

MC346 Paradigmas de programação 2S2024

Horário

3a: das 16h as 18h

5a: das 16h as 18h

Início das aulas

dia 6/Agosto as 16 horas.

Descrição da disciplina

O objetivo da disciplina é apresentar ao aluno linguagens de programação que diferem de forma significativa das linguagens que ele já conhece: Python, C, e Java. Estas linguagens apresentam um conjunto de conceitos (também chamado de paradigmas) que vão expandir as formas com que um programador pode pensar na solução para um problema.

A disciplina abordará as seguintes linguagens de programação, talvez não nesta ordem:

- [Haskell](https://www.haskell.org/) (<https://www.haskell.org/>), particularmente a implementação GHC 9.X para a parte de linguagens funcionais.
- Alguns tópicos avançados em Python (versão 3.7 ou maior). Embora Python não seja por si

só um exemplo de um novo paradigma de programação, ela nos permitirá discutir outros conceitos de programação em uma linguagem imperativa/tradicional. A disciplina assume que o aluno já sabe Python básico.

- Linguagens da família Lisp: Lisp, Scheme e Closure
- Breves conceitos sobre algumas linguagens recentes como Go e Rust (linguagens que substituem C), Scala e Julia (linguagens no nível de abstração de Python), e talvez ML ou F# (versões menos estritas de linguagens funcionais).

Avaliação

A avaliação sera composta de *tarefas* que serão feitas em aula em 15 minutos, e *projetos* que tem alguns dias para serem feitos e serão submetidos via Google Classroom

Haverá pelo menos 7 tarefas. As tarefas serão nas 3a feiras, provavelmente iniciando as 16:15 e terminando em 15 a 30 minutos. As tarefas são programas curtos nas diferentes linguagens discutidas em aula, e devem ser feitas em papel e entregue em aula. Alguns tarefas serão individuais, e algumas em pares. Isso será indicado na tarefa. Cada tarefa recebe as notas 0 se há mais de um erro ou um erro mais serio, 5 se há apenas um erro de menor impacto, e 10 se não há erros.

Haverá pelo menos 7 projetos. Os projetos são programas mais longos nas diferentes linguagens da disciplina. Projetos terão de 3 a 7 dias (dependendo do projeto) para serem submetidos. Não aceitarei projetos atrasados. Projetos serão individuais.

As duas menores notas das tarefas e as duas menores notas dos projetos serão descartadas. A nota final será a média das notas restantes das tarefas e dos projetos.

Não há substitutiva para as tarefas e os projetos não poderão ser entregues atrasados. Se voce faltar na aula quando há uma tarefa, voce tirará 0 na tarefa, mas note que até dois 0 serão descartados para computar a nota final.

Alunos que tiverem feito pelo menos 4 tarefas e 4 projetos tiverem uma nota final entre 2.5 e 4.9 poderão fazer o exame.

O exame será dia 12/12 no horário de aula.

Observações

Qualquer tentativa de fraude nas tarefas implicará em nota final o (zero) para todos os envolvidos, sem demais implicações. Exemplos de fraudes são:

- Compartilhar trechos de código de qualquer forma.
- Copiar ou comprar uma tarefa.

Referencias

Haskell

Livro texto: Learn you a Haskell for greater good (<http://learnyouahaskell.com/>)

Uma lista de recursos (<https://www.haskell.org/documentation/>) para aprender Haskell

Python

- Não instale a versão 2.7. Usaremos a versão 3.10 ou maior. Uma alternativa para instalar o Python é usar o Anaconda Python (<https://www.continuum.io/downloads>) e talvez melhor ainda é instalar o Miniconda (<https://conda.io/miniconda.html>).
- Instale também o Numpy (<https://www.numpy.org/>) (já vem no Anaconda).
- Documentação do Python (<http://docs.python.org/index.html>)

Closure

- site (<https://clojure.org/>)

ML

- OCaml (<https://ocaml.org/>)
- F# (<https://fsharp.org/>)

Scala

- [site \(https://www.scala-lang.org/\)](https://www.scala-lang.org/)

Lisp

- [um site \(https://lisp-lang.org/\)](https://lisp-lang.org/)

Go

- [site \(https://golang.org/\)](https://golang.org/)

Rust

- [site \(https://www.rust-lang.org/\)](https://www.rust-lang.org/)

Julia

- [site \(https://julialang.org/\)](https://julialang.org/)