Plano de Desenvolvimento da Disciplina

Identificação

Disciplina: MC346 – Paradigmas de Programação
Período: 2º semestre de 2017
Instrutor: João Meidanis

Missão

Nesta disciplina, nossa missão é dar aos alunos a oportunidade de conhecer e utilizar com desenvoltura outros paradigmas de programação diferentes daqueles aprendidos nas disciplinas introdutórias. Isto permitirá que tenham um arcabouço maior de recursos para resolver problemas em suas carreiras profissionais, o que é uma vantagem significativa num mundo com grandes e rápidas mudanças como o de hoje.

Visão

A disciplina está dividida em três partes, cada uma abordando um paradigma diferente de programação.

- Paradigma **declarativo** (ou **lógico**), através da linguagem **Prolog**.
- Paradigma **funcional**, através da linguagem **Haskell**.
- Paradigma **orientado a objetos**, através da linguagem **Python**.

Para melhor atingir os objetivos da disciplina, os recursos disponíveis na sala, como possibilidade de acesso a computador remoto, onde estarão instalados interpretadores ou compiladores das linguagens estudadas, serão usados para testar código ou observar estilos de programação.

A nota dos alunos será composta de duas partes. Uma das partes será constituída das provas individuais, onde os alunos deverão basicamente desenvolver trechos de código ou responder a questões teóricas a respeito dos paradigmas e linguagens apresentados em aula. A outra parte corresponde a projetos de programação, onde os alunos desenvolverão código. Este desenvolvimento permitirá aos alunos avançarem em suas habilidades de programação e pensamento lógico.

Avaliação

Os alunos serão avaliados em cada uma das linguagens cobertas no semestre e receberão uma nota específica para cada linguagem. A nota de aproveitamento será calculada em função das notas em cada linguagem. Caso a nota de aproveitamento seja menor que 5,0 (cinco), a aluna ou aluno deverá fazer o exame final para obter média suficiente para aprovação.

Em cada linguagem, haverá dois tipos de avaliação: a prova escrita e o projeto de programação. A prova escrita valerá 2/3 da nota da linguagem, enquanto que o projeto de programação dará os outros 1/3 da nota.

As provas escritas terão duração de 1:40 horas, sem consulta. Haverá três provas escritas, uma para cada linguagem.

Os projetos de programação consistem no desenvolvimento de programas para resolver problemas específicos. Para este semestre, os alunos deverão escrever código para o jogo conhecido como Master Mind, em cada uma das linguagens abordadas.

Cada projeto deverá ser entregue na data designada no cronograma. Projetos entregues com atraso perderão pontos, a uma taxa de 0.0138889% por minuto de atraso, que equivale a uma taxa de 20% da nota por dia. Por exemplo, um projeto entregue com 36 horas (2160 minutos) de atraso perderá 30% da nota. Um atraso de 5 minutos perderá apenas 0.07%, praticamente desprezível. Após 5 dias, não vale mais a pena entregar o projeto.

Os códigos dos alunos serão analisados para ver se contém comandos espúrios e maliciosos, como tentativa de acesso a arquivos que não fazem parte do sistema sendo desenvolvido, tentativa de acesso à rede, etc. Quaisquer arquivos fonte que contiverem comandos considerados maliciosos serão descartados e o aluno que os tiver submetido receberá **zero na disciplina** como punição. Programas iguais ou muito parecidos serão passíveis da mesma punição. Além disso, cada aluno deverá ser capaz de explicar todo o seu código ao instrutor, em entrevista marcada especialmente para este fim. Caso o aluno não demonstre cabal conhecimento de seu próprio código, receberá também **zero na disciplina**.

A nota de aproveitamento será calculada com base nas notas de cada linguagem. Se a nota em alguma linguagem for menor que 4,0 (quatro), a nota de aproveitamento será igual à menor nota dentre as notas de linguagens. Se todas as notas de linguagens forem maiores ou iguais a 4,0 (quatro), então a nota de aproveitamento será a média aritmética das notas de linguagens. Com isto, para escapar do exame, os estudantes deverão obter nota de aproveitamento maior ou igual a 5.0 (cinco), e também uma nota maior ou igual a 4.0 (quatro) em **cada linguagem**.

