Fundação CECIERJ - Vice Presidência de Educação Superior a Distância

Curso de Tecnologia em Sistemas de Computação
AD1 de Programação III
1° semestre de 2007

Nome:
Matrícula:
Pólo:

Exercício:
Em uma loja de produtos eletrônicos, alguns itens podem ser comprados avulsos (como placas, processadores e monitores de vídeo) enquanto outros (como computadores e gabinetes) são montados a partir de itens avulsos e/ou itens montados. Por exemplo, um computador pode ser montado com um monitor de vídeo, um teclado e um gabinete, sendo esse último montado com um processador e uma placa.

Todos os itens têm um código seqüencial que os identifica univocamente. Qualquer item, independente de ser um item avulso ou um item montado, deve ser capaz de informar qual o seu preço. O preço de um item de compra é calculado da seguinte forma:

- todos os itens avulsos de um mesmo tipo possuem o mesmo preço;
- um item montado tem o seu preço calculado pela soma dos preços de suas partes.

Projete classes para os seguintes itens de compra:

- processadores: são itens avulsos cujo preço é de US$100. Possui um tipo;
- monitores de Vídeo: são itens avulsos cujo preço é de US$500. Possui um tamanho (em polegadas);
- placas de Memória: são itens avulsos cujo preço é de US$30 por cada 64MB que possui;
- teclados: são itens avulsos cujo preço é de US$50;
- gabinetes: são itens que o usuário pode montar;
- computadores: são itens que o usuário pode montar.

Pense em uma solução genérica que possa servir para qualquer produto da loja que possa ser montado a partir de outros já existentes.

Você pode usar para o programa de teste a seguir para testar seu programa:

```
class Teste {

   public static void main (String[] args) {
      /* Marca os preços dos ítens avulsos */
      Placa.preco=30F;
      Processador.preco=100F;
      Monitor.preco=500F;
      Teclado.preco=50F;

      /* Cria diversos itens avulsos */
      Processador cpu1 = new Processador ("Pentium III");
      Processador cpu2 = new Processador ("AMD");
      Processador cpu3 = new Processador ("Pentium IV");
      Monitor mon1 = new Monitor (15);
      Monitor mon2 = new Monitor (19);
      Teclado tec1 = new Teclado ();
      Teclado tec2 = new Teclado ();
      Placa mem1 = new Placa (64);
      Placa mem2 = new Placa (64);
      Placa mem3 = new Placa (128);

      /* Cria os gabinetes */
      Gabinete gab1 = new Gabinete ();
      Gabinete gab2 = new Gabinete ();

      /* Cria os computadores */
      Computador com1 = new Computador();
      Computador com2 = new Computador();

      /* Monta os gabinetes */
      gab1.inclui (cpu1);
      gab1.inclui (mem1);
      gab1.inclui (mem2);
      gab2.inclui (cpu2);
      gab2.inclui (mem3);

      /* Monta os computadores */
      com1.inclui (gab1);
      com1.inclui (mon1);
      com1.inclui (tec1);
      com2.inclui (gab2);
      com2.inclui (mon2);
      com2.inclui (tec2);

      /* Mostra os computadores */
      System.out.println (com1);
      System.out.println (com2);
   }
}
```

```
public class Teste {
   public static void main (String[] args) {
      /* Marca os preços dos ítens avulsos */
      Placa.preco=30F;
      Processador.preco=100F;
      Monitor.preco=500F;
      Teclado.preco=50F;
      /* Cria diversos itens avulsos */
      Processador cpu1 = new Processador ("Pentium III");
      Processador cpu2 = new Processador ("AMD");
      Processador cpu3 = new Processador ("Pentium IV");
      Monitor mon1 = new Monitor (15);
      Monitor mon2 = new Monitor (19);
      Teclado tec1 = new Teclado ();
      Teclado tec2 = new Teclado ();
      Placa mem1 = new Placa (64);
      Placa mem2 = new Placa (64);
      Placa mem3 = new Placa (128);
      /* Cria os gabinetes */
      Gabinete gab1 = new Gabinete ();
      Gabinete gab2 = new Gabinete ();
      /* Cria os computadores */
      Computador com1 = new Computador();
      Computador com2 = new Computador();
      /* Monta os gabinetes */
      gab1.inclui (cpu1);
      gab1.inclui (mem1);
      gab1.inclui (mem2);
      gab2.inclui (cpu2);
      gab2.inclui (mem3);
      /* Monta os computadores */
      com1.inclui (gab1);
      com1.inclui (mon1);
      com1.inclui (tec1);
      com2.inclui (gab2);
      com2.inclui (mon2);
      com2.inclui (tec2);
      /* Mostra os computadores */
      System.out.println (com1);
      System.out.println (com2);
   }
}


abstract class Item {
   private int cod;
   private static int proxCod=0;
```

```
   Item () {
      this.cod=++proxCod;
   }

   abstract float obtemPreco();

   int obtemCodigo() { return this.cod; };

   public String toString() {
      return "(" + cod + ")";
   }
}


class Monitor extends Item {
   static float preco;
   private int tam;
   public Monitor (int tam) {
      this.tam=tam;
   }
   float obtemPreco () {
      return this.preco;
   }
   public String toString() {
      return "Monitor de tamanho " + tam + super.toString();
   }
}


class Placa extends Item {
   private int capacidade;
   static float preco;
   Placa (int capacidade) {
      this.capacidade=capacidade;
   }
   float obtemPreco () {
      return capacidade/64*preco;
   }
   public String toString() {
      return "Placa de " + capacidade + "de memória " +
super.toString();
   }
}


class Processador extends Item {
```

```
   static float preco;
   private String tipo;
   public Processador (String tipo) {
      this.tipo = tipo;
   }
   float obtemPreco () {
      return this.preco;
   }
   public String toString() {
      return "Processador " + tipo + super.toString();
   }
}


class Teclado extends Item {
   static float preco;
   float obtemPreco () {
      return this.preco;
   }
   public String toString() {
      return "Teclado " + super.toString();
   }
}


class ItemComposto extends Item {
   private final int TAM_MAX = 100;
   private Item[] partes;
   private int pos_livre;

   ItemComposto () {
        partes = new Item[TAM_MAX];
        pos_livre = 0;
   }

   void inclui (Item e) {
      if (pos_livre != TAM_MAX) partes[pos_livre++] = e;
   }

   float obtemPreco () {
      float total=0;
      for (int i = 0; i < pos_livre; i++)
         total += partes[i].obtemPreco();
      return total;
   }

   public String toString() {
```

```
      String texto="";
      for (int i = 0; i < pos_livre; i++)
         texto += partes[i];
      return texto + "\n";
   }
}


class Computador extends ItemComposto {
   public String toString() {
      return "Computador cujas partes são: " +
super.toString();
   }
}


class Gabinete extends ItemComposto {
   void inclui (Item e) {
      if ((e instanceof Processador) || (e instanceof Placa))
         super.inclui(e);
   }

   public String toString() {
      return "Gabinete cujas partes são: " +
super.toString();
   }
}
```