Universidade Federal do Agreste de Pernambuco

AP3

Thaís Alves Burity Rocha

Agenda

- Diretrizes sobre bom projeto de classes
- Uso de teclado com a classe Scanner
- Exercícios extraídos da lista 2

Classes bem projetadas

- Mesmo que não seja explicitamente solicitado...
- A classe deve representar uma entidade ou responsabilidade principal, com significado claro
- A classe não é um programa completo
- Os atributos devem estar alinhados à classe
- Os métodos devem representar ações especializadas
- Projetar construtores sempre que fizer sentido, ainda que a questão não solicite

Entrada padrão

- Classe **Scanner** para ler dados do teclado
  - Também processa String e arquivos
- Incluir antes da declaração da sua classe

```
import java.util.Scanner;
```

- Criar um objeto do tipo Scanner

```
Scanner sc = new Scanner(System.in);
```

- Usar métodos de leitura

```
int inteiro = sc.nextInt();
String palavra = sc.next();
String linha = sc.nextLine();
```

Questão 1

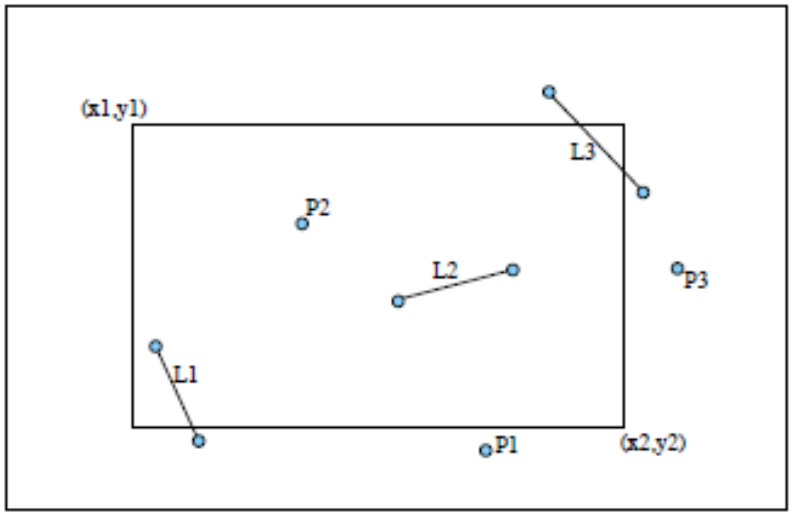
- Escreva uma classe Ponto2D que representa um ponto no plano cartesiano.
- A classe, além de atributos condizentes, deve permitir que a criação de objetos se dê pelos 4 meios a seguir
  - Por default, o ponto deve ser criado na origem do espaço 2D.
  - O ponto pode ser criado num local indicado por dois parâmetros (coordenadas x e y).
  - O ponto pode ser criado num local indicado por apenas um parâmetro. Nesse caso, as coordenadas x e y serão iguais.
  - O ponto pode ser criado em um local indicado por outro ponto. Nesse caso, o novo ponto será criado em posição oposta ao ponto passado como argumento. Por exemplo, o oposto do ponto com coordenadas (5, 8) é o ponto com coordenadas (-5, -8).

Questão 1 (continuação)

- Utilize a entrada padrão para testar seu código.
- Ou seja, o método main deve solicitar do usuário os dados necessários para criar objetos Ponto2D.

Questão 2

- Crie uma classe Retangulo para representar um retângulo cujos pontos extremos sejam 2 instâncias da classe Ponto2D.
- Por exemplo, na imagem (x1; y1) e (x2; y2) são os pontos extremos que definem o retângulo.
- Deve ser possível criar objetos Retangulo a partir de seus dois pontos extremos (dois objetos Ponto2D) ou das coordenadas destes (quatro valores inteiros).



Questão 2 (continuação)

- A classe deve ter um método para verificar se um ponto passado como argumento está localizado dentro de um retângulo. O ponto deve ser representado por uma instância da classe Ponto2D. O método deverá retornar true se o ponto estiver contido no retângulo, e false se não estiver.
- Dica: Verifique se as coordenadas do ponto estão dentro das coordenadas do retângulo. Por exemplo, de acordo com a figura, P1 estaria fora do retângulo, pois sua coordenada y é menor do que a menor coordenada y do retângulo. Já o ponto P2 estaria dentro do retângulo, e o ponto P3 também estaria fora do retângulo.

Questão 3

- Qual é o tipo de associação entre as classes Ponto2D e Retangulo, de acordo com a UML: agregação ou composição? Explique.

- Na sua visão, qual seria a melhor forma de relacionar essas classes? Justifique.

- Observação: As respostas devem ser salvas no arquivo Q3.txt

Questão 4

- Modifique a classe Retangulo para que esta contenha um método calculaIntersecção, que recebe como argumento uma outra instância de Retangulo e que calcula as coordenadas de um retângulo que é a intersecção do retângulo encapsulado com o passado como argumento, retornando uma nova instância da classe Retangulo correspondente à intersecção.
- Dicas: Os pontos do novo retângulo podem ser calculados com regras simples, implementadas através de ifs encadeados. Nem sempre existe intersecção entre dois retângulos.

Questão 4 (continuação)

- Considere a figura: No lado esquerdo existem dois retângulos (mostrados em cores diferentes) que têm intersecção, e, no lado direito, dois que não têm. No caso de não existir intersecção, o método deve retornar null.