

# Exame Modelo de Paradigmas de Programação

1º ano da Licenciatura em Engenharia Informática do ISEP

Parte teórica (6 valores); todas as questões têm a mesma cotação

Prova sem consulta Duração: 20 minutos Responda na própria folha, marcando com um X ou ■ (pode usar lápis). Se pretender anular uma resposta escreva "anulada" à esquerda da mesma.	
Nome:	Número:
1. Considere a definição das seguintes classes:	
<pre>class A2 {     public static void print() {         System.out.print("A_"); }</pre>	
}	
<pre>class B2 extends A2 {     public static void print() {          System.out.print("B_");     } }</pre>	
<pre>class C2 extends B2 {     public void print2() {</pre>	
Qual é a saída produzida pelo seguinte código:	
A2. print(); B2. print(); C2. print(); C2 inst = <b>new</b> C2(); inst. print2();	
$\square$ ABCC	
■ ABBC	
$\square$ ABCB	
□ ABBB	
2. Selecione as afirmações corretas:	
O construtor por defeito é chamado automaticame sem o fornecimento de argumentos.	ente quando uma instância da classe é criada
$\hfill\square$ Não é possível invocar um construtor a partir de	um outro construtor da mesma classe.

- $\square$  Os métodos de uma classe não podem ser invocados a partir dos construtores porque eles ainda não estão inicializados.
- □ Quando uma classe contém a definição de vários construtores, o compilador escolhe o primeiro para construir os objetos.
- 3. Entre os exemplos de código que se seguem selecione os que compilam sem erros:

```
■ interface Ifc1 {
             void metodo();
   interface Ifc2 {
             void metodo();
   class Cls1 implements Ifc1 , Ifc2 {
     public void metodo() { System.out.println("Hello_world!"); }
   }
```



```
■ interface Ifc3 {
                   void metodo();
         interface Ifc4 extends Ifc3 {
                   \mathbf{void}\ \mathrm{metodo}\,(\,)\,;
         class Cls2 implements Ifc3, Ifc4 {
                   public void metodo() { System.out.println("Hello_world!"); }
         }
     ☐ interface I1 {
                   public int a = 10;
         interface I2 {
                   \hat{public} int a = 20;
                   public int b = 40;
         }
         class C1 implements I1, I2 {
                   int var1;
                   \mathbf{public} \ \mathbf{C1}() \ \{ \ \mathbf{var1} = \mathbf{a} + \mathbf{b}; \ \}
     ☐ interface I3 {
                   void metodo1();
         interface I4 {
                   void metodo1();
                   void metodo2();
         class C3 implements I3, I4 {
                   public void metodo1() { System.out.println("Hello1!"); }
public void metodo2() { System.out.println("Hello2!"); }
         class C4 {
                   public C4() {
                             I4 \ a = new \ C3();
                             I3 b = (I3)a;
                             a.metodo1();
                             b.metodo2();
                   }
        }
4. ...
5. ...
6. ...
```



## Exame Modelo de Paradigmas de Programação

#### Licenciatura em Engenharia Informática

#### Parte Prática - sem consulta

Duração: 2h00

Cotações: 1. – 5%; 2. – 15%; 3. – 40%; 4. – 20%; 5. – 20% (parte prática: 14 valores)

Considere uma aplicação Java para simular a eleição de um líder por parte dos membros do seu partido. Responda às alíneas seguintes tendo sempre em consideração os princípios de abstracção, encapsulamento, herança e polimorfismo.

- 1. Defina a classe Candidato de acordo com a seguinte descrição. A classe Candidato representa as pessoas que se candidatam ao lugar de líder do partido. Um candidato é caracterizado por:
- O seu nome atributo nome;
- O número de votos no candidato atributo numVotos.

## A classe Candidato deve cumprir os seguintes requisitos:

- O construtor da classe inicializa o nome usando um valor recebido por parâmetro e o número de eleitores do candidato a 0.
- · A classe inclui os seguintes métodos públicos:
  - um método incrementarVotos(), para incrementar o número de votantes no candidato;
  - o um método inicializarNumVotos() para inicializar o número de votos a zero.

