. ATENÇÃO: as entregas serão penalizadas em 10% da nota, por dia de atraso!

Cada aluno também terá 10 dias de crédito para a entrega de atividades em atraso (após a data de entrega e antes da data final), sem que ocorra o desconto de 10% por dia de atraso. Esses dias de crédito devem ser controlados por você. Utilize com sabedoria, mantenha os dias de crédito para situações nas quais você realmente pode precisar (como doenças ou outros problemas sérios). Ao esgotar seus dias de crédito, toda entrega em atraso será penalizada.

8 Integridade acadêmica

Uma questão fundamental relacionada ao trabalho que você fará nesta disciplina (e, na verdade, em qualquer ambiente acadêmico) diz respeito a sua **integridade acadêmica**, ou seja: nós esperamos que suas escolhas sejam íntegras, responsáveis e honestas.

As principais violações acadêmicas que os alunos costumam cometer são: plágio, colaboração não autorizada, “cola” e entregar o trabalho de um colega (total ou parcialmente) como se fosse seu. Por favor, não quebre as regras de integridade acadêmica: você está se enganando, enganando o professor e enganando a UVV.

Para auxiliá-lo a entender melhor o que pode ou o que não pode ser feito, sugiro que você leia o manual *Academic Integrity at MIT: A Handbook for Students*, que é um dos melhores e mais objetivos textos a respeito do assunto:

 [Academic Integrity at MIT: A Handbook for Students](#)

O site com as regras de integridade acadêmica do MIT traz muitas informações resumidas de forma clara e objetiva para os alunos, e também recomendo que você dê uma olhada:

 [Academic Integrity at MIT](#)

Um outro site importante que você deve conhecer é o site com as regras de integridade acadêmica da disciplina *Harvard University: CS50* (a disciplina de introdução à ciência da computação de Harvard). O site da disciplina tem uma lista bem clara de coisas que podem e que não podem ocorrer:

 [Academic Honesty at Harvard CS50](#)

Atenção: **esta disciplina seguirá as regras de integridade acadêmica do MIT e de Harvard**, conforme explicadas nos links acima. É sua responsabilidade ler, e seguir, essas regras. Em caso de dúvidas quanto a alguma regra, entre em contato com o professor. Alguns exemplos do que pode e do que não pode ser feito, baseados nas regras acima citadas de Harvard e do MIT:

🔊 Atitudes permitidas:

- Conversar com os colegas a respeito das atividades, exercícios e PSETs, desde que você não esteja pedindo a resposta;
- Discutir os materiais do curso com os colegas, para compreender melhor o conteúdo da matéria e esclarecer dúvidas;
- Ajudar um colega a identificar um erro ou bug em alguma atividade ou exercício, desde que você não faça a correção ou forneça a resposta (o seu colega é que tem que se esforçar, baseado em suas dicas);
- Incorporar algumas linhas de código que você encontrou online na sua própria solução, **desde que essas linhas não sejam a solução** para o problema, e que **cite e identifique** quais foram as linhas (e coloque um link para os originais);
- Estudar as atividades, exercícios e PSets dos semestres passados como forma de aprendizado extra (e não, as atividades e PSets deste semestre não serão iguais aos dos semestres passados);
- Olhar as respostas e códigos de seus colegas para ajudá-los caso eles estejam tendo dificuldades, desde que você apenas identifique onde estão os erros/bugs e não forneça a solução, apenas dê dicas e orientação de como eles podem ser resolvidos;
- Buscar ajuda do professor (incluindo monitores, tutores e alunos mais experientes) para esclarecer a matéria e ajudar na compreensão e resolução das atividades e PSets, desde que você faça uma pergunta específica e não queira a solução;
- Buscar material extra na biblioteca ou na internet para aprender e estudar, desde que esse material extra não seja a solução para as atividades e PSets; e
- Explicar para seu colega como obter a solução de um problema usando frases, diagramas ou pseudocódigo genérico, mas sem mostrar ou fornecer a resposta final.

🚫 Atitudes proibidas:

- Pedir a solução de um problema para seu colega, ou procurar na internet uma solução pronta;
- Pedir para ver a solução de um colega para saber se a sua solução está “batendo”;
- Não citar o nome de colegas com os quais você trabalhou em grupo na resolução de alguma atividade ou PSet;
- Não citar a fonte e não indicar o link de linhas de código que você encontrou online e utilizou como parte de sua solução;
- Dar ou mostrar para os colegas a solução para questões que eles ainda não conseguiram responder (você pode ajudar e tirar dúvidas, mas não pode mostrar a sua solução);
- Colar nas provas online (principalmente através da internet), ou seja, durante as provas online não é permitido que os alunos se reúnam de forma online para resolver a prova coletivamente;
- Pegar a resposta de um colega (ou até mesmo pequenas partes da resposta) e apresentar como sua;

- Pagar para alguém fazer o trabalho em seu lugar;
- Postar as questões em fóruns e sites na internet, solicitando que alguém as responda;
- Dividir com os colegas as atividades e exercícios de forma que cada um faça uma pequena parte do trabalho e depois compartilhe com o restante (você deve trabalhar em todos os problemas);
- Pegar o trabalho de um colega, mudar algumas frases e vírgulas, e entregar como se fosse seu; e
- Qualquer outra coisa que sua consciência considere desonesta.

8.1 Política sobre trabalho colaborativo

Cientistas, programadores, estatísticos, engenheiros e, em geral, todas as pessoas ligadas à área de “STEM” (*Science, Technology, Engineering, Math*) costumam trabalhar em grupos. As interações sociais são críticas para o resultado de seus estudos e grandes idéias costumam aparecer em discussões com os colegas. Assim, nesta disciplina, o trabalho em grupo é fortemente recomendado e estimulado, desde que algumas regras sejam seguidas.

Seguiremos integralmente a política sobre trabalho colaborativo adotada na disciplina “*MIT 6.001: Structure and Interpretation of Computer Programs*”³⁹ que, em tradução livre, é a seguinte:

“A maioria das pessoas aprende com mais eficácia quando estuda em pequenos grupos e coopera de várias outras maneiras no dever de casa. Isso pode ser particularmente verdadeiro em tarefas que envolvem programação, onde trabalhar com um parceiro geralmente ajuda a evitar erros e descuidos. Somos totalmente a favor desse tipo de cooperação, desde que todos os participantes se envolvam ativamente em todos os aspectos do trabalho — não apenas dividam a tarefa e cada um fica responsável apenas por uma parte.

[...] encorajamos você a trabalhar com uma ou duas outras pessoas mas, ao entregar seu projeto, você deve identificar com quem trabalhou. Esperamos, no entanto, que você esteja envolvido em todos os aspectos do projeto, que escreva e comente seu próprio conjunto de código e que escreva seus resultados separadamente. Quando você entrega um material com seu nome, presumimos que você está certificando que este é o seu trabalho e que esteve envolvido em todos os aspectos dele. Não entregue apenas uma cópia de um único arquivo; escreva sua própria versão. Isso significa que você mesmo cria esse arquivo e não apenas anota uma cópia que recebeu de outra pessoa. Sabemos que isso pode soar como replicação de trabalho, mas uma parte importante do aprendizado do material é tornar o processo ativo, especialmente com relação à edição, execução e depuração de software, o que você faz garantindo que criou e consegue explicar sua solução.

Aqui está um exemplo de como um bom trabalho de cooperação deve ocorrer:

- Ambos (todos) sentam-se com lápis e papel e planejam juntos como resolverão e testarão as coisas. Você vai com seus colegas para algum local adequado e sentam-se em máquinas adjacentes. Após você resolver um problema, você verifica se seus colegas também conseguiram e se estão todos no mesmo ritmo. Quando um de vocês tem um problema, os outros olham e tentam ajudar. Por exemplo, seu parceiro tem um bug em uma

³⁹<https://cmprz.me/sicpocw>

parte e você pode apontar onde está o bug e como corrigi-lo. Em cada parte do problema, você escreve seu próprio código e solução, buscando ajuda dos outros quando tem dificuldades. No trabalho final, cada um de vocês lista os nomes de todos os seus colaboradores.

Aqui está um exemplo de cooperação inapropriada:

- Você envia ao seu amigo uma cópia do seu trabalho até agora. Ele continua o seu trabalho para concluir o procedimento que você não conseguiu, e corrige um bug ou erro em outra parte. Cada um de vocês envia este código e solução compartilhados. Mesmo que você liste os nomes um do outro como colaboradores, esta é uma colaboração inadequada porque vocês não estiveram envolvidos em todos os aspectos do trabalho — cada um de vocês não escreveu sua própria implementação, mesmo que para um plano comum, e você compartilhou um conjunto comum de código e redação.

Não listar o nome de um colaborador será considerado trapaça. Da mesma forma, lembre-se de que copiar o trabalho de outra pessoa e apresentá-lo como seu próprio trabalho é uma ofensa acadêmica grave e será tratado como tal.

Em geral, recomendamos fortemente que você trabalhe em grupo. É uma maneira muito eficaz de detectar erros conceituais e outros, e de refinar o pensamento e a compreensão [...].”

Em resumo: colabore com seus colegas, estude em grupo, mas siga as regras de trabalho colaborativo. Uma dica: antes de participar de um grupo de estudo, tente resolver as questões sozinho primeiro; isso lhe dará uma visão geral dos problemas a serem resolvidos, do grau de dificuldade, e proporcionará que você vislumbre um caminho para a solução que pode ser discutido com seus colegas.

8.2 Cláusula de arrependimento

Esta disciplina oferecerá uma **cláusula de arrependimento**⁴⁰ conforme praticada no curso *Harvard CS50*⁴¹:

Cláusula de Arrependimento

Se você cometer algum ato que viole as políticas de integridade acadêmica aqui estabelecidas mas, em até 72 horas, se arrepender e comunicar o fato ao professor, poderá ter a nota reduzida ou zerada nesse trabalho/PSET específico, mas o caso não será levado à coordenação.

9 Perguntas freqüentes

- **Por que tenho que seguir as regras de integridade acadêmica do MIT e de Harvard?** Em primeiro lugar porque você está na graduação e tem a obrigação de agir com honestidade acadêmica. Em segundo lugar porque as regras do MIT e de Harvard são claras, objetivas e disponíveis livremente na internet para que qualquer pessoa possa utilizar. E, para saber se os alunos estão lendo este documento, a primeira pessoa de cada turma que seguir todas as instruções e links de referências detalhadas aqui e me mandar um e-mail com a “primeira lei de abrantés” ganhará uma caixa de bombom.

⁴⁰<https://cmprz.me/cs50arrependimento>

⁴¹Ver *Teaching Academic Honesty in CS50*, de David Malan, Brian Yu e Doug Lloyd.

E, por último, eu (seu professor) acredito em integridade acadêmica e acredito que essas regras são importantes.

- **As normas de integridade acadêmica serão obrigatórias?** Sim, serão. Por favor, não tente a sorte copiando o trabalho de seu colega ou, se assim o fizer, utilize a cláusula de arrependimento. Isso evitará maiores problemas e dissabores para todo mundo.
- **Li o *syllabus* mas tenho dúvidas!** Entre em contato com o professor!