Terceira Prova de MAB 240 — Computação II

Fabio Mascarenhas

6 de Julho de 2012

A prova é individual e sem consulta. Responda as questões na folha de respostas, a lápis ou a caneta. Se tiver qualquer dúvida consulte o professor.

Nome: _____

DRE: _____

Questão:	1	2	3	Total
Pontos:	4	2	4	10
Nota:				

1. Funções reais de uma variável podem ser modeladas em Java pela seguinte interface:

```
interface Funcao {
  double valor(double x);
}
```

   (a) (2 pontos) Defina a classe `Exp`, que implementa `Funcao` e representa a família de funções exponenciais $f(x) = n^x$, onde n é um número inteiro passado no construtor da classe `Exp`. Por exemplo, `(new Exp(9)).valor(0.5)` dá como resultado o número 3 ($9^{\frac{1}{2}}$ ou $\sqrt{9}$). A operação de exponenciação em Java é feita com a função `Math.pow`.

   (b) (2 pontos) O valor aproximado da derivada f' de uma função f qualquer em um ponto x pode ser obtido facilmente através da seguinte fórmula:

$$f'(x) = \frac{f(x+\epsilon) - f(x)}{\epsilon}$$

   onde ϵ é um número pequeno (quanto menor, melhor a aproximação). Defina a classe `Derivada`, que implementa `Funcao` e representa a derivada de uma função qualquer passada no construtor, dado um ϵ também passado no construtor. Por exemplo, `new Derivada(new Exp(2), 0.0001))` é a derivada da função $f(x) = 9^x$ com um ϵ de 0.0001.

2. (2 pontos) A classe a seguir representa uma modelagem bastante simples de uma conta corrente:

```
class ContaCorrente {
  private int numero;
  private double saldo;
```

```
  public ContaCorrente(int numero, double saldo) {
    this.numero = numero;
    this.saldo = saldo;
  }

  public void deposita(double valor) {
    this.saldo += valor;
  }

  public void retira(double valor) {
    this.saldo -= valor;
  }
}
```

Defina a classe `ContaCorrenteLanc` que extende `ContaCorrente` para incluir uma lista de lançamentos (uma instância de `ArrayList<String>`). Essa classe deve redefinir os métodos `deposita` e `retira` de `ContaCorrente` para adicionar as strings *"DEPÓSITO DE n"* ou *"RETIRADA DE n"* na lista de lançamentos a cada depósito ou retirada, respectivamente, onde n é o valor depositado ou retirado.

3. O padrão *enumerador* é uma maneira de se percorrer uma sequência de elementos. Um objeto enumerador possui dois métodos, um produz os elementos da sequência um por um, dando um erro se já produziu o último elemento, e o outro diz se já se chegou ao final da sequência ou não. Enumeradores para sequências de inteiros podem ser modelados pela seguinte interface:

```
interface Enumerador {
  int proximo();
  boolean final();
}
```

   (a) (2 pontos) Defina a classe `EnumVetor`, que implementa `Enumerador` e representa um enumerador para um vetor de inteiros passado no construtor. O trecho a seguir mostra um uso desse enumerador:

```
Enumerador e = new EnumVetor(new int[] { 1, 3, 5 });
System.out.println(e.proximo()); // imprime 1
System.out.println(e.final()); // imprime false
System.out.println(e.proximo()); // imprime 2
System.out.println(e.proximo()); // imprime 3
System.out.println(e.final()); // imprime true
```

   (b) (2 pontos) Escreva o corpo da função abaixo, que recebe um enumerador e retorna uma lista com todos os elementos que ele pode produzir:

```
static List<Integer> listaDeEnum(Enumerador e) {
  // corpo
}
```