Fundação CECIERJ - Vice Presidência de Educação Superior a Distância

Curso de Tecnologia em Sistemas de Computação
Disciplina: Programação III
AP1 1° semestre de 2008.

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) (2.5 pontos)

Qual será o conteúdo das variáveis **`valores`**, **`produto e quociente`** ao término da execução do método **`main`**? Justifique sua resposta.

```
class ClasseMisteriosa{
  public static void main (String[] args){
    double[] valores = {1,2,3,4,5,6};
    double[] primeiraCópia = valores;
    double[] segundaCópia = valores;
    primeiraCópia [1] = 1;
    segundaCópia [2] = valores [0] + primeiraCópia [1];
    primeiraCópia[3] = valores[1]+ segundaCópia [2];
    valores[4] = primeiraCópia [2] + segundaCópia [3];
    valores[5] = segundaCópia [3] + primeiraCópia [4];
    int x, produto, quociente;
    x = produto = quociente = 5;
    produto *= x++;
    x = 5;
    quociente /= ++x;
  }
}
```

RESPOSTA:
valores = {1.0, 1.0, 2.0, 3.0, 5.0, 8.0}
produto = 25
quociente = 0

Questão 2) (2.5 pontos)
Implemente um programa que faça a inversão de um vetor de inteiros. Você deve passar, como parâmetro de entrada, o vetor a ser invertido. A saída deste programa será o vetor de entrada invertido (isto é, o elemento que estava inicialmente em primeiro passa a ser o último, o segundo elemento do vetor inicial será o penúltimo, e assim sucessivamente).

RESPOSTA:

```
class Inverte{

 public static void main (String[] args){
  int vet[] = {10, 9, 8, 7, 6, 5, 4, 3, 2, 1};

  inv_vet(vet);

  for (int i = 0; i < vet.length; i++) System.out.print(vet[i] + " ");
 }

 static void inv_vet(int vet[]){
  int n = vet.length;

  for (int i = 0; i < (n / 2); i++){
   int temp = vet[i];
   vet[i] = vet[n - i - 1];
   vet[n - i - 1] = temp;
  }
 }
}
```

Questão 3) (2.5 pontos)
Defina uma classe *IntervaloDeTempo,* cujos objetos representam um intervalo de tempo, em número de horas, minutos e segundos. 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: 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*; 1 método que retorne uma string, à partir do conteúdo do objeto, com o formato dado acima: “x horas, y minutos e z segundos”. Evite, sempre que possível, a replicação de código.

RESPOSTA:

```
class IntervaloDeTempo {
	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
      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;
            }
      }
}
```

Questão 4) (2.5 pontos)
Considere o código abaixo que modela objetos de um jogo de guerra:

```
            class ObjetoVisual {
                  float vida; // Varia entre 0, .25, .5, 1
                  int pontuacao; // + 1 para cada acerto
```

```
        int pos_x; // Posicao no eixo x
        int pos_y; // Posicao no eixo y
        float largura; // Largura do objeto
        float altura; // Altura do objeto

        void desenha();
    }

    class Aviao extends ObjetoVisual {
        void movimenta (int x, int y) {
            pos_x = x;
            pos_y = y;
        }
    }
```

a) (0.5 pontos) Que mudança(s) precisamos fazer no código para que a classe *Avião* seja obrigada a redefinir o método *desenha()* da classe *ObjetoVisual*?

b) (0.5 pontos) Para as 2 classes listadas, defina construtores que recebam parâmetros que inicializem os objetos das respectivas classes.

c) (0.75 ponto) Defina uma classe *Canhão* que, além das características de um objeto visual, esta também possua um atributo decimal chamado *angulo*, o qual armazenará o ângulo de inclinação do canhão. Note que um canhão também precisará ser desenhado.

d) (0.75 ponto) Crie uma classe chamada *Bomba*, cujos objetos poderão ser criados à partir de um avião. Cada bomba terá uma posição (x, y) e suas altura e largura são desprezíveis (valem 0). Além disso, possui um atributo chamado alcance, que fornece o raio de destruição da bomba. Uma bomba não possui dados como vida ou pontuação.

RESPOSTA:

```
abstract class ObjetoVisual {
      float vida; // Varia entre 0, .25, .5, 1
      int pontuacao; // + 1 para cada acerto
      int pos_x; // Posicao no eixo x
      int pos_y; // Posicao no eixo y
      float largura; // Largura do objeto
      float altura; // Altura do objeto

      ObjetoVisual(float v, int p, int x, int y, int l, int a) {
            vida = v;
            pontuacao = p;
            pos_x = x;
            pos_y = y;
            largura = l;
            altura = a;
      }

      abstract void desenha();
}

class Aviao extends ObjetoVisual {
      public Aviao(float v, int p, int x, int y, int l, int a) {
            super(v, p, x, y, l, a);
```

```
	}

	void movimenta (int x, int y) {
		pos_x = x;
		pos_y = y;
	}

	void desenha() {
		// Comandos para o desenho de um aviao
	}
}

class Canhao extends ObjetoVisual {
	float angulo; // 0 a 180

	public Canhao(float v, int p, int x, int y, int l, int a, int al)
{
		super(v, p, x, y, l, a);
		angulo = al;
	}

	void desenha() {
		// Comandos para o desenho de um canhao
	}
}

class Bomba {
	int pos_x; // Posicao no eixo x
	int pos_y; // Posicao no eixo y
	float alcance;

	public Bomba(int x, int y, float a) {
		pos_x = x;
		pos_y = y;
		alcance = a;
	}
}
```