Fundação CECIERJ - Vice Presidência de Educação Superior a Distância

Curso de Tecnologia em Sistemas de Computação
Disciplina: Programação III
AP1 2° semestre de 2010.

Nome –

Assinatura –

Observações:

1. Prova sem consulta e sem uso de máquina de calcular.
2. Use caneta para preencher o seu nome e assinar nas folhas de questões e nas folhas de respostas.
3. Você pode usar lápis para responder as questões.
4. Ao final da prova devolva as folhas de questões e as de respostas.
5. Todas as respostas devem ser transcritas nas folhas de respostas. As respostas nas folhas de questões não serão corrigidas.

Questão 1) (4.0 pontos)

Projete e implemente uma classe empregado, gerente, secretaria e data. Todo empregado tem um nome, um salário e o ano de contratação. Para um dado empregado, deve ser possível: obter seu nome, obter seu salário, aumentar seu salário de acordo com um percentual, obter o ano de contratação e imprimir seus dados na tela. Um gerente (que também é um empregado) realiza todas as operações realizadas por um empregado, mas diferentemente de empregado, ele possui uma secretária (que também é um empregado) e um vetor de dois empregados. Nesta classe, deve ser possível aumentar o salário de todos os seus subordinados, inclusive de sua secretária, de um percentual passado como parâmetro de entrada. A secretária também possui todas as responsabilidades de empregado e deve imprimir os cinco últimos nomes de pessoas que entraram em contato com seu gerente. Use o programa de teste abaixo para analisar suas classes criadas.

```
public class Teste{

  public static void main(String[] args){

   Empregado[] e = new Empregado[6];

   e[0] = new Empregado("Jair Oliveira", 500.0, new Data(31, 7, 2010));

     e[1] = new Empregado("Jose da Silva", 700.0, new Data(20, 10,
1967));

   e[2] = new Secretaria("Maria Souza", 1000.0, new Data(2, 1, 1967));
```

```
   e[3] = new Secretaria("Nair Reis", 705.5, new Data(3, 6, 1987));

    e[4] = new Gerente("Vitor Massad", 7000.0, new Data(20, 10, 1998),
(Secretaria) e[2], e[0], e[1]);

   e[5] = new Gerente("Miguel Mello", 20000.0, new Data(17, 12, 2000),
(Secretaria) e[3], e[0], e[1]);



   int i;

   for(i = 0; i < e.length; i++) System.out.println(e[i]);

   if(e[5] instanceof Gerente)
     ((Gerente)e[5]).aumentaSalarioSubordinados(5);



   for(i = 0; i < e.length; i++) System.out.println(e[i]);



   ((Secretaria) e[3]).anotaRecado("Raul Cortez");

   ((Secretaria) e[3]).anotaRecado("Cazuza");

   ((Secretaria) e[3]).anotaRecado("Cassia Eller");

   ((Secretaria) e[3]).anotaRecado("Elis Regina");

   ((Secretaria) e[3]).anotaRecado("Paulo Autran");

   ((Secretaria) e[3]).imprimeContato();

   ((Secretaria) e[3]).anotaRecado("Luis Gonzaga");

   ((Secretaria) e[3]).anotaRecado("Gonzaguinha");

   ((Secretaria) e[3]).imprimeContato();

  }

}
```

RESPOSTA:

```
class Data{

  private int dia, mes, ano;

  Data(int dd, int mm, int aa){

    dia = dd;

    mes = mm;
```

```
    ano = aa;

  }

  public String toString(){
 return dia + "/" + mes + "/" + ano;
 }

}

class Empregado{

  protected String nome;

  protected double salario;

  protected Data data;


  Empregado(String n, double s, Data d){

   nome = n;

   salario = s;

   data = d;

  }

  String obterNome(){
 return nome;
 }

  double obterSalario(){
return salario;
 }

  Data obterAno(){
 return data;
 }

  void aumentaSal(double t){
 salario *= (1 + t);
 }

  public String toString(){

    return "Empregado: " + nome + " (" + data.toString() + ") " +
salario;

  }

}
```

```
class Gerente extends Empregado{

  protected Secretaria secret;

  protected Empregado[] subord;

   Gerente(String n, double s, Data d, Secretaria sec, Empregado e1,
Empregado e2){

   super(n, s, d);

   subord = new Empregado[2];

   secret = sec;

   subord[0] = e1;

   subord[1] = e2;

  }

  public String toString(){

    String s = "Gerente: " + nome + " (" + data.toString() + ") " +
salario + "\n";

   s += secret.toString() + "\n";

   for(int i = 0; i < 2; i++) s += subord[i].toString() + "\n";

   return s;

  }

  void aumentaSalarioSubordinados(double percentual){

   secret.aumentaSal(percentual/100.0);

   for(int i = 0; i < 2; i++) subord[i].aumentaSal(percentual/100.0);

 }

}

class Secretaria extends Empregado{

   protected String[] contatos;

   protected int pos_livre = 0;

   Secretaria(String n, double s, Data d){

     super(n, s, d);

     contatos = new String[5];
```

```
    }
    void anotaRecado(String n){
      contatos[pos_livre++] = n;
      if(pos_livre == 5)
        pos_livre = 0;
   }
    void imprimeContato(){
      int i = pos_livre, j = 0;
      while(j < 5){
        j++;
        if(contatos[i] != null) System.out.println(contatos[i++]);
        if(i == 5) i = 0;
      }
   }
   public String toString(){
       return "Secretaria: " + nome + " (" + data.toString() + ") " +
salario;
   }
}
```

