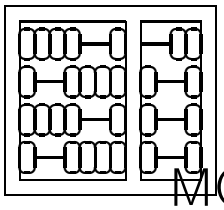

Instituto de Computação
Unicamp

MC-346 – Paradigmas de Programação

2º Semestre de 2016

Programa do Curso

1 Aulas e Atendimento

	Dia	Horário	Sala
Aulas	3ª	21:00-22:40	CB 03
	5ª	19:00–20:40	CB 03
Atendimento	5ª	17:45-18:45	IC 12
Professor	Siome Goldenstein		
PAD	Fábio Sartorato		

2 Critério de Avaliação

A disciplina este semestre utilizará fortemente o Moodle para coordenação, divulgação de informações e entrega de trabalhos. É responsabilidade de cada aluno verificar que possui o devido acesso ao sistema no início do semestre. A autenticação é feita pelo sistema do CCUEC utilizando o LDAP da DAC e não é responsabilidade nem do docente nem do Instituto de Computação. Não é permitido fornecer suas credenciais de acesso ao sistema à outro aluno, isto será tratado como fraude.

Teremos uma nota para Programação Funcional (Fun), uma para Programação Lógica (Log) e uma para prototipagem (Prot). Cada uma delas será a média aritmética de no mínimo três e no máximo cinco mini testes surpresa de 20 minutos (descartando a menor nota). Utilizaremos o recurso de peer-grading do Moodle, onde a nota da atividade é a média ponderada da nota do seu teste (peso 3) e da sua nota da correção dos testes dos colegas (peso 2). Apenas aqueles que participaram do teste poderão corrigí-lo.

A nota (N) é calculada como

$$N_1 = 0,2 \times Prot + 0,40 \times Log + 0,4 \times Fun,$$

e a nota pré-exame é

$$N_2 = \begin{cases} & Prot < 2,5 \\ \min(4,9, N_1) & Log < 2,5 \\ & Fun < 2,5 \\ N_1 & \text{caso contrário.} \end{cases}$$

Alunos com $N_2 < 2,5$ estão reprovados por nota, sem direito a exame. Alunos com $2,5 \leq N_2 < 5,0$ precisam fazer o exame (E). A fórmula da nota final (NF) é

$$NF = \min\left(\frac{N_1 + E}{2}, \max(N_1,\ 5)\right).$$

O exame será ministrado na terça-feira 20/12/2016, no mesmo horário e sala das aulas regulares.

Qualquer caso de fraude ou comportamento anti-ético implicará em média zero na disciplina para todos os envolvidos. Isto é critério de avaliação e não penalidade, logo não impede atitudes posteriores por parte de outras instâncias da Universidade.

3 Programa

1. Programação rápida (prototipação):
   (a) Programação interativa.
   (b) Variáveis não tipadas.
   (c) Expressões regulares.
   (d) Listas, matrizes e dicionários.
2. Programação funcional:
   (a) Funções e recursão.
   (b) Ausência de atribuições explícitas.
   (c) Funções como argumentos de funções.
   (d) Funções anônimas.
3. Programação lógica:
   (a) Fatos e regras.
   (b) Inferência e unificação.
   (c) Retrocesso.
   (d) Modificação dinâmica de programa.

4 Bibliografia

1. **SICP** — Struct. and Interp. of Computer Progs. Abelson, Sussman, and Sussman. MIT Press, 1992.
2. **Clocksin** — Programming in Prolog. Clocksin and Mellish. Springer, 2003.
3. **Python** — Learning Python, 5^{th} Edition. Mark Lutz. O'Reilly Media, 2013.

Outras Referências

1. **PLAL** — Programming Languages: An Active Learning Approach. Kent D. Lee. Springer, 2008.
2. **LSchm** — The Little Schemer , 4^{th} Edition, Daniel P. Friedman and Matthias Felleisen. The MIT Press, 1995.
3. **LAMBDA** — An Intro. to Func, Prog. through Lambda Calculus. Greg Michaelson. Dover, 2011.
4. **ArtPROLOG** — The Art of Prolog: Advanced Programming Techniques. Leon Sterling and Ehud Shapiro. The MIT Press, 1986.
5. **CrftPROLOG** —The Craft of Prolog. Richard O'Keefe. Leon Sterling and Ehud Shapiro. The MIT Press, 2009.
6. **LAMBDA** — An Intro. to Func, Prog. through Lambda Calculus. Greg Michaelson. Dover, 2011.
7. **PLPP** — Prog. Languages: Principles and Paradigms. Gabbrielli and Martini. Springer, 2010.