Introdução à Linguagem Java

Lista 05

25 de abril de 2024

Reutilização de Classes

Nome do projeto java: ListaX-NUSP. Sendo X o número da lista e NUSP o seu número usp.
Arquivo de envio: . ESTE EXERCÍCIO DEVERÁ SER FEITO EM DUPLA. Entregar um ÚNICO projeto java zipado, ou seja, enviar o arquivo ListaX-NUSP1-NUSP2 zipado.

Seguir as seguintes instruções, sob pena de desconto de nota por código confuso:

- Não enviar arquivos soltos.
- Não enviar mais de um projeto java.
- Não enviar um projeto com muitas subpastas fazendo com que dê trabalho para encontrar o exercício.
- Não enviar rascunhos dos exercícios. Envie apenas os arquivos que vocês efetivamente querem que sejam corrigidos para que não corram o risco de corrigirmos os arquivos errados.
- Colocar cada exercício dentro de um pacote diferente (o pacote fica dentro do projeto java). Exemplo: dentro do projeto Lista04-NUSP, inserir os pacotes ex1, ex2, ex3, ex4 e ex5. Dentro do pacote ex1 poderia haver os arquivos ContaCorrente.java e ContaCorrenteTeste.java.
- Separar a classe funcional da classe de testes deixando-as em arquivos diferentes.

Exercícios

1. Verifique a que a construção automática de classes derivadas funciona, através do seguinte exemplo:

```
class Base {
    Base() {
        System.out.println("Constrói Base");
    }
}
public class Derivada extends Base {
    Derivada () {
        System.out.println("Constrói Derivada");
    }
    public static void main(String []argc){
       Derivada obj = new Derivada();
    }
}
```

   Adicione um parâmetro inteiro ao construtor da classe Base, o que deve gerar uma mensagem de erro. Insira um comentário explicando o erro. Corrija este erro sem alterar novamente a classe Base.

2. Crie uma classe abstrata `NumeroAritmetico` com os seguintes elementos:

   - um atributo `protected long valor`.
   - um método concreto `public final boolean mesmoValor(NumeroAritmetico)`;
   - um método abstrato `someMeCom`, que recebe um `NumeroAritmetico` e devolve um `NumeroAritmetico`.
   - um método abstrato `subtraiaDeMim`, que recebe um `NumeroAritmeticoe` devolve um `NumeroAritmetico`.
   - um método abstrato `multipliqueMePor`, que recebe um `NumeroAritmeticoe` devolve um `NumeroAritmetico`.
   - um método abstrato `dividaMePor`, que recebe um `NumeroAritmeticoe` devolve um `NumeroAritmetico`.

3. Crie uma classe concreta `InteiroAritmetico` tal que:

   - estende a classe `NumeroAritmetico`;
   - crie um construtor público que recebe um parâmetro `long`.
   - implementa as operações aritméticas de maneira usual, devolvendo a si mesmo em cada caso;
   - Sobrecarrega o método `equals()` da classe `Object`, que usa o método `mesmoValor()` da classe mãe.
   - Sobrecarrega o método `toString()` da classe `Object`.
   - Insira uma classe de testes que permita testar a funcionalidade dos métodos criados; se precisar, crie novos métodos públicos que permitam os testes.

4. Crie uma classe concreta `InteiroModular` tal que:

   - estende a classe `InteiroAritmetico`;
   - possui um segundo parâmetro `private int m`;
   - crie um construtor público que recebe dois parâmetros `long` e `int`, que chama o construtor `super`.
   - implementa as operações aritméticas de modo que o resultadoseja sempre módulo `m` (o resto da divisão por `m`), devolvendo a si mesmo em cada casoe que utiliza a operação correspondente da classe mãe;
   - Sobrecarrega o método `equals()` da classe mãe, verificando a igualdade de todos os membros.
   - Sobrecarrega o método `toString()` da classe mãe.
   - Insira uma classe de testes que permita testar a funcionalidade dos métodos criados; se precisar, crie novos métodos públicos que permitam os testes.

5. Crie uma classe concreta `InteiroModularComPotencia` tal que estende `InteiroModular`. Considere um objeto `var o = new InteiroMudularComPotencia`(n, m) tal que:

   - É possível realizar `o.elevadoA(`k`)` (com um inteiro k) e obter `o.valor` $= n^k \mod$ `o.m`. Utilize os métodos herdados para computar o resultado. O método devolve a si mesmo. Dica: $a^n \mod m = ((aa) \mod m)\ a^{n-2}) \mod m$
   - Encontre valores de n e de m tais que a sequência gerada por `o.elevadoA`(k) para k variando de 1 a $m-1$ (**a**) não tenha repetição, (**b**) tenha repetição. Escreva 2 testes, com o JUnit, para cada caso. Dica: teste com 11 módulo 13 e com 11 módulo 12.