INF1301 Programação Modular
Período: 2019-1
Prof. Flavio Bevilacqua
3o. Trabalho
Data de divulgação: 20 de maio (segunda-feira)
Data de entrega: 17 de junho (segunda-feira)

1. Descrição do trabalho do período

O objetivo do trabalho do período é implementar um jogo de Gamão para ser jogado por duas pessoas. As regras do Gamão podem ser facilmente encontradas na Internet.

A aplicação ao final do terceiro trabalho deverá conter os seguintes módulos:

Tabuleiro – para a estrutura do tabuleiro, este módulo deverá se acoplar ao módulo de Lista Duplamente Encadeada Genérica com Cabeça. Cada casa do tabuleiro será um nó desta lista. Como mais de uma peça da mesma cor pode estar armazenada em uma casa do tabuleiro, o Nó da casa do tabuleiro armazena uma Lista em que cada nó armazena uma peça que esteja posicionada nesta casa. Em resumo, o tabuleiro com as peças será uma lista de listas.

Peça – Tipo abstrato de dados que encapsula uma estrutura Peça. Aqui pode se criar, destruir, obter cor e qualquer outra ação que o grupo necessitar. (importante: Cuidado com as atribuições de cada módulo. O módulo que visualiza a posição de cada peça é o Tabuleiro. Não é atribuição do módulo Peça movimentar a peça).

DadoPontos – Tipo abstrato de dados que encapsula uma estrutura DadoPontos (aquele que tem em seus lados o valor 2, 4, 8, 16, 32 e 64). Este módulo também deve armazenar quem é o jogador que pode dobrar o valor da partida no momento, informar a pontuação do jogo e qualquer outra ação que o grupo achar necessária dentro das atribuições desta entidade.

PeçasCapturadas (BAR) – Tipo abstrato de dados que encapsula uma lista de estruturas Peça capturadas de cada jogador. Este módulo utiliza o módulo de Lista Duplamente Encadeada Genérica com Cabeça e é acionado ao longo do jogo para entradas e saídas de peças capturadas e retornadas ao tabuleiro.

PeçasFinalizadas – Tipo abstrato de dados que encapsula uma lista de estruturas Peça capturadas de cada jogador. Este módulo utiliza o módulo de Lista Duplamente Encadeada Genérica com Cabeça e é acionado ao longo do jogo para entradas de peças que tenham chegado ao seu destino final.

Dado – Módulo responsável por gerar valores aleatórios para dois dados sempre que acionado. Será utilizado por cada jogador nas várias rodadas da partida.

Jogo – Módulo Principal que administrará todos os outros para a realização da partida, informará quantos pontos o jogador ganhou, gravará e recuperará jogos e implementará as regras do gamão para verificar se tudo está sendo operado de forma correta pelos dois jogadores. Este módulo disponibilizará uma interface ASCII apresentando o tabuleiro e todas as ações definidas pelo grupo para que uma partida se realize.

Obs: O grupo pode criar outros módulo conforme a necessidade.

2. Descrição do terceiro trabalho

O objetivo do trabalho 3 é completar a aplicação Gamão. Cada módulo implementado nesta etapa deverá ser testado separadamente de maneira completa utilizando o Arcabouço de Teste Automatizado da Disciplina gerando um projeto para cada, com o respectivo executável e script correspondente.

Passos a serem seguidos neste trabalho 3.

1. Mantenha atualizado o documento que contém a **Especificação de Requisitos em PDF** de **TODO O TRABALHO DO PERÍODO**.

2. Mantenha atualizada a arquitetura do programa em PDF, apresentando um diagrama contendo toda a modularização elaborada pelo grupo de forma que atenda plenamente o jogo. Novamente, isto se refere a **TODO O TRABALHO DO PERÍODO**. Este diagrama apresentará cada módulo necessário, seus inter-relacionamentos, e as funções disponibilizadas em cada interface.

3. Mantenha atualizado o modelo estrutural do trabalho em PDF. É importante que haja apenas UMA estrutura para cada jogo, ou seja, a lista de listas do tabuleiro, a lista das peças capturadas, a lista das peças finalizadas e qualquer outra estrutura que o grupo achar necessário deverá estar ancorada em um Cabeça da Partida. A utilização desta estrutura permitirá gravar o estado atual do jogo para continuar depois. Ao gravar, esta estrutura principal será toda armazenada em um arquivo-texto. Ao recuperar, esta estrutura principal será toda remontada a partir de um arquivo-texto.

4. Elabore e teste **separadamente** os módulos **que não foram testados no trabalho 2**. Cada módulo deve prever tudo que é possível fazer com cada estrutura. O grupo tem que realizar uma análise bem abrangente para relacionar tudo que cada módulo precisa e, principalmente, sem misturar atribuições de módulos (por exemplo: andar com peça no Módulo Peça ou indicar vencedor no módulo tabuleiro).

Obs: como a Lista faz parte da implementação, o módulo tem que estar previsto na arquitetura e no modelo estrutural.

3. Entrega do Trabalho

O trabalho deve ser feito em grupos de dois ou três alunos. Os programas devem ser redigidos em "**C**". Não será aceita nenhuma outra linguagem de programação. Todos os programas devem estar em conformidade com os padrões dos apêndices de 1 a 10 do livro-texto da disciplina. Em particular, os módulos e funções devem estar devidamente especificados.

