UFABC - MCTA016-13 - Paradigmas de Programação Q2 - 2021

Permalink

Turmas: DAMCTA016-13SA, NAMCTA016-13SA e NA1MCTA016-13SA
Professores: Emilio Francesquini e Fabrício Olivetti
E-mail: e.francesquini@ufabc.edu.br / folivetti@ufabc.edu.br

---

1. Avisos

- `[2021-05-19 qua]` - Página online

---

2. Informações Gerais

- **Oferecimentos**
  - Turma NAMCTA016-13SA (F. Olivetti)
  - Turma DAMCTA016-13SA e NA1MCTA016-13SA (E. Francesquini)

Diante da pandemia de COVID-19, que impõe a necessidade de adoção de medidas por parte do Poder Público (e da universidade) para a contenção da disseminação da doença, e considerando que o prazo de suspensão das atividades acadêmicas presenciais tende a ser longo, neste quadrimestre a disciplina será ministrada de maneira totalmente online. As regras que regulam esta modalidade são definidas pela Resolução ConsEPE Nº 240/2020.

> **Note**
>
> Todas as aulas, com e sem participação dos alunos, serão gravadas e disponibilizadas online segundo a Licença Creative Commons Atribuição-NãoComercial 4.0 Internacional (CC-BY-NC). Todos os participantes do curso dão sua tácita e irrevogável autorizaçao para que suas imagens e falas sejam transmitidas, gravadas e editadas segundo a licença acima pelo docente responsável, sem nenhuma cobrança, para uso em distintos canais de comunicação e peças publicitárias sem fins comerciais.

2.1. Dinâmica de ensino

Serão disponibilizadas videoaulas com explicação do conteúdo teórico além do conteúdo prático com codificação em tempo real. Como material de apoio os alunos terão acesso aos códigos desenvolvidos durante as aulas, além de poderem contar com o uso do Discord (veja abaixo) para tirar dúvidas e discutir assuntos pertinentes à disciplina.

Semanalmente serão disponibilizadas listas de exercícios sobre o conteúdo apresentado. As entregas destas listas serão consideradas como uma avaliação e deverão ser feitas em até uma semana após a sua disponibilização. As notas dessas atividades serão utilizadas para a composição da média final. As aulas não serão dadas em tempo real devido aos inúmeros possíveis problemas técnicos, já que são dezenas de

alunos matriculados na turma atualmente. Nos dias e horários em que haveria aula e atendimento, estaremos online na ferramenta Discord (que além de chat, faz captura de voz e tela) para tirar dúvidas sobre o conteúdo previsto para aquela data.

---

3. Dias, horários e local das aulas

> **Tip**
>
> Inscreva-se o quanto antes no servidor do Discord da disciplina (https://discord.gg/JSgnfdE). Todos os anúncios e comunicações serão feitos por lá.

Figura 1: Servidor do Discord da disciplina.

Os professores **estarão online no Discord** para tirar dúvidas nos seguintes horários:

- **Terças**
  - 08h00 às 10h00 (Emílio)
  - 19h00 às 21h00 (Fabrício)

- **Sexta**
  - 20h00 às 22h00 (Emílio)

Eventuais dúvidas e questionamentos poderão ser enviados em outros horários. Contudo, fora dos horários acima, o professor pode não atendê-los prontamente devido as suas outras atividades.

Lembramos também que a disciplina é unificada, mesmos trabalhos, mesmos critérios, mesmas aulas entre todas as turmas. Então fique a vontade para escolher dentre os horários acima qual o melhor te atende, independente da turma onde estiver matriculado.

Os vídeos das aulas serão disponibilizados online, na seguinte página:
http://haskell.pesquisa.ufabc.edu.br/

Eventualmente também teremos aulas síncronas, com live coding. Essas aulas serão anunciadas tanto nesta página quanto no Discord (link acima).