Questão 2) (3.0 pontos)

Defina uma classe *IntervaloDeTempo,* cujos objetos representam um intervalo de tempo em número de horas, minutos e segundos, ou seja, existem campos na classe para cada um desses valores. O construtor de objetos dessa classe deve receber como argumento um número inteiro positivo, representando o número de segundos decorridos desde o instante inicial 00:00:00 horas, e retornar um objeto da classe *IntervaloDeTempo* correspondente. Por exemplo, a expressão *new IntervaloDeTempo(3500)* deve retornar um objeto que represente 0 horas, 58 minutos e 20 segundos. Além do construtor citado, defina para a mesma classe:

a) 2 métodos de soma, sendo que um recebe como parâmetro o tempo a ser somado em segundos, enquanto que o outro recebe um objeto IntervaloDeTempo;
b) 1 método chamado *toString()*, que à partir do conteúdo do objeto, retorne uma string com o formato dado acima: “x horas, y minutos e z segundos”.

Evite, sempre que possível, a replicação de código.

RESPOSTA:

```
class IntervaloDeTempo {
	// Não coloquei os modificadores de acesso
	//(private ou protected) apenas para tornar o
	//código menor. Mas estes são recomendáveis
	//na criação de aplicações reais.
	int hora, min, seg;

	// Construtor que converte segundos em h,m,s
	public IntervaloDeTempo(int tempo) {
		hora = tempo / 3600;
		min = (tempo - hora*3600) / 60;
		seg = tempo - hora*3600 - min*60;
	}

	// Método que realiza some de objetos
	// Chamada do método simplifica para o caso em que a
	//a soma de segundos ou minutos exceder 60 unidades
	public IntervaloDeTempo soma (IntervaloDeTempo t) {
		this.hora = this.hora + t.hora;
		this.min = this.min + t.min;
		this.seg = this.seg + t.seg;
		this.simplifica();
		return this;
	}

	// Método que adiciona unidades de segundo ao objeto
	// Novamente, chamada ao método simplifica
	public IntervaloDeTempo soma (int t) {
		IntervaloDeTempo aux = new IntervaloDeTempo(t);

		aux.hora = aux.hora + this.hora;
		aux.min = aux.min + this.min;
		aux.seg = aux.seg + this.seg;

		aux.simplifica();

		return aux;
	}

	// Método que retorna uma string do objeto
	public String toString () {
		String saida = hora + " horas, " + min + " minutos e " + seg + " segundos";
		return saida;
	}

	// Método que simplifica o objeto. Após uma soma, caso
	// o campo segundos seja maior que 60, cada 60 unidades destas
	// são transformadas em minutos.
	// Este método não era pedido na questão.
	public void simplifica () {
		if (this.seg >= 60) {
			this.seg = this.seg % 60;
			this.min = this.min + this.seg/60;
		}

		if (this.min >= 60) {
			this.min = this.min % 60;
			this.hora = this.hora + this.min/60;
		}
	}
}
```

```
// Classe de teste, apenas para verificar a corretude
//do código acima
public class AP1_2010_2_Q2 {
	public static void main(String[] args) {
		IntervaloDeTempo t = new IntervaloDeTempo(3500);
		System.out.println(t.toString());
		System.out.println(t.soma(100).toString());
		System.out.println(t.toString());
	}
}
```

Questão 3) (3.0 pontos)

Endereços WWW são cadeias de caractecres (strings) que usualmente se referem a algum domínio virtual e possuem o seguinte formato (exemplo): http://www.cederj.edu.br Eventualmente, estes endereços também podem apontar para recursos específicos dentro deste domínio: http://www.cederj.edu.br/vestibular/ , onde /vestibular é o recurso específico desejado. A comunicação entre servidores e navegadores web normalmente é feita através da porta padrão 80. Sem entrar em detalhes sobre o que isso representa, os endereços http://www.cederj.edu.br:80 e http://www.cederj.edu.br são equivalentes. Entretanto, podem existir domínios que respondam por uma porta não padrão, por exemplo: http://www.cederj.edu.br:8080

Crie uma classe que permite a manipulação de um endereço WWW, contendo:

a) Campos básicos de um endereço WWW: o domínio, a porta (80, caso não seja fornecida) e o recurso apontado
b) Construtor que receba as 3 informações em separado
c) Construtor que receba a string (endereço WWW) completa
d) Métodos para retornar cada campo separamente (domínio, porta e recurso)
e) Um método que verifique a consistência de um endereço. Por exemplo: os endereços tem que começar com http, ou www (quando o construtor que recebe as strings completas é acionado); a porta precisa ser um valor numérico; o domínio e o recurso não podem conter os caracteres '*', '@', '%' e '&'

Obs.: Para os que conhecem um pouco de programação para a web, no item e), assim como em toda esta questão, não está sendo considerada, para efeitos de simplificação, a passagem de parâmetros em urls.

RESPOSTA:

```
class EnderecoWWW {
	// Declarações do item a)
	private String dominio;
	private int porta;
	private String recurso;

	// Construtores pedidos no item b)
	public EnderecoWWW(String dom, int port, String rec) {
		dominio = dom;
		porta = port;
		recurso = rec;
	}

	public EnderecoWWW(String endereco) {
		// Para criação deste objetos foram implementados métodos estéticos
```

```
                //que recuperam cada campo de uma string www em separado

                dominio = "";
                porta = 0;
                recurso = "";

                if (this.verificaConsistencia(endereco)) { // Após item e)
                        dominio = EnderecoWWW.obtemDominio(endereco);
                        porta = EnderecoWWW.obtemPorta(endereco);
                        recurso = EnderecoWWW.obtemRecurso(endereco);
                }
        }

        // Métodos do item d)
        public static String obtemDominio(String endereco) {
                String inicioPadrao = "http://", dominio;
                if (endereco.startsWith(inicioPadrao)) // Remote inicio padrão
                        endereco = endereco.substring(inicioPadrao.length(),
endereco.length());

                int fimDominio = endereco.indexOf(':');
                if (fimDominio == -1) { // testa se existe porta não padrão
                        fimDominio = endereco.indexOf('/');
                        if (fimDominio == -1)
                                fimDominio = endereco.length();
                }
                dominio = endereco.substring(0, fimDominio);
                return dominio;
        }

        public static int obtemPorta(String endereco) {
                int fimDominio, fimPorta, porta;
                String inicioPadrao = "http://", sporta;
                if (endereco.startsWith(inicioPadrao)) // Remote inicio padrão
                        endereco = endereco.substring(inicioPadrao.length(),
endereco.length());

                fimDominio = endereco.indexOf(':');
                if (fimDominio == -1)
                        porta = 80;
                else {
                        fimPorta = endereco.indexOf('/', fimDominio);
                        if (fimPorta == -1)
                                fimPorta = endereco.length();
                        sporta = endereco.substring(fimDominio + 1, fimPorta);
                        porta = Integer.parseInt(sporta);
                }

                return porta;
        }

        public static String obtemRecurso(String endereco) {
                int inicioRecurso;
                String inicioPadrao = "http://", recurso;
                if (endereco.startsWith(inicioPadrao)) // Remote inicio padrão
                        endereco = endereco.substring(inicioPadrao.length(),
endereco.length());

                inicioRecurso = endereco.indexOf('/');
                if (inicioRecurso != -1)
                        recurso = endereco.substring(inicioRecurso+1, endereco.length());
                else
                        recurso = null;
                return recurso;
```

```
	}

	// Método exemplo de exibição (não solicitado na questão)
	public void exibe() {
		String saida = "URL criada: " + "http://" + dominio;
		if (porta != 80) {
			saida = saida + ":" + porta;
		}
		if (recurso != null) {
			saida = saida + "/" + recurso;
		}
		System.out.println(saida);
	}

	// Métodos de obtenção e alteração de campos solicitados no item c)
	public String getDominio() {
		return dominio;
	}

	public void setDominio(String dominio) {
		this.dominio = dominio;
	}

	public int getPorta() {
		return porta;
	}

	public void setPorta(int porta) {
		this.porta = porta;
	}

	public String getRecurso() {
		return recurso;
	}

	public void setRecurso(String recurso) {
		this.recurso = recurso;
	}

	// Item e)
	public boolean verificaConsistencia (String endereco) {
		if ((endereco.startsWith("http") || endereco.startsWith("www")) &&
				!(endereco.contains("*")) &&
				!(endereco.contains("@")) &&
				!(endereco.contains("%")) &&
				!(endereco.contains("&")))
			return true;
		return false;
	}
}

// Classe de teste para o código acima
public class AP1_2010_2_Q3 {
	public static void main(String[] args) {
		String www = "http://www.cederj.edu.br/vest";
		EnderecoWWW cederj = new EnderecoWWW(www);
		cederj.exibe();
	}
}
```