Fundação CECIERJ - Vice Presidência de Educação Superior a Distância

Curso de Tecnologia em Sistemas de Computação
Disciplina: Programação III
AP1 2° 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)
Implemente um programa que teste se um número é um palíndromo (**Dica:** se um número pode ser lido, indiferentemente, da esquerda para a direita e vice-versa, ele é considerado um palíndromo). Você deve passar, como parâmetro de entrada para o método de verificação, o número a ser testado. A saída deste método será um valor booleano que indicará se o vetor é palíndromo (**`true`**) ou não (**`false`**). O seu programa deverá tratar a saída deste método, imprimindo as seguintes mensagens "É UM PALÍNDROMO" ou "NÃO É UM PALÍNDROMO" na console.

RESPOSTA:

```
class Palindromo{
  public static void main (String[] args){
    int n = args[0].length();
    char vet[] = new char[n];
    int i;

    for (i = 0; i < n; i++)
      vet[i] = args[0].charAt(i);

    if(testa(vet))
      System.out.println("E UM PALINDROMO");
    else
      System.out.println("NAO E UM PALINDROMO");
  }
```

```
  public static boolean testa (char[] vet){
    int i, n = vet.length;

    for(i = 0; i < n / 2; i++)
      if (vet[i] != vet[n - 1 - i])
        return false;
    return true;
  }
}
```

Questão 2) (2.5 pontos)
Implemente um método que verifica se uma matriz quadrada de inteiros é uma matriz identidade. Você deve passar, como parâmetro de entrada, a matriz a ser testada e a dimensão. A saída deste método será um valor booleano que indicará se a matriz é identidade (**`true`**) ou não (**`false`**).

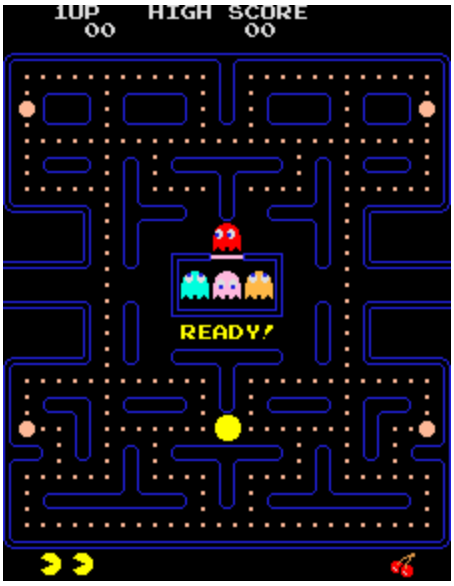
RESPOSTA:

```
public static boolean identidade (int[][] mat, int n){
    int i,j;

    for(i = 0; i < n; i++)
      for(j = 0; j < n; j++){
        if (i == j){
          if (mat[i][j] != 1)
          return false;
        }
        else if (mat[i][j] != 0)
          return false;
      }

    return true;
}
```

Questão 3) (2.5 pontos)
O jogo Pac-Man (Come-come) é um jogo clássico e foi lançado no ano de 1980 (ver figura a seguir). Neste jogo, o objetivo principal é obter a maior pontuação possível (high score). Para tal, o jogador (bolinha amarela) obtém pontos "comendo" os pontinhos espalhados pelo mapa. Os inimigos do jogador são os fantasmas (4 bonecos coloridos no centro da figura). Entretanto, o jogador consegue comê-los, por um curto intervalo de tempo (período de imortalidade), após comer algum ponto especial (pontos maiores nos extremos da figura). Após este período de tempo, o jogador pode ser morto por algum fantasma. Tanto jogador quanto fantasmas podem se movimentar livremente pelo mapa, respeitando os obstáculos que nele existir (formas geométricas azuis na figura).



Imagine que vamos implementar este jogo. Crie classes para cada um dos seguintes elementos e as relacione, se for o caso, de forma a se aproveitar do reuso de código: Jogador, Fantasma, Ponto, PontoEspecial e Objeto. A classe Objeto possui um campo chamado posição, a qual conterá a posição (x,y) de um objeto (imagine que o mapa é uma matriz onde cada objeto ocupa uma posição).

RESPOSTA:

```
// Classe que será usada por todos os objetos que forem desenhados
class Posicao {
	int x, y;

	public Posicao(int x, int y) {
		this.x = x;
		this.y = y;
	}

	public int getX() {
		return x;
	}

	public void setX(int x) {
		this.x = x;
	}

	public int getY() {
		return y;
	}

	public void setY(int y) {
		this.y = y;
	}
}
```

```
// Classe abstrata que contém o atributo Posicao, um construtor
//padrão e métodos de acesso ao atributo, os quais serão herdados
//pelas classes especializadas de Objeto
abstract class Objeto {
	Posicao pos;

	public Objeto(Posicao pos) {
		this.setPos(pos);
	}

	public Posicao getPos() {
		return pos;
	}

	public void setPos(Posicao pos) {
		this.pos = pos;
	}
}

// Classe que modela um jogador do jogo
class Jogador extends Objeto {
	int pontuacao = 0;

	public Jogador(Posicao p) {
		super(p);
	}

	public int getPontuacao() {
		return pontuacao;
	}

	public void setPontuacao(int pontuacao) {
		this.pontuacao = pontuacao;
	}
}

// Classe que modela um fantasma no jogo
class Fantasma extends Objeto {
	public Fantasma(Posicao p) {
		super(p);
	}
}

// Classe que modela um ponto no jogo
class Ponto extends Objeto {
	int valor = 1;

	public Ponto(Posicao p) {
		super(p);
	}

	public int getValor() {
		return valor;
	}

	public void setValor(int valor) {
		this.valor = valor;
```

```
	}
}

//Classe que modela um ponto especial no jogo
class PontoEspecial extends Ponto {
	public PontoEspecial(Posicao p) {
		super(p);
		this.setValor(5);
	}
}

class Obstaculo extends Objeto {
	public Obstaculo(Posicao pos) {
		super(pos);
	}

}

// Exemplo de uso das classes acima, mas que não era pedido na questão
// Suponha uma matriz de tamanho 40x40 que modele o jogo
// Suponha também que a posição 0x0 é o canto esquerdo superior
class Pacman {
	Jogador j;
	Fantasma f1, f2, f3, f4;
	PontoEspecial p1, p2, p3, p4;
	int imortalidade = 10; // 10 segundos
	int estado = 0; // 0: jogador mortal, 1: jogador imortal

	public Pacman() {
		f1 = new Fantasma(new Posicao(20,20));
		f2 = new Fantasma(new Posicao(20,20));
		f3 = new Fantasma(new Posicao(20,20));
		f4 = new Fantasma(new Posicao(20,20));
		j = new Jogador(new Posicao(20,30));

		// Criação dos pontos
		p1 = new PontoEspecial(new Posicao(0,0));
		p2 = new PontoEspecial(new Posicao(0,40));
		p3 = new PontoEspecial(new Posicao(40,0));
		p4 = new PontoEspecial(new Posicao(40,40));

		// Criação dos obstáculos
		// Criação de instâncias da classe Obstáculo
	}
}
```

Questão 4) (2.5 pontos)

No código abaixo temos a declaração de 3 classes: uma classe para inteiros (Inteiro), outra para racionais (Racional), que estende a de inteiros, e uma classe para teste (Teste). Qual o valor impresso pelos 2 métodos *print* chamados pelo método *main* da classe Teste?

```
interface Valor {
	void inc ();
	void print ();
```

```
}

class Inteiro implements Valor {
	int a;
	Inteiro (int x) {
		a = x;
	}
	public void add (Inteiro x) {
		a = a + x.a;
	}
	public void inc () {
		a = a + 1;
	}
	public void print () {
		System.out.println(a);
	}
}

class Racional extends Inteiro {
	int b;
	Racional (int x, int y) {
		super(x); b = y;
	}
	public void add (Racional x) {
		a = a * x.b + b * x.a;
		b = b * x.b;
	}
	public void inc () {
		a = a + b;
	}
	public void print () {
		System.out.print(a);
		System.out.print("/");
		System.out.println(b);
	}
}

public class Teste
{
	public static void main(String[] a) {
		Inteiro i = new Inteiro(1);
		Racional r = new Racional(2,3);
		i = r;
		i.inc();
		i.print();
		i.add(new Inteiro(1));
		i.print();
	}
}
```

RESPOSTA:
5/3
6/3

Nesta questão, o objetivo era avaliar o aluno com respeito à atribuição de objetos e suas conseqüências em Java. Neste caso, a variável *i*, que inicialmente tinha valor *1*,

passa a apontar para o conteúdo do racional após a atribuição *i = r;*. Após a atribuição, todos os métodos executados são da classe Racional, exceto a chamada *i.add(new Inteiro(1));*, a qual fornece um argumento do tipo Inteiro, somente disponível na classe Inteiro.