4. Sobre a Disciplina

MCTA016-13 - Paradigmas de Programação

- TPI: 2-2-4
- Recomendação: Processamento da Informação, Programação Orientada a Objetos

> ***Objetivos***
>
> *Esta disciplina traz à atenção do aluno as diversas diferenças fundamentais entregrandes famílias de linguagens de programação, tanto em teoria como de forma prática. Estavisão tem efeitos importantes, nem sempre perceptíveis: ao apreciar diversas técnicas deprogramação e mecanismos peculiares de linguagens de programação diferentes, o estudanteexpande seu leque de técnicas e rompe sua rigidez de concepção a respeito do que vem a serprogramar. Além disso, há situações onde um paradigma se aplicará com mais sucesso do queoutro. Pode-se sem dúvida afirmar que a programação em diferentes paradigmas auxilia namelhoria da qualidade da programação de forma geral.*
>
> ***Conteúdo Programático***
>
> *Visão comparativa entre os paradigmas de programação. Paradigma funcional. Paradigma concorrente.*

Fonte: Projeto Pedagógico do BCC 2017

5. Datas Importantes

5.1. Listas semanais

Teremos listas semanais que serão utilizadas para a avaliação. Cada lista semanal é composta de:

- Resumo/anotações de estudo (2 pontos)
  - Os resumos deverão possuir no máximo 300 palavras e serão publicados no site da disciplina junto ao tópico de estudo ao qual se relacionam.

- Exercícios de programação (8 pontos)
  - Os exercícios semanais de programação serão divididos em: 50% nível fácil, 25% nível intermediário e 25% nível desafiador

As datas de liberação dos enunciados e prazo para entrega são listados abaixo:

Lista	Enunciado/Prazo	Resolução	Link	Link Ouvintes
1	28/05 - 07/06			
2	04/06 - 14/06			

Lista	Enunciado/Prazo	Resolução	Link	Link Ouvintes
3	11/06 - 21/06			
4	18/06 - 28/06			
5	25/06 - 05/07			
6	02/07 - 12/07			
7	09/07 - 19/07			
8	16/07 - 26/07			
9	23/07 - 02/08			
10	30/07 - 09/08			

❗ Warning

- Todas as entregas devem ser feitas **exclusivamente pelo Github** Classroom através dos links acima. Listas enviadas por outros meios não serão aceitas.
- Entrega toda segunda-feira até às 23:59. Listas entregues com atraso de até um dia serão aceitas, contudo com um desconto de 50% da nota.

❗ Danger

- O código submetido **precisa** compilar ao executar `stack test`
- Caso não consiga fazer algum exercício coloque a palavra-chave `undefined` como definição da função. Ex.: `isEven = undefined`
- A execução de `stack test` não pode travar e nem entrar em loop infinito.

A falha em qualquer um desses casos acarretará em nota 0 na lista inteira. Assistam os vídeos com as correções das listas de exercício (em especial das listas 1 e 2) do oferecimento anterior da disciplina que podem ser encontrados aqui. Eles exploram por cada um dos pontos acima.

5.2. Projeto de programação

Data para entrega: **16/08/2021**

- Entregas via Github Classroom
  - Diurno: Link em breve
  - Noturno: Link em breve

- O projeto é **individual**[1].
- Todos os *commits* e o *push* final devem ter sido feitos até no máximo às **23h59 do dia 16/08/2021**.
  - Atenção, caso ocorram/commits/pushes após o prazo, será considerada apenas a última versão para avaliação. Caso o último esteja com atraso > 3 dias, então será considerado o último commit

até 3 dias (valendo no máximo 5).
  - Entregas em atraso sofrerão descontos conforme a tabela abaixo.

Dias em atraso	Nota máxima
1 dia	7
2 dias	6
3 dias	5
>3 dias	0

- No mesmo repositório do seu código, deverá haver um relatório de no máximo 2 páginas (em PDF! nada de `.md`, `.docx`, `.odt`, ou qualquer outro formato da moda...) descrevendo:
  - Qual é o seu projeto
  - Como utilizar (se for qualquer coisa diferente de `stack run`) o seu código
  - Dificuldades, surpresas e destaques do seu código

- O seu relatório também deverá conter **obrigatoriamente** um link para um vídeo que:
  - Esteja abrigado no Youtube (ou qualquer serviço semelhante)
  - Seja privado/não listado e disponível apenas àqueles com o link
  - Tenha no máximo 5 minutos (ou seja, <= 300 segundos). **Sem exceções**.
  - Mostre a compilação, o uso e uma apresentação do código do projeto

- Durante a correção os professores podem pedir por esclarecimentos adicionais.

Que tipos de projetos serão aceitos?

