DCC024 Linguagens de Programação
2023.1

Informações Gerais
Visão geral do conteúdo

Área de Linguagens de Programação DCC/UFMG

O professor

- **Haniel Barbosa**

  hbarbosa@dcc.ufmg.br `https://homepages.dcc.ufmg.br/~hbarbosa/`

- Formação:

  2017: Doutorado em Ciência da Computação (Université de Lorraine, França)
  2012: Mestrado em Ciência da Computação (UFRN)
  2010: Bacharelado em Ciência da Computação (UFRN)

- Experiência profissional:

  2019-...: Professor adjunto (UFMG)
  2017-2019: Professor assistente visitante (University of Iowa, EUA)
  2017-2019: Pesquisador pós-doutor (University of Iowa, EUA)
  2013: Professor substituto (UFRN)
  2012: Estágio (Clearsy, França)
  2010: Estágio (AeS - Acesso e Segurança, Brasil)

- Interesses de pesquisa:

  - métodos formais,
  - automatização de raciocínio lógico,
  - síntese de programas,
  - satisfatibilidade módulo teorias

Bibliografia

- Livro-texto:
  - **Modern Programming Languages: A Practical Introduction**, by Adam Webber
  - Apresentações sobre os capítulos do livro estão disponíveis em: http://www.webber-labs.com/mpl/
- Além do livro-texto, leituras recomendadas serão colocadas na página da disciplina
- Notas e exemplos usados em aula também serão disponibilizados

Métodos de avaliação

- Atividades:
  - **2 Provas**: 60% da nota final.
  - **Projeto**: 30% da nota final
    - Grupos de 2 alunos
  - **Listas de exercícios**: 10% da nota final.

- Haverá uma **prova substitutiva** que:
  - substitui uma prova **perdida** durante o semestre,
  - ocorre ao final do semestre, e
  - cobre toda a matéria lecionada no curso.

Comunicação e monitoria

- Para material didático, informações gerais, e calendários, acesse:

  `https://hanielb.github.io/2023.1-lp/`

  e também o Moodle da disciplina:

  `https://virtual.ufmg.br/20231/course/view.php?id=8740`

  - Grupos de discussões e avisos urgentes (como eventuais cancelamentos de aula de última hora) também ocorrem no Moodle da disciplina.

- E-mails sobre a disciplina devem iniciar o campo "assunto" / "*subject*" com o indicativo **[LP]** para facilitar a organização das mensagens.

Visão geral do conteúdo

Princípios

- Conceitos que caracterizam linguagens de programação:
  - Sintaxe
  - Semântica
  - Nomes
  - Tipos
  - Abstrações
- Para qualquer linguagem:
  - seus criadores devem definir estes conceitos
  - seus programadores devem dominar estes conceitos

Sintaxe

- A *sintaxe* de uma linguagem de programação é a descrição precisa de *todos* os seus programas gramaticamente corretos

- Quando estudamos sintaxe, nos perguntamos:
  - Qual é a gramática da linguagem?
  - Qual é o vocabulário?
  - Como erros de sintaxe são detectados?

Semântica

- O *significado* de um programa é definido pela *semântica* de sua linguagem.

- Ao estudar semântica, nos perguntamos:
  - Quando um programa é executado, o que acontece com os valores de suas variáveis?
  - O que cada elemento do programa faz?
  - Que modelo rege a execução, por exemplo com a chamada de uma função?
  - Como variáveis e objetos são alocadas na memória durante a execução?

Nomes

- Vários tipos de entidades em programas possuem *nomes*:
  - variáveis, tipos, funções, parâmetros, classes, objetos...

- Entidades nomeadas em programas são restritas de acordo com:
  - escopo
  - visibilidade
  - tipo
  - tempo de vida

Tipos

- Um *tipo* é uma coleção de valores e de operações sobre esses valores
  - Tipos simples
  - Tipos estruturados

- O *sistema de tipos* de uma linguagem pode ajudar a:
  - determinar que operações são permitidas
  - identificar erros de tipagem
  - otimizar certas operações

Abstrações

- Mecanismo para *generalização* de dados ou de computação:
  - Procedimentos / funções
  - Módulos
  - Tipos de dados abstratos
  - Classes
  - Modelos de memória

Paradigmas de programação

- Um *paradigma de programação* é um padrão de construção de soluções que permeia um dado grupo de programas e linguagens

- Existem diversos paradigmas de programação:
  - Imperativo
  - Orientado a objeto (OO)
  - Funcional
  - Lógico
  - ...

Paradigma imperativo

- Segue o clássico modelo von-Neumann:
  - Programa e dados são indistinguíveis na memória
  - Programa: sequência de comandos modificando um *estado* atual
  - Estado: valores de todas as variáveis quando o programa é executado
  - Programas maiores usam abstração através de procedimentos

- Exemplos de linguagens imperativas:

  C, C++...

Paradigma orientado a objeto (OO)

- Um programa OO é uma coleção de objetos que interagem trocando mensagens que modificam o estado atual

- Principais propriedades:
  - Encapsulamento de estado
  - Troca de mensagens
  - Herança
  - Subtipagem

- Exemplos de linguagens OO:

  C++, Java, Python...

Paradigma funcional

- Programação funcional modela computação como uma coleção de funções (matemáticas)
  - Entrada : domínio
  - Saída : imagem
- Principais propriedades:
  - Composição
  - Recursão
  - Transparência referencial
- Exemplos de linguagens funcionais:

  Standard ML, Lisp, Haskell, F#, ...

Paradigma lógico

- Programação lógica declara que resultado o programa *deve* ter em vez de *como* obtê-lo

- Principais propriedades:
  - Programas como conjuntos de restrições a um problema
  - Computação de todas as soluções possíveis
  - Computação não-determinística

- Exemplos de linguagens para programação lógica:

  Prolog, SMT-LIB...

Conteúdos do curso

- O curso consistirá em cobrir conceitos de linguagens de programação no contexto dos paradigmas:
  - Funcional (utilizando ML)
  - Imperativo / OO (utilizando Python, C++ e outras)
  - Lógico (utilizando Prolog)
- Introduções a estas linguagens serão feitas conforme cobrimos conceitos relevantes aos respectivos paradigmas
- Análises sintática e semântica serão pervasivos durante o curso

Projeto

- O curso terá um projeto abordando:
  - Verificação de tipos de uma linguagem dada
  - Execução de um programa em uma linguagem dada
- Deve ser em duplas
- Implementação em SML
- O projeto objetiva exercitar os conceitos e paradigmas que estudaremos durante o curso.