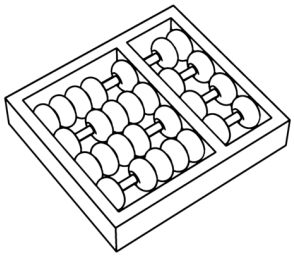


**MC302 - Programação Orientada a Objetos**
**Instituto de Computação - Unicamp**
Primeiro Semestre de 2017
Profa. Esther Colombini `esther@ic.unicamp.br`
PEDs: Elisangela Santos (ra149781@students.ic.unicamp.br)
Lucas Faloni (lucasfaloni@gmail.com)
Lucas David (lucasolivdavid@gmail.com)
Wellington Moura (wellington.tylon@hotmail.com)
PAD: Igor Torrente (igortorrente@hotmail.com)
`http://www.ic.unicamp.br/~esther/teaching/2017s1/mc302`

## Lista de Exercícios 2

---

### Tópicos:

- Arrays e Classe Random
- Polimorfismo
- Herança Múltipla e Interfaces
- Tratamento de Exceções
- Enumerações
- Arquivos

---

**1** Quais dos seguintes processos são permitidos com classe abstratas?

- A. declarar objetos
- B. retornar um objeto de uma função
- C. enviar um objeto como argumento para uma função
- D. declarar ponteiros

**2** Crie uma estrutura hierárquica que contenha as seguintes classes: Veiculo (classe abstrata), Bicicleta e Automóvel. Os métodos da classe Veiculo são todos abstratos e possuem a seguinte assinatura:

- public float acelerar(float velocidade);
- public void parar();

Estes métodos são implementados nas subclasses Automóvel e Bicicleta. Acrescentar na classe Automóvel o método public void trocarOleo(float litros).

**3** [4] Implemente uma classe abstrata de nome Forma onde são declarados dois métodos abstratos: float calcularArea(); float cacularPerimetro(); Crie, como subclasse de Forma, uma classe de nome Retangulo cujas instâncias são caracterizadas pelos atributos lado e altura ambos do tipo float. Implemente na classe Retangulo os métodos herdados de Forma e outros que ache necessários. Crie, como subclasse de Forma, uma classe de nome Circulo cujas instâncias são caracterizadas pelo atributo raio do tipo float. Implemente na classe Circulo os métodos herdados de Forma e outros que ache necessários. Crie, como subclasse de Retangulo, uma classe de nome Quadrado cujas instâncias são caracterizadas por terem os atributos lado e altura com o mesmo valor. Elabore um programa de teste onde é declarado um array, de dimensão 5, do tipo estático Forma. Nesse array devem ser guardadas instâncias de Retangulo, Circulo e Quadrado seguindo uma ordem aleatória. Depois implemente um ciclo que percorra o array evocando, relativamente a cada um dos objetos guardados, os métodos calcularArea e calcularPerimetro.

**4** Qual a diferença entre exceção verificada (checked) e não verificada (unchecked)? Dê exemplos de ambas.

**5** Como é a estrutura de controle para se usar/capturar exceções em Java? Explique como funciona o bloco try-catch-finally.

**6** Existe alguma hierarquia nas exceções? Como esta suposta hierarquia influencia na sequência de escrita de múltiplo blocos catch para um único try?

**7** O que acontece se múltiplos blocos catch correspondem ao tipo da exceção lançada?

**8** Explique o que são enumerações. O que representa cada constante de uma enumeração? O que faz o método values() de uma enumeração? Por que atributos estáticos não podem ser referenciados em um construtor de uma enumeração?

**9** Escreva instruções que atribuem inteiros aleatórios à variável n nos seguintes intervalos utilizando Math.random():

```
a) 1 <= n <= 2
b) 1 <= n <= 100
c) 0 <= n <= 9
d) 1000 <= n <= 1112
e) -1 <= n <= 1
f) -3 <= n <= 11
```

**10** Considere uma classe Pilha. Implemente a classe e suas duas principais operações, push e pop. Push é um método que deve lançar uma exceção do tipo PilhaCheia sempre que não couber mais elementos na pilha. Pop deve lançar uma exceção do tipo PilhaVazia sempre caso a pilha não tenha mais elementos a serem retirados. Ambas exceções devem herdar da classe PilhaExcecao. Implemente as três classes de exceção e os métodos que as lançam (push e po). Implemente também uma classe Teste que mostra como criar uma pilha e invocar os métodos push e pop com os respectivos tratamentos de exceções.