Lembre-se que você terá o quadrimestre todo para entregar o seu projeto. Logo é esperado que o seu projeto reflita um esforço de desenvolvimento compatível com o prazo disponível.

Como exemplo, em oferecimentos anteriores da disciplina tivemos alunos que implementaram jogos (inclusive dois deles que implementaram um clone completo do Pac-Man), algoritmos de IA e aprendizado de máquina e avaliações de desempenho, paralelizações de algoritmos clássicos, ferramentas completas para sincronização de arquivos, clientes de e-mail...

Caso esteja com dúvidas se a sua ideia é o suficiente para um projeto final que será bem avaliado converse com o professor o quanto antes!

---

6. Listas de Exercícios Complementares

Listas de exercícios extras (não valem nota) para aqueles que quiserem ir um pouco além...

- Lista 1
- Lista 2
- Lista 3

- Lista 4
- Lista 5
- Lista 6

Exercícios adicionais:

- Cifra de César
- Recursão
- Sistemas de voto
- Tautologias
- Jogo da Zebra
- K-Means Paralelo (Arquivo de Entrada - Wine Quality Data)

---

7. Aulas

Neste oferecimento, a disciplina será oferecida em conjunto pelos professores Emilio Francesquini e Fabrício Olivetti. Por isto todo o material das aulas (slides, vídeos, textos) serão compartilhados. O site com esse material compartilhado está disponível em:

http://haskell.pesquisa.ufabc.edu.br/haskell.html

A playlist com todos os vídeos das aulas teóricas do curso pode ser encontrada aqui:

https://www.youtube.com/playlist?list=PLYItvall0TqJ25sVTLcMhxsE0Hci58mpQ

Para facilitar, na tabela abaixo você encontra os assuntos juntamente com links diretos para o conteúdo e vídeos.

Semana Sugerida	Tópico	Material de Estudo	Playlist
1	Preparando o ambiente	https://haskell.pesquisa.ufabc.edu.br/posts/haskell/00.bootstraping.html	https://www.youtube.com/playlist?list=PLYItvall0TqLedblNsncIUfk3cHv_F
1	Paradigmas de programação	https://haskell.pesquisa.ufabc.edu.br/posts/haskell/01.paradigmas.html	https://www.youtube.com/playlist?list=PLYItvall0TqIZ_ih1mSoc1roCZGIfj
2	Cálculo Lambda	https://haskell.pesquisa.ufabc.edu.br/posts/haskell/02.lambda.html	https://www.youtube.com/playlist?list=PLYItvall0TqKPbnSblJ$_{fxNIFRgEoI}$-7_
2	Conceitos Básicos - Parte 1	https://haskell.pesquisa.ufabc.edu.br/posts/haskell/03.haskell.basico.1.html	https://www.youtube.com/playlist?list=PLYItvall0TqLICPN9vbDIc8FAKhG
2	Conceitos Básicos - Parte 2	https://haskell.pesquisa.ufabc.edu.br/posts/haskell/04.haskell.basico.2.html	https://www.youtube.com/playlist?list=PLYItvall0TqLICPN9vbDIc8FAKhG
3	QuickCheck	https://haskell.pesquisa.ufabc.edu.br/posts/haskell/05.quickcheck.html	https://www.youtube.com/playlist?list=PLYItvall0TqJ25sVTLcMhxsE0Hci5