### Considere ainda que os seguintes métodos de acesso estão já declarados:

- getNumVotos() que devolve o número de eleitores que votaram no candidato;
- getNome() que devolve o nome do candidato.
- 2. Defina a classe Escrutinio responsável por efectuar a eleição de um líder. A classe Escrutínio é caracterizada por:
- Conjunto de todos os candidatos a líder atributo candidatos;
- Número máximo de eleitores (todos os membros do partido) atributo numEleitores;
- Número de votantes atributo numVot;
- A data (dia) da eleição (um inteiro para simplificar) atributo data.

## A classe Escrutinio deve cumprir os seguintes requisitos:

- O construtor inicializa a zero o número de votantes; as restantes variáveis de instância devem ser inicializadas com valores recebidos por parâmetro;
- Os candidatos serão armazenados num ArrayList;
- A classe terá que fornecer os seguintes métodos públicos:
  - o um método calcularVotantes() que atribui ao atributo numVot, que representa o número efectivo de votantes, a soma do número de votantes em todos os candidatos;
  - um método inicializarVotosCandidatos() que inicialize a zero o número de votos de todos os candidatos;
  - o um método vencedor () que devolva o(s) candidato(s) mais votado(s).
- 3. Os membros do partido manifestam a sua preferência por um líder através do voto (classe Voto). Declare a classe Voto, tal como se descreve a seguir:
- um candidato (que é escolhido pelo boletim em questão) atributo candidato;
- a data do voto (um inteiro) atributo data;
- a data limite do voto a partir da qual o voto n\u00e3o poder\u00e1 ser considerado (um inteiro) atributo dataLim.

A classe Voto deve fornecer:



- um método para testar a validade do voto: boolean eValido();
- um construtor que inicializa o candidato, a data do voto e a data limite do voto, passados por parâmetro;
- a reescrita do método toString() para produzir uma representação do boletim de voto de acordo com o seguinte formato: <nome do candidato> -> inválido, se o boletim é inválido, e <nome do candidato> -> válido, se o voto é válido.

Considere ainda que os seguintes métodos de acesso estão já declarados:

- Candidato getCandidato();
- int getData();
- int getDataLimite().

Um voto pode ser feito através de um boletim em papel (classe VotoPapel), eletronicamente (classe VotoEletronico) ou pode ser submetido por correio (classe VotoCorreio).

Considere o seguinte no que respeita à validação dos votos:

- um voto eletrónico é inválido se a data for estritamente maior que o prazo limite subtraído de dois dias (devem ser enviados antes dos outros tipos de votos);
- um voto em papel é inválido se o boletim não está assinado (um atributo booleano indica se o voto está assinado). Este atributo será inicializado pelo construtor que receberá o parâmetro respectivo;
- um voto por correio é inválido se não estiver assinado ou se é estritamente maior do que a data limite.

As classes cuja validade do voto depende de uma data devem implementar a interface validaVotos que define o método boolean verificaData(). O método retornará true se o voto é válido.

As subclasses de Voto também oferecem redefinições do método toString() para produzir descrições respeitando os seguintes formatos:

- para votos em papel: voto em papel para <nome do candidato> -> inválido, se o voto é inválido e voto em papel para <nome do candidato> -> válido, se o voto é válido;
- para votos enviados por correio: envio de voto em papel para <nome do candidato> ->
  inválido, se o voto é inválido e envio de voto em papel para <nome do candidato> ->
  válido, se o voto é válido;
- Para votos eletrónicos: voto eletrónico para <nome do candidato> -> inválido, se o voto é
  inválido e voto eletrónico <nome do candidato> -> válido, se o voto é válido.
- 4. Defina a classe SimulaEleição contendo os seguintes membros:
- O atributo votos, representando um conjunto de votos de diferentes tipos (um ArrayList);
- O método contarVotos() capaz de atualizar o número de votos de cada candidato em função do conteúdo do ArrayList votos: para cada voto válido favorável ao candidato p, incrementar o número de votos do candidato p.
- 5. Pretende-se disponibilizar um formulário para introdução de votos eletrónicos. Defina o código necessário para criar um JDialog *modal* que contenha campos de texto, associados a componentes JLabel para descrição dos campos, que permitam introduzir o nome de um candidato e a data do voto (inteiro). Devem ser usados gestores de posicionamento que permitam manter a posição relativa entre componentes quando a caixa de diálogo é redimensionada. Deve ainda incluir um botão que quando accionado pelo utilizador imprima na consola uma descrição do voto, utilizando o método void actionPerformed(ActionEvent event) da interface ActionListener.