Universidade Federal da Fronteira Sul
Ciência da Computação
Programação I - Prof. Fernando Bevilacqua

Prova 2

Nome: _____ **Data:** 01/07/2014

1. A prova pode ser feita a lápis, porém o professor se dará ao direito de não aceitar reclamações relativas à correção.
2. Início da prova 7h30, término 10h00. Manter celulares desligados!
3. Coloque o seu nome nas folhas de resposta.
4. A compreensão das questões faz parte da prova.

Boa prova!

Questões

Você foi encarregado de fazer a modelagem e arquitetura de um software de desenho CAD. Durante o levantamento de requisitos, você descobriu o seguinte. O sistema inteiro funciona dentro de um plano cartesiano, ou seja, todas as formas possuem uma posição representada por coordenadas (x, y). A forma mais primitiva de desenho é o ponto, que é usado sozinho ou em conjunto para compor e manipular outras formas mais complexas. Por exemplo, uma linha é composta por dois pontos, o triângulo é feito com 3 pontos, etc. Os usuários informaram que precisam, além da forma primitiva ponto, as seguintes formas complexas: quadrado, retângulo, círculo, triângulo e trapésio. Algumas das formas podem ser rotacionadas pelo usuário, como o quadrado e o retângulo; algumas das formas podem ser transfomadas ("aumentadas" ou "encolhidas"), como o círculo e o triângulo. Algumas formas podem ser rotacionadas e transformadas, como o trapésio. A rotação funciona com base em ângulos, ex.: rotacionar a forma `32.5` graus, já a transformação funciona com base em um valor, ex.: aumentar uma forma `3.4` vezes. Por fim, você descobriu que o programa pode ter um número infinito de formas na tela, porém todas devem ser guardadas num mesmo vetor na memória do programa. No futuro, novas formas complexas serão adicionadas por outros desenvolvedores.

1)(3.0) Faça o diagrama UML que descreva a modelagem desse sistema.

2)(2.5) Implemente a classe `GerenciadorAlteracao`. Essa classe é responsável por realizar operações de rotação e transformação em um conjunto de formas que estejam selecionadas na tela em um determinado instante. A classe possui dois métodos. O método `transforma` recebe um vetor de formas como parâmetro (esse vetor por ter qualquer forma da aplicação). Esse método percorre o vetor informado, rotacionando as formas que podem ser rotacionadas e encolhendo as formas que podem ser encolhidas. O método `moveParaOrigem` também recebe um vetor de formas e o percorre movendo todas essas formas para a posição `(0,0)` do plano (origem).

3)(1.5) Implemente o código necessário para criar a classe `OrganizadorPosicao`. Essa classe possui um método chamado `confina`, que recebe como parâmetro uma forma. Esse método deve levantar uma exceção chamada `EPosicaoInvalida` se a posição `x` ou `y` da forma passada for negativa. O método levanta a exceção `EPosicaoForaQuadrante` se a soma da posição `x` e `y` da forma for maior que 300. Por fim, o método levanta a exceção `EPosicaoAngular` se a posição `x` for igual à posição `y` da forma. Faça um pequeno programa de teste que instancie um `OrganizadorPosicao` e uma forma trapésio; o programa deve passar o objeto trapésio para o método `confina`, pegando todas as exceções que podem ser levantadas (para cada exceção pega, imprima na tela o nome dessa exceção).

4)(3.0) Você foi encarregado de implementar uma interface que ajude na depuração da aplicação. O objetivo da interface é obrigar as classes que a implementem a terem um método que retorna um vetor de `Segmento`, que são os lados que compõem aquela forma. Por exemplo, um quadrado terá 4 segmentos, um triângulo terá 3 segmentos, etc. Um `Segmento` é composto de dois pontos (que é a forma mais primitiva de desenho da aplicação). Escreva o código dessa interface. Faça também o código de uma das classes que representa uma forma reta(ex. quadrado, retângulo, etc), fazendo com que essa classe implemente a interface criada por você.