Especificação do Trabalho Final

O **objetivo** deste trabalho consiste em fornecer aos alunos a oportunidade de conhecer uma linguagem moderna e que agregue características multiparadigma, combinando *orientação a objetos* com *características funcionais*. Permitirá ainda com que os alunos demonstrem que aprenderam os princípios de programação relacionados com os diferentes paradigmas estudados, além de oportunizar demonstrarem que são capazes de analisar e avaliar linguagens de programação seguindo os critérios estudados em aula.

1 Introdução

Cada grupo, formado de **três pessoas**, deve **selecionar uma linguagem entre as seguintes opções**: Commom Lisp, C++ (versão 11), C#, F#, JavaScript, Objective C, Objective CAML, Perl, PHP, Python, Ruby, Scala ou Scheme. Tais linguagens foram selecionadas por apresentarem características funcionais e orientadas a objeto. Podem ser utilizadas outras linguagens, desde que possuam princípios de programação orientada a objetos combinados com programação funcional. Caso o grupo opte por escolher outra linguagem, antes de fechar esta escolha, converse com o professor.

2 Especificação do Problema

Cada grupo deve **especificar um problema a ser resolvido com a linguagem escolhida**. O objetivo é demonstrar como as funcionalidades orientadas a objeto e funcionais da linguagem podem ser usadas de maneira conjunta na solução do problema. Caso o grupo opte por escolher um problema que não está na lista de sugestões abaixo, o mesmo deve encaminhar sua ideia ao professor, descrita em detalhes, que avaliará sua viabilidade na realização deste trabalho final.

2.1 Sugestão de Problemas

Sugerem-se os seguintes problemas:

2.1.1 Jogo de torre de defesa

O objetivo deste tipo de jogo é tentar parar os inimigos de cruzar um mapa através da construção de prédios-armadilhas e torres que atiram projéteis cinéticos. As construções devem ser feitas de forma a diminuir o avanço dos inimigos. Mais informações sobre este tipo de jogo podem ser obtidas na Wikipedia (`http://en.wikipedia.org/wiki/Tower_defense/`).

2.1.2 Simulador de galáxias ou de partículas

Implementar um simulador de partículas, considerando forças físicas de repulsão e atração. Uma possibilidade é utilizar as leis gravitacionais para construir um simulador de órbitas para estrelas e planetas. Outra possibilidade é utilizar uma força elétrica de repulsão (todas as partículas com carga positiva, por exemplo), e forças de atração baseadas em molas. Deve-se ter cuidado com a escalabilidade do algoritmo utilizando, dando preferências para o algoritmo de Barnes-Hut. Um exemplo utilizando a linguagem C já está disponível em `http://github.com/schnorr/viva/tree/master/src/libtupi`, e pode ser utilizado como inspiração para a modelagem.

2.1.3 Jogo de tática por turnos

O funcionamento deste tipo de jogo é a separação das jogadas em turnos. Em cada jogada, os jogadores criam e manifestam suas táticas de defesa ou ataque em um ambiente multi-jogador ou jogador/computador. A simulação destas táticas ocorre de forma não-simultânea, em ambientes de guerra e históricos. Mais informações podem ser também obtidas na Wikipedia (`http://en.wikipedia.org/wiki/Turn-based_tactics`)

2.1.4 Jogos de artilharia em duas (ou três) dimensões

Nestes jogos, o combate é feito por turnos onde o objetivo de cada jogador é destruir o seu adversário através do planejamento de uma trajetória correta para um projétil cinético. Técnicas sofisticadas de cálculo balístico, envolvendo força, vento, etc, podem ser empregadas para facilitar ou dificultar o jogo. Mais informações (`http://en.wikipedia.org/wiki/Artillery_game`)

2.1.5 Jogo do tipo War

Segundo a Wikipedia (`http://pt.wikipedia.org/wiki/War`):

> *O* jogo é disputado com um mapa do mundo dividido em 6 regiões (Europa, Ásia, África, América do Norte, América do Sul e Oceania). Cada jogador recebe uma carta com um determinado objetivo e quem completar primeiro o seu e declará-lo cumprido é o vencedor. É disputado em rodadas, nas quais os participantes colocam exércitos (as bolinhas menores, vistas nas imagens abaixo) e atacam outros oponentes.

3 Escolha do Problema

O problema e a linguagem escolhidos, assim como os membros de grupo devem ser informados ao professor através do fórum apropriado disponível no moodle (aba Trabalho Final). Para tal, o líder do grupo deve criar um novo tópico cujo título deve ser o identificador do problema e a linguagem escolhida. Os membros do grupo devem ser informados na mensagem do tópico.

4 Recursos Necessários

A solução implementada deve usar os recursos especificados nesta seção. Caso um recurso não esteja disponível na linguagem, explique e justique os motivos e utilize um mecanismo alternativo. Esta escolha deve estar presente no relatório. Segue a lista de recursos necessários e que devem estar presentes na solução do grupo.