Desta forma, a nota final de cada estudante será dada por:

Nome	Símbolo	Fórmula
Nota final	NF	(NA + NE)/2, se fez exame NA, caso contrário
Exame	NE	nota obtida no exame final (de 0 a 10)
Aproveitamento	NA	NPr/3 + NH/3 + NPy/3, se NPr ≥ 4 e NH ≥ 4 e NPy ≥ 4 mínimo(NPr, NH, NPy), se NPr < 4 ou NH < 4 ou NPy < 4
Nota de Prolog	NPr	(2*PePr + PrPr)/3
Nota de Haskell	NH	(2*PeH + PrH)/3
Nota de Python	NPy	(2*PePy + PrPy)/3
Prova escrita de Prolog	PePr	nota obtida na prova escrita sobre Prolog (de 0 a 10)
Prova escrita de Haskell	PeH	nota obtida na prova escrita sobre Haskell (de 0 a 10)
Prova escrita de Python	PePy	nota obtida na prova escrita sobre Python (de 0 a 10)
Nota do programa de Prolog	PrPr	nota obtida no projeto de programação Prolog (de 0 a 10).

Nota do programa de Haskell	PrH	nota obtida no projeto de programação Haskell (de 0 a 10).
Nota do programa de Python	PrPy	nota obtida no projeto de programação Python (de 0 a 10).

Caso NF seja maior que 10, será rebaixada para 10. Caso seja detectada **fraude**, a nota NF será zero para todos os envolvidos, sem prejuízo de outras sanções.

Cronograma

Atenção: O Congresso de Iniciação Científica da Unicamp ocorrerá durante o dia e as nossas aulas são à noite. Portanto, este congresso não afetará o nosso cronograma.

MC346 A Paradigmas de Programação 2o. Sem/2017
Instrutor: João Meidanis
CRONOGRAMA PRELIMINAR

Data	Aulas Teóricas	Data	Eventos / Campeonatos
01-08-2017	Apresentação. SECOMP.		
03-08-2017	Apresentação. SECOMP.		
08-08-2017	Prolog: primeiros programas, interpretador.		
10-08-2017	Prolog: fatos, regras, perguntas, variáveis.		
15-08-2017	Prolog: sintaxe, unificação, aritmética.		
17-08-2017	Prolog: estruturas, funtores, aridade, listas.		
22-08-2017	Prolog: recursão, acumuladores; predicados básicos.		
24-08-2017	Prolog: ressatisfação e corte; predicados do sistema.		
29-08-2017	Prolog: entrada, saída, arquivos, operadores.		
31-08-2017	Prolog: gramáticas.	30-08-2017	**Prolog: início submissões**
05-09-2017	Prolog: modelo de execução, depuração.		
07-09-2017	*Não haverá atividades*		
12-09-2017	**Prolog: Prova**		
14-09-2017	Haskell: haskell.org, learn you a Haskell		
19-09-2017	Haskell: funções, listas, tuplas		
21-09-2017	Haskell: tipos e typeclasses	20-09-2017	**Prolog: término submissões (23:59)**
26-09-2017	Haskell: definindo funções		
28-09-2017	Haskell: recursão e projeto	27-09-2017	*Desistência de Matrícula em Disciplina*
03-10-2017	Haskell: funções de ordem superior		
05-10-2017	Haskell: módulos	04-10-2017	**Haskell: início submissões**
10-10-2017	Haskell: definindo novos tipos		
12-10-2017	*Não haverá atividades*		
17-10-2017	*Reunião de Avaliação do Curso*	17-10-2017	*Reunião de Avaliação do Curso*
19-10-2017	Haskell: entrada e saída	19-10-2017	*Trancamento de Matrícula*
24-10-2017	**Haskell: prova**		
26-10-2017	Python: tutorial em python.org.br	25-10-2017	**Haskell: término submissões (23:59)**
31-10-2017	Python: introdução informal		
02-11-2017	*Não haverá atividades*		
07-11-2017	Python: controle de fluxo	06-11-2017	**Python: início submissões**
09-11-2017	Python: estruturas de dados		
14-11-2017	Python: módulos		
16-11-2017	Python: entrada e saída		
21-11-2017	Python: erros e exceções		
23-11-2017	Python: classes		
28-11-2017	Python: bibliotecas padrão (amosta)	27-11-2017	**Python: término submissões (23:59)**
30-11-2017	**Python: prova**		
05-12-2017	Semana de estudos.		
07-12-2017	Semana de estudos.		
12-12-2017	**Exame**		
14-12-2017			

Bibliografia

Adotaremos os seguintes textos neste semestre:

Programming in Prolog, de William F. Clocksin e Christopher S. Mellish, 5a. edição, Springer, 2003.

Learn You a Haskell for Great Good! A Beginner's Guide, de Miran Lipovača, No Starch Press, Abril 2011, 400 pp.

O tutorial de Python, Release: 2.7, Data: 15 de outubro de 2012, por Rodrigo Senra, Pedro Werneck e outros. `http://turing.com.br/pydoc/2.7/tutorial/index.html`