Mestrado Integrado em Engenharia de Comunicações
Métodos de Programação II
1º Ano
Ano lectivo 2008/2009
2º Teste Teórico

17/6/2009 Duração: 2h00

Leia atentamente as seguintes perguntas. Tente responder claramente a todas as perguntas desenvolvendo somente aquilo que é pedido.

1. Considere a seguinte definição de classe em Java1.5:

```
public abstract class Servico
{
    private int codigo;
    private double cc;       // volume em mts^3.
    private int unidades;    // número de unidades.
    private double preco;  // em euros.

    Servico(int k, double c, int u, double m)
    { this.codigo = k; this.cc = c; this.unidades = u; this.preco = m; }

    public int getCodigo()
    { return this.codigo;}
    public double getCC()
    { return this.cc;}
    public int getUnidr()
    { return this.unidades;}
    public double getPreco()
    { return this.preco;}

    public abstract double valor_servico();


    public Servico clone()
    { return new Servico(this.codigo, this.cc, this.unidades, this.preco);}

}
```

1.1 A classe descrita representa o serviço contratado entre um cliente e um Transitário (empresa de transportes de mercadorias). Dependendo do tipo de mercadoria, o valor do serviço (valor_servico) é calculado interpretando a variável preco como o preço por unidade (no caso do tipo Unidade) ou o preço por metro cúbico (no caso do tipo Espaço). Escrevas as subclasses Unidade e Espaco (na esqueça de apresentar os métodos equals()).

1.2 Escreva uma classe Armazem que regista todos os serviços prestados pela empresa. A classe representa para cada cliente (que tem associado um código de cliente) uma lista de serviços prestados. A classe deve conter as seguintes funcionalidades:

- Variáveis de instância necessárias para armazenar as correspondências entre clientes e serviços prestados,
- O construtor para criar um novo Armazem,
- Um método que dado um código de cliente e um serviço adiciona um novo serviço a um cliente,
- Um método que devolve uma colecção de códigos de clientes (sem repetições) que gastaram mais de 10000 euros em serviços nesta empresa,
- Um método para retornar o valor total de serviços efectuados pela empresa.
- Finalmente, um método que retorna uma estrutura de dados que relaciona códigos de serviços com número de unidades transportadas (assuma que dois serviços podem ter o mesmo código de operação mas com configurações diferentes i.e. diferentes valores de cubicagem e unidades!).

1.3 Apresente a árvore binária ordenada (mostrando para cada nó da árvore a célula representativa com as referências para os seus sucessores) para a sequência de valores [7,1,5,3,9,0,4,6]. Apresente a árvore que resulta da eliminação do nó que contém o valor 5.

O docente responsável

Paulo Azevedo

Correcção da Prova

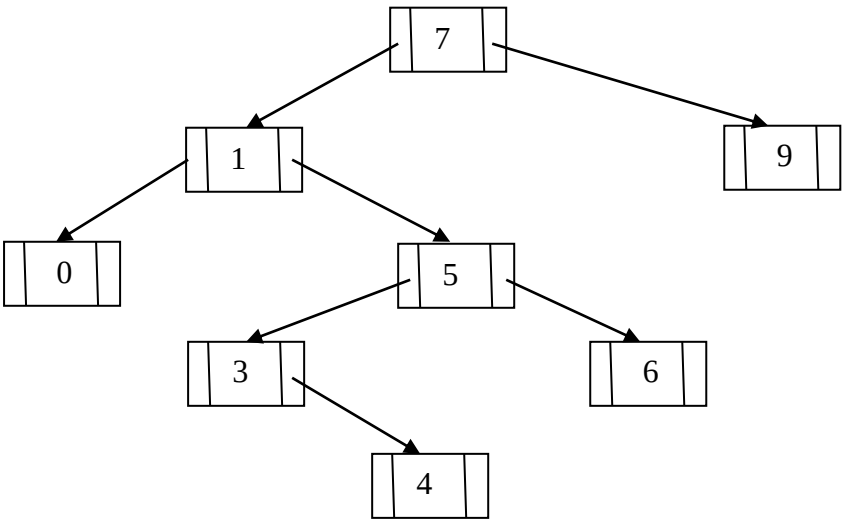
1.1

```
public class Unidade
{
        Unidade(int k, double c, int u, double m)
        { super(k,c,u,m);  }

        public double valor_servico();
        { return super.getPreco() * super.getUnidr();}

        public boolean equals( Object e)
        {       if(e == this) return(true);
                if(e == null) return(false);
                if(this.getClass() != e.getClass()) return(false);
                Unidade t = (Unidade)e;
                return( this.getCodigo() == e.getCodigo() && this.getCC() ==
e.getCC() && this.getUnidr() == e.getUnidr() && this.getPreco() == e.getPreco() );
        }
}

public class Espaco
{
        Espaco(int k, double c, int u, double m)
        { super(k,c,u,m);  }

        public double valor_servico();
        { return super.getPreco() * super.getCC();}

        public boolean equals( Object e)
        {       if(e == this) return(true);
                if(e == null) return(false);
                if(this.getClass() != e.getClass()) return(false);
                Espaco t = (Espaco)e;
                return( this.getCodigo() == e.getCodigo() && this.getCC() ==
e.getCC() && this.getUnidr() == e.getUnidr() && this.getPreco() == e.getPreco() );
        }
}
```

1.2
```
public class Armazem
    {
      private HashMap<Integer,ArrayList<Servico>> lista;

    Armazem()
    { this.lista = new HashMap<Integer,ArrayList<Servico>>();  }

    public void insere(int c, Servico novo)
    {
        if(this.lista.containsKey(c)
        {  ArrayList<Servico> temp = this.lista.get(c);
           temp.add(novo.clone());
        {
        else
           this.lista.put(c,new ArrayList<Servico>().add(novo.clone());
    }

public Set<Integer> maisdoque()
{
  HashSet<Integer> temp = new HashSet<Integer>();
  for(Integer i : this.lista.keySet())
  {  ArrayList<Servico> t = this.lista.get(i);
     double total = 0.0;
     for(Servico x: t)
     {  total += x.valor_servico();  }
     if(total > 10000) temp.add(i);
  }
  return(temp);
}


public double totalempresa()
{
   double total = 0.0;
   for(Integer i : this.lista.keySet())
   {
     ArrayList<Servico> t = this.lista.get(i);
     for(Servico x: t)
     {  total += x.valor_servico();  }
   }
  return(total);
}
```

```
public map<Integer,Integer> servicos_qty()
{
   HashMap<Integer,Integer> temp = new HashMap<Integer,Integer>();
   for(Integer k:this.lista.keySet())
      for(Servico x: this.lista.get(k))
         if(temp.containsKey(x.getCodigo())
         {
                  int t = temp.get(x.getCodigo()) + x.getUnidr();
                  temp.put(x.getCodigo(),t);
         }
         else
                  temp.put(x.getCodigo(),x.getUnidr()):
   return(temp);
}
```

1.3 Árvore após a inserção da sequência.



Árvore após a eliminação do elemento 5.