- Definição e uso de classes (sejam utilitárias ou para representar as estruturas de dados utilizadas pelo programa)
- Encapsulamento e proteção dos atributos, com os devidos métodos de manipulação (setters/getters).
- Especificação de construtores-padrão para a inicialização dos atributos e, quando possível, construtores
- Especificação de destrutores (ou métodos de clean-up), se necessário.
- Organização do código em espaços de nome diferenciados, conforme a função ou estrutura de cada classe
- Mecanismo de herança, em especial:
  - especificação de pelo menos três níveis de hierarquia, sendo uma classe mais genérica (e abstrata, ver a seguir)
  - especificação de pelo menos uma classe abstrata (ou interface) a ser especializada (ou implementada) nas classes filhas
  - demonstração de polimorfismo por inclusão (variável ou coleção genérica manipulando entidades de classes filhas, chamando métodos ou funções específicas correspondentes)
- Polimorfismo paramétrico:
  - especificação de algoritmo (método ou função genérico) utilizando o recurso oferecido pela linguagem (p. ex., generics, templates)
  - especificação de estrutura de dados genérica utilizando o recurso oferecido pela linguagem (p. ex., generics, templates)
- Polimorfismo por sobrecarga (vale construtores alternativos)
- Especificação e uso de funções como elementos de 1ª ordem
- Especificação e uso de funções de ordem maior (map, reduce, foldr/foldl ou similares)
- Uso de listas para a manipulação de estruturas, entidades e elementos em funções de ordem maior. As funções devem ser puras, ou seja, devem criar novas listas (e não manipular as que são recebidas)
- Especificação e uso de funções não nomeadas (ou lambda)
- Especificação e uso de funções que usem currying
- Especificação de funções que utilizem pattern matching na sua definição
- Uso de recursão como mecanismo de iteração (pelo menos em funções de ordem maior que manipulem listas)
- Especificação e uso de delegates

5 Relatório

O grupo deve apresentar um relatório técnico com os itens descritos abaixo. Sugere-se que este relatório seja escrito utilizando recursos do LaTeX. Segue a lista dos itens obrigatórios.

1. **Capa**. Com identificação do grupo, da linguagem escolhida e do problema.
2. **Problema**. Apresentação do problema solucionado, descrevendo quais são os requisitos e as funcionalidades implementadas. Apresentação dos frameworks, bibliotecas e ferramentas utilizados no desenvolvimento.

3. **Linguagem**. Apresentação da linguagem, descrevendo suas características, fundamentos, funcionalidades, aplicações (aplicabilidade em questões práticas), benefícios, problemas, limitações.

4. **Implementação**. Detalhamento da implementação, focando na descrição de como os recursos descritos acima (na seção anterior) foram usados para resolver o problema escolhido. Insira trechos de código que demonstrem onde cada recurso foi usado, informando em qual arquivo fonte e linha estes recursos aparecem.

   Explicar vantagens ou desvantagens, dificuldades ou facilidades que o seu uso trouxe para a solução do problema. Se possível, fazer paralelo com outra linguagem que não implemente o recurso em questão indicando a dificuldade que seria resolver o problema sem o recurso (ou indicar uma forma alternativa).

5. **Análise Crítica**. Uma análise crítica da linguagem, envolvendo uma tabela com os critérios e propriedades estudados em aula (i.e. simplicidade, ortogonalidade, expressividade, adequabilidade e variedade de estruturas de controle, mecanismos de definição de tipos, suporte a abstração de dados e de processos, modelo de tipos, portabilidade, reusabilidade, suporte e documentação, tamanho de código, generalidade, eficiência e custo), com notas/valores justificados (ou seja, dar exemplos de código ou de situações que contariam como pontos favoráveis ou desfavoráveis para cada critério ou propriedade).

6. **Conclusões** Conclusões, descrevendo as facilidades e dificuldades encontradas

7. **Referências** Referências (livros, artigos, páginas na Internet, ...)

6 Boas Práticas de Implementação

O professor sugere uma lista de boas práticas para a implementação deste trabalho. Elas são opcionais. Caso o grupo tenha interesse em utilizá-las, o professor pode auxiliar no seu uso durante o semestre.

Git para manter o histórico de modificações, por exemplo, utilizando o serviço do GitHub ou Bitbucket.

CMake para manter o ambiente de compilação e os arquivos Makefile

7 Forma da Entrega

A entrega do trabalho é realizada através de um arquivo compactado (em formato .tar.gz ou .zip) com:

- O relatório técnico do trabalho em formato PDF
- Código-fonte e os Makefiles para compilação

8 Apresentação

Os resultados dos trabalhos serão apresentados ao professor e à turma, em datas definidas no cronograma da disciplina disponibilizado na plataforma de apoio pedagógico.

9 Avaliação

A avaliação do trabalho incluirá os seguintes critérios: desenvolvimento e detalhamento dos itens do relatório, aplicação dos conceitos de programação estudados, utilização correta dos recursos da linguagem escolhida, correção, legibilidade, confiabilidade e originalidade, uso de referências, formatação e estilo do texto. Outros aspectos de avaliação poderão ser incluídos a critério do professor. O peso deste trabalho corresponde ao valor especificado no plano da disciplina disponível na plataforma de apoio pedagógico.

10 Dúvidas

Em caso de dúvidas, não hesite em consultar o professor.