Semana Sugerida	Tópico	Material de Estudo	Playlist
3	Funções de Alta Ordem	https://haskell.pesquisa.ufabc.edu.br/posts/haskell/06.higher.order.html	https://www.youtube.com/playlist?list=PLYItvall0TqLBLt6oXFVBaloU7-xZ
4	Tipos de Dados Algébricos	https://haskell.pesquisa.ufabc.edu.br/posts/haskell/07.adt.html	https://www.youtube.com/playlist?list=PLYItvall0TqJWdfiLuMsIm_4ag3fJj
5	Monoid e Foldable	https://haskell.pesquisa.ufabc.edu.br/posts/haskell/08.monoids.foldable.html	https://www.youtube.com/playlist?list=PLYItvall0TqIzShhLgcVVti0d0kAaJ
5	Functor	https://haskell.pesquisa.ufabc.edu.br/posts/haskell/09.functors.html	https://www.youtube.com/playlist?list=PLYItvall0TqKtSdeBAjz0GOM3oq3p
6	Applicative Functor e Traversable	https://haskell.pesquisa.ufabc.edu.br/posts/haskell/10.applicatives.traverse.html	https://www.youtube.com/playlist?list=PLYItvall0TqLyKtI_Cnx0g_H4Bk68
7	Monads - Parte 1	https://haskell.pesquisa.ufabc.edu.br/posts/haskell/11.monads.html	https://www.youtube.com/playlist?list=PLYItvall0TqLW_mPtIpVA8qHpuQ
8	Monads - Parte 1	https://haskell.pesquisa.ufabc.edu.br/posts/haskell/11.monads.html	https://www.youtube.com/playlist?list=PLYItvall0TqII6y3FydOuEjFMH6jp
9	Avaliação Preguiçosa	https://haskell.pesquisa.ufabc.edu.br/posts/haskell/12.laziness.html	https://www.youtube.com/playlist?list=PLYItvall0TqJwLa9rY-bT_B9-Emtai
9	Haskell Paralelo	https://haskell.pesquisa.ufabc.edu.br/posts/haskell/13.paralelismo.html	https://www.youtube.com/playlist?list=PLYItvall0TqJ1jteUbGOHfBycg5NI
10	Programação Concorrente	https://haskell.pesquisa.ufabc.edu.br/posts/haskell/14.concorrencia.html	https://www.youtube.com/playlist?list=PLYItvall0TqKz0Jw8RTA2epq8Vim
10	Estruturas de dados funcionais e persistência	https://haskell.pesquisa.ufabc.edu.br/posts/eds/01.persistencia.html	https://www.youtube.com/playlist?list=PLYItvall0TqIOxQzCMsK3zciIXxxg
11	Árvores: Roseiras e Rubro-Negras	https://haskell.pesquisa.ufabc.edu.br/posts/eds/02.arvores.html	
11	Heaps funcionais	https://haskell.pesquisa.ufabc.edu.br/posts/eds/03.heaps.html	
12	Estruturas funcionais avançadas: Finger Tree		

Vídeos e códigos adicionais sobre assuntos específicos gravados durante oferecimentos anteriores da disciplina podem ser vistos aqui:

Assunto	Vídeo	Código
Contador de palavras	https://youtu.be/XutIhXNxBXY	contador.zip

Assunto	Vídeo	Código
Stack + Github Classroom + Exercícios	https://youtu.be/cAcBwg2cG-c	-
Criando gráficos com HVega	https://youtu.be/DReqEs-g-9g	plotsVega.zip
Gráficos 2D com OpenGL usando Gloss	https://youtu.be/jtgcJrDQR8U	paredao.zip
Jogo de Damas	https://youtu.be/0Rjmy-I2c3s	checkers.zip
🐇 → 🍅: ADTs, Grafos e Busca em largura	https://youtu.be/nUkvGPQ9-Ps	aula-grafos.zip
Tabelas Monoids	https://youtu.be/Tz3kEiEyETA	tabelas.zip
Revisão Haskell Básico - Recursão e folds	https://youtu.be/xm9FTZJ9rX0	aula-noturno201026.zip
🦓 Zebras e Monads 🍡	https://youtu.be/eZIV_WgwGJQ	zebra.zip
Resolução da Lista 6	https://youtu.be/pNwSJ-Rq3L0	lista06.zip
Revisão: Monoids, Functor, Applicative e Monads	https://youtu.be/xwgbjv2br2I	RevisaoDiurno.zip
Revisão: Monoids. Estudo de caso: State Transformer	https://youtu.be/I4Obrvdn78I	monoids-statetransformer.zip
Revisão State e IO Monads	https://youtu.be/3rabpK1gMho	monadas.zip
Concorrência *vs.* Paralelismo	https://youtu.be/LLoi8hyeaGY	aula-23nov.zip
F-Álgebra	https://youtu.be/EDIS73znRwA	falgebra.zip
Par Monad + Criterion + Threadscope	https://youtu.be/5Ny43b5Tfpc	fractal.zip

8. Notas

Em breve!