**11** [5] Sobre enumerações em Java, não é correto afirmar que...

- Uma constante de enumeração pode ser considerada como uma variável estática e final que referencia um objeto do tipo enumerado
- A declaração de uma constante de enumeração corresponde à instanciação de um objeto, logo argumentos podem ser passados para o construtor
- Um tipo enumerado não contém outras instâncias fora aquelas definidas por constantes
- Os construtores de um tipo enumerado devem ser declarados públicos

**12** [5] Qual o objetivo de declarar um atributo como final?

A. O atributo não pode ser ocultado em uma sub-classe

B. O valor do atributo não pode ser alterado

C. Todos os objetos enxergam o mesmo valor do atributo

D. O atributo é uma constante de classe

**13** [5] Quais das seguintes declarações não causam erro de compilação?

```
1. float[] f0 = new float(3);
2. float f1[] = new float[ ];
3. float[] f2 = new float[3];
4. float f3[] = new float[3];
5. float f4[] = {1.0f, 2.0f, 2.0f};
```

**14** [5] Qual das instruções abaixo é uma declaração válida para vetor?

```
A. int *x;
B. int x[5];
C. int[3] x = 1,2,3;
D. int[][] x[];
```

**15** [5] A respeito de atributos de instância finais, é incorreto afirmar que...

A. devem ser inicializadas exclusivamente durante a declaração ou no método construtor

B. seus valores, uma vez inicializados, não podem mais ser modificados

C. podem ser variáveis de tipo primitivo ou tipo referenciado

D. também são denominadas constantes da classe

**16** [5] Assinale a alternativa que mostra o intervalo de todos os valores possíveis para a variável randomValue.

```
int randomValue = 3 + randomNumbers.nextInt(5)
```

A. [3, 7]

B. [3,8]

C. [4,8]

D. [4, 7]

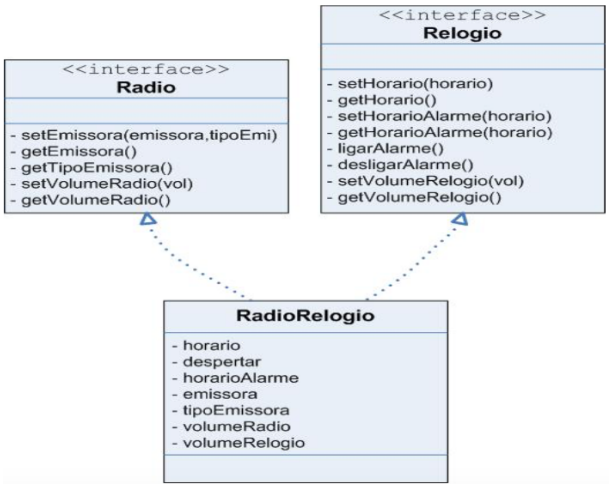
**17** Quais palavras-chave devem ser utilizadas na declaração de uma constante de classe em Java?

**18** Crie a seguinte hierarquia de classes:

- Uma interface para representar qualquer forma geométrica, definindo métodos para cálculo do perímetro e cálculo da área da forma;
- Uma classe abstrata para representar quadriláteros. Seu construtor deve receber os tamanhos dos 4 lados e o método de cálculo do perímetro já pode ser implementado;
- Classes para representar retângulos e quadrados. A primeira deve receber o tamanho da base e da altura no construtor, enquanto a segunda deve receber apenas o tamanho do lado;
- Uma classe para representar um círculo. Seu construtor deve receber o tamanho do raio.

No programa principal, pergunte ao usuário quantas formas ele deseja criar. Em seguida, para cada forma, pergunte se deseja criar um quadrado, um retângulo ou um círculo, solicitando os dados necessários para criar a forma. Todas as formas criadas devem ser armazenadas em um vetor. Finalmente, imprima: (a) os dados (lados ou raio); (b) os perímetros; e (c) as áreas de todas as formas. Para (b) e (c), tire vantagem do polimorfismo, enquanto que para (a) utilize instanceof e downcast.

**19** Implementar um RadioRelogio, na linguagem Java, segundo o diagrama simples apresentado abaixo. A classe RadioRelogio deve ter o seguinte construtor: public RadioRelogio(Date horario).



**20** [5] Na linguagem Java, em contraste com C (por exemplo) vetores são objetos. Liste 3 implicações desse fato e descreva como esses pontos seriam diferentes se os vetores não fossem objetos.

**21** [5] Faça um programa com uma classe `Cidadao`, com nome, sexo e data de nascimento. Declare uma enumeração parar representar o sexo e use ela na classe `Cidadao`. Crie um método estático para converter um valor do tipo da enumeração em uma descrição textual.

**22** [1] Considere um vetor bidimensional de inteiros $t_{2\times3}$.

a) Escreva um comando para instanciá-lo.

b) Quantas linhas há em `t`?

c) Quantas colunas há em `t`?

d) Quantos elementos há em `t`?

e) Escreva expressões para acessar todos os elementos na linha 1 de `t`.

f) Escreva expressões para acessar todos os elementos na coluna 2 de `t`.

g) Escreva um único comando para guardar zero no elemento da linha 0 e coluna 1 de `t`.

h) Escreva comandos individuais para inicializar cada elemento de `t` em zero.

i) Escreva um for aninhado para inicializar cada elemento de `t` em zero.

j) Escreva um for aninhado para pedir cada elemento de `t` do usuário.

k) Escreva uma série de comandos para encontrar o menor elemento de `t`.

l) Escreva um único printf que imprima todos os elementos da primeira linha de `t`.

m) Escreva um comando que some todos os elementos da terceira coluna de `t`. Não use repetições.

n) Escreva uma série de comandos para imprimir `t` no formato de tabela. Liste os índices das colunas no topo e os índices das linhas à esquerda.