Recomenda-se fortemente a leitura do exemplo e do texto explanatório do arcabouço. Além de mostrar como implementar um teste automatizado dirigido por script, ilustra também as características de um programa desenvolvido conforme os padrões do livro.
O trabalho deve ser enviado por e-mail em um único arquivo **.zip** (codificação do attachment: **MIME**). Veja os Critérios de Correção de Trabalhos contidos na página da disciplina para uma explicação de como entregar.

O arquivo **.zip** deverá conter:

- Documento atualizado em PDF com Especificação de Requisitos, Arquitetura do Programa, Modelo Estrutural conforme notação UML apresentada na aula de Modelagem (contendo o Modelo, um Exemplo da estrutura com base no modelo e as respectivas assertivas estruturais).

- Documento em PDF com o script de teste manual da aplicação Gamão

- Manual de Usuário da Aplicação Gamão

- Os arquivos-fonte dos diversos módulos que compõem o programa.

- Os arquivos de script de teste completos desenvolvidos pelo grupo (para os novos módulos que ainda não haviam sido testados no trabalho2)

- Os arquivos batch (**.bat**) que coordenam a execução dos testes.

- programa executável (construto) em cada projeto: **TRAB3-1.EXE (Aplicação do Gamão)**, **TRAB3-2.EXE, TRAB3-3.EXE, etc (um para cada módulo restante a ser testado)**. Caso tenha havido modificação do módulo lista, é necessário um projeto para testá-lo separadamente também. **Deverão existir vários projetos (cada um com sua pasta específica e tudo que é necessário para a geração do executável respectivo) dentro do ZIP a ser entregue. Se esta organização não for respeitada, perde-se dois pontos no trabalho.**

- Um arquivo **LEIAME.TXT** contendo a explicação de como utilizar o(s) programa(s).

- Tantos arquivos **RELATORIO-nome.TXT** quantos forem os membros do grupo. O tema **nome** deve identificar o membro do grupo ao qual se refere o relatório. Estes arquivos devem conter uma tabela de registro de trabalho organizada como a seguir:

Data ; Horas Trabalhadas ; Tipo Tarefa ; Descrição da Tarefa Realizada

Na descrição da tarefa redija uma explicação breve sobre o que o componente do grupo fez. Esta descrição deve estar de acordo com o Tipo Tarefa. Cada Tipo Tarefa identifica uma natureza de atividade que deverá ser discriminada explicitamente, mesmo que, durante uma mesma sessão de trabalho tenham sido realizadas diversas tarefas.
Os tipos de tarefa são:

- estudar
- especificar os módulos
- especificar as funções
- revisar especificações
- projetar
- revisar projetos
- codificar módulo
- revisar código do módulo
- redigir script de teste
- revisar script de teste
- realizar os testes
- diagnosticar e corrigir os problemas encontrados

Observações:

- Dica: Preencha esta tabela de atividades ao longo do processo. NÃO DEIXE PARA ÚLTIMA HORA, POIS VOCÊ NÃO SE LEMBRARÁ DO QUE FEZ TAL DIA, TAL HORA. Com relatórios similares a esse você aprende a planejar o seu trabalho.

- **Importante**: O arquivo **ZIP**, DEVERÁ CONTER SOMENTE OS ARQUIVOS RELACIONADOS A ESTE TRABALHO. Caso o arquivo enviado contenha outros arquivos que os acima enumerados (por exemplo: toda pasta do arcabouço, arquivos **.bak**, arquivos de trabalho criados pelo ambiente de desenvolvimento usado, etc.) o grupo perderá 2 pontos. Gaste um pouco de tempo criando um diretório de distribuição e um **.bat** que copia do diretório de desenvolvimento para este diretório de distribuição somente os arquivos que interessam. Verifique se esta cópia está realmente completa! A mensagem de encaminhamento deve ter o assunto (subject) **INF1301-Trab03-idGrupo** conforme o caso. O tema **idGrupo** deve ser formado pelas iniciais dos nomes dos membros do grupo. O texto da mensagem deve conter somente a lista de alunos que compõem o grupo (formato: número de matrícula, nome e endereço do e-mail). Perde-se 2 pontos caso não seja encaminhado desta forma. Mais detalhes podem ser encontrados no documento Critérios de Correção dos Trabalhos disponível na página da disciplina.

- O programa será testado utilizando o programa compilado fornecido. Deve rodar sem requerer bibliotecas ou programas complementares. O sistema operacional utilizado durante os testes será o **Windows 7**. Assegure-se que a versão do programa entregue é uma versão de produção, ou seja, sem dados e controles requeridos pelo debugger (versão release). **CABE RESSALTAR QUE SE O PROGRAMA FOR EXECUTADO PELO PROFESSOR E NECESSITAR DE BIBLIOTECAS ADICIONAIS DESNECESSÁRIAS, O TRABALHO PERDERÁ 8 PONTOS MESMO TENDO RODADO NA MÁQUINA DOS COMPONENTES DO GRUPO. SUGESTÃO: ENVIE UM EXECUTÁVEL ANTES DO DIA DA ENTREGA PARA SER TESTADO NA MÁQUINA DO PROFESSOR.**
- Haverá perda de 1 ponto por dia de atraso considerando dias úteis de segunda a sábado. A tolerância é até 7h do dia seguinte.

4. Critérios de correção básicos

Leia atentamente o documento Critérios de Correção dos Trabalhos disponível na página da disciplina. Muitas das causas para a perda substancial de pontos decorrem meramente da falta de cuidado ao entregar o trabalho.

Não deixem para a última hora.
Este trabalho dá muito trabalho!