9. Critério de avaliação

>  ***Honestidade Acadêmica***
>
> *Entre outros, o código de ética da UFABC estabelece em seu artigo 25 que é* ***eticamente inaceitável*** *que os discentes:*
> *I - fraudem avaliações;*
> *II - fabriquem ou falsifiquem dados;*
> *III - plagiem ou não creditem devidamente autoria;*
> *IV - aceitem autoria de material academico sem participação na produção;*
> *V - vendam ou cedam autoria de material acadêmico próprio a pessoas que não participaram da produção.*
>
> *Muitos ainda têm dúvidas sobre a interpretação das regras definidas pelo Código de Ética da UFABC. Por esta razão, diversos professores elaboraram um documento (**disponível aqui**) com vários exemplos e esclarecendo a interpretação das regras acima. Abaixo uma versão resumida.* ***Sempre consulte o documento completo ou converse com o seu professor em caso de dúvidas!***
>
> - ***Regra 1*** *- Você não pode enviar para avaliação um trabalho que não seja de sua própria autoria ou que seja derivado/baseado em soluções elaboradas por outros.*
> - ***Regra 2*** *- Você não pode compartilhar a sua solução com outros alunos nem pedir aos seus colegas que compartilhem as soluções deles com você.*
> - ***Regra 3*** *- Nos trabalhos enviados para avaliação você deve indicar eventuais assistências que você tenha recebido.*
>
>   *ATENÇÃO: todos os trabalhos enviados para avaliação poderão ser verificados por um sistema automatizado de detecção de plágio.*
>
> *Qualquer violação às regras descritas acima implicará:*
>
> - *Descarte dos conceitos atribuídos a TODAS as tarefas avaliativas regulares de TODOS os envolvidos, causando assim suas reprovações automáticas com conceito F .*
> - *Possível denúncia à Comissão de Transgressões Disciplinares Discentes da Graduação, a qual decidirá sobre a punição adequada à violação que pode resultar em advertência, suspensão ou desligamento , de acordo com os artigos 78-82 do Regimento Geral da UFABC.*
> - *Possível denúncia apresentada à Comissão de Ética da UFABC, de acordo com o artigo 25 do Código de Ética da UFABC.*

A avaliação da disciplina será composta pelas seguintes notas:

- N_P é a nota do projeto
- N_L é a nota das listas de exercícios semanais

A nota N_L será calculada como a média aritmética das listas semanais, cada lista valendo 10 pontos, descartando-se a lista com a pior nota. Ou seja, se por qualquer motivo você não puder enviar uma das listas ou caso tenha sido avaliada como uma nota baixa, basta que faça todas as demais para que não haja prejuízo na sua nota.

A nota final (N_F) será determinada pela **média harmônica ponderada** de N_P e N_L, com pesos 7 e 3 respectivamente:

$$N_F = \frac{10}{\frac{7}{\max\{0.1, N_P\}} + \frac{3}{\max\{0.1, N_L\}}}$$

O conceito final (C_F) será obtido de acordo com a equação abaixo:

$$C_F = \begin{cases} \mathbf{F}, & \text{se } N_F \in [0,0; 5,0) \\ \mathbf{D}, & \text{se } N_F \in [5,0; 6,0) \\ \mathbf{C}, & \text{se } N_F \in [6,0; 7,0) \\ \mathbf{B}, & \text{se } N_F \in [7,0; 8,5) \\ \mathbf{A}, & \text{se } N_F \in [8,5; 10,0] \end{cases}$$

Caso seja verificado ocorrência de plágio no projeto final, o aluno será automaticamente reprovado com F

9.1. Recuperação

A resolução ConsEPE nº 182 assegura a todos os alunos de graduação com C_F igual a **D** ou **F** o direito a fazer uso de mecanismos de recuperação.

A recuperação será feita através de uma nova entrega do projeto final levando em consideração a correção do projeto original e seguido de uma entrevista com o docente a ser marcada entre os dias 13/09/2021 e 17/09/2021. A sua nota será utilizada para compor a o conceito pós-recuperação C_R conforme as equações abaixo:

$$N_R = \frac{P_R + N_F}{2}$$

Caso 1 $C_F = D$**:**

$$C_R = \begin{cases} \mathbf{C}, & \text{se } N_R \geq 6,0 \\ \mathbf{D}, & \text{caso contrário} \end{cases}$$

Caso 2 $C_F = F$**:**

$$C_R = \begin{cases} \mathbf{D}, & \text{se } N_R \geq 5,0 \\ \mathbf{F}, & \text{caso contrário} \end{cases}$$

9.2. Regulamentações Relevantes

- Resolução ConsEPE Nº 240 - Estabelece a autorização para a oferta excepcional de componentes curriculares e de outras atividades acadêmicas remotas durante o(s) chamado(s) "Quadrimestre(s) Suplementar(es)"
- Resolução ConsEPE nº 182 - Regulamenta a aplicação de mecanismos de recuperação
- Código de Ética da UFABC
- Resolução ConsUni nº 63 - Regimento Geral da UFABC

10. Recursos Online

10.1. Grupos, listas, páginas, ...

- Grupo de Estudos em Haskell da UFABC
  - **Não esqueça de participar do Discord**: https://discord.gg/JSgnfdE

- Haskell Home Page
- #haskell IRC channel
- StackOverflow
- A lista Haskell-beginners é um bom lugar para buscar respostas para perguntas básicas
- A lista Haskell-cafe é boa para buscar respostas a perguntas mais elaboradas
- O Haskell Wiki tem uma lista de respostas às perguntas mais comuns

10.2. Disciplinas em Haskell

- MCTA016-13: Paradigmas de Programação (em Haskell), UFABC. 2020, 2019, 2018.
- CR062-Programação Funcional em Haskell, UFABC. 2019, 2018.
- Estruturas de Dados Puramente Funcionais. 2019.
- G51PGP: Programming Paradigms, University of Nottingham. 2019.
- CS653: Functional Programming, Indian Institute of Technology Kanpur. 2018.
- CIS 194: Introduction to Haskell, University of Pennsylvania. 2016, 2015, 2014, 2013.
- ... mais exemplos aqui

10.3. Leituras

- Haskell Wikibook
- A Gentle Introduction to Haskell
- Haskell Cheat Sheet
- What I Wish I Knew When Learning Haskell

10.4. Documentação

- Standard library documentation
- Hackage - Repositório de pacotes
- Hoogle - Procurando funções para as quais você não sabe o nome?
- Hayoo - Parecido com o Hoogle, porém mais completo (procura todo o Hackage, mas está frequentemente fora do ar...)
- Se você realmente precisar, a linguagem e a biblioteca padrão do Haskell são definidos pelo Haskell 2010 - Language Report

---

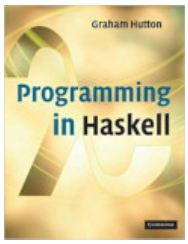
11. Bibliografia

Os principal texto utilizado neste curso será o **GH Segunda Edição**.

[GH]

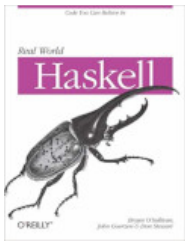
- ***Programming in Haskell***. 2nd Edition.
  - Por *Graham Hutton*.

A primeira edição, que tem boa parte do conteúdo da segunda edição, está disponível na biblioteca:



1st Edition (antiga)

- Link Biblioteca: http://biblioteca.ufabc.edu.br/index.php?codigo_sophia=15287



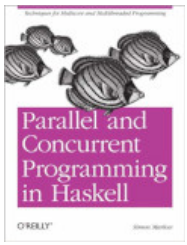
[SGS]

- ***Real World Haskell***.
  - Por *Bryan O'Sullivan*, *John Goerzen* e *Don Stewart*.
  - Disponível **gratuitamente** em: http://book.realworldhaskell.org/

[ML]

- ***Learn You a Haskell for Great Good!: A Beginner's Guide***.
  - Por *Miran Lipovača*.

- Disponível **gratuitamente** em: http://learnyouahaskell.com/



[SM]

- ***Parallel and Concurrent Programming in Haskell: Techniques for Multicore and Multithreaded Programming***.
  - Por *Simon Marlow*.
  - Disponível **gratuitamente** em: https://www.oreilly.com/library/view/parallel-and-concurrent/9781449335939/

Última atualização: 2021-05-24 09:15
Emacs 27.2 (Org-mode 9.4.5)

---

[1] Pode fazer em grupo? Sim. Grupos de no máximo uma pessoa! 😉