**23** [5] Usando a classe `Random` para gerar valores aleatórios e vetores multidimensionais, implemente o jogo **2048** (`http://gabrielecirulli.github.io/2048/`). Use a saída padrão para imprimir a matriz e peça as direções (cima, baixo, esquerda e direita) a partir de comandos texto do usuário. Para aprender mais sobre o jogo, (jogue! e) acesse `http://en.wikipedia.org/wiki/2048_(video_game)`.

Algumas dicas:

1. Use uma matriz (vetor de duas dimensões) $4x4$ com valores inteiros para representar cada bloco;
2. Use números aletórios para gerar a posição do novo bloco a aparecer na tela;
3. Crie funções para imprimir o jogo na tela;
4. Crie funções para fazer o movimento na matriz, cuidando para não avançar seus limites;
5. Use uma classe somente para representar o jogo e outra para fazer a iteração com o usuário. Na primeira não use nenhuma função de entrada ou saída;

6. Para simplificar, o bloco novo pode ser sempre o de valor 2.

**24** [1] De que modo o polimorfismo permite programar em um nível "geral" ao invés de programar em um nível "específico"? Discutir as vantagens da programação no nível "geral".

**25** [1] O que são métodos abstratos? Discuta em quais circunstâncias é apropriado utilizar um método abstrato.

**26** Compare classes abstratas e interfaces. Em quais circunstância é mais apropriado utilizar classes abstratas? E interfaces?

**27** [1] Crie uma interface `Animal` contendo um único método `emitirSom`. Em seguida, crie as classes `Cachorro`, `Gato` e `Vaca` que implementam `Animal`. Em um método `main`, instancie um vetor de animais contendo objetos dos tipos `Cachorro`, `Gato` e `Vaca`. Em seguida, através de um loop, percorra o vetor chamando o método `emitirSom`. Discuta os resultados.

**28** [6] Quais afirmações são verdadeiras?

a) Uma classe abstrata não pode ser instanciada.

b) Uma interface pode extender multiplas interfaces

c) Todos os métodos de uma classe abstrata devem ser abstratos

d) Se a classe abstrata B estende a classe abstrata A, então a classe B deve implementar todas as classes abstratas de A

e) Se a classe concreta C estende a classe concreta B, e B implementa a interface A, então métodos da interface A podem ser chamadas em uma instância de C.

**29** [6] Dado:

```
1. abstract class A {}
2. class B {}
3. interface C {}
4. interface D {}
5. //insira código aqui
```

Qual das opções abaixo, ao ser inserida na linha 5, resulta num erro de compilação?

a) `class E extends A {}`

b) `class E extends A, B {}`

c) `class E implements C {}`

d) `class E implements C, D {}`

e) `interface E extends C, D {}`

f) `class E extends B implements D {}`

**30** [1] Explique porque o tratamento de exceções é um meio eficaz para lidar com falhas.

**31** Explique o comportamento das classes (e subclasses) a seguir e suas diferenças, em termos de tratamento de exceções:

a) Throwable

b) Exception

c) RuntimeException

d) Error

**32** [1] Escreva um programa que mostra que a ordem dos blocos catch é importante, ou seja, se for capturado um tipo de exceção antes de um outro tipo de exceção que é subclasse da primeira, é possível que o compilador gere novos erros.

**33** [7] Utilize herança para criar uma superclasse de exceção (chamado ExceptionA) e subclasses de exceção ExceptionB e ExceptionC, onde ExceptionB herda de ExceptionA e ExceptionC herda de ExceptionB. Escreva um programa para demonstrar que o bloco catch para o tipo ExceptionA captura exceções dos tipos ExceptionB e ExceptionC.

**34** [1] Encontre o erro do código abaixo.

```
class ExceptionA extends Exception {}

class ExceptionB extends ExceptionA {}

public class Test{
  void thrower() throws ExceptionA{
    throw new ExceptionA();
  }

  public static void main(String[] args){
    Test t = new Test();
    try{t.thrower();}
    catch(ExceptionB e) {}
  }
}
```

**35** [1] Escreve um programa definindo as classes ExceptionA (que herda da classe Exception) e ExceptionB (que herda de classe ExceptionA). Em seu programa, criar blocos try que disparam exceções de tipos ExceptionA, ExceptionB, NullPointerException e IOException. Todas as exceções devem ser capturadas com blocos catch especificando o tipo de exceção.

**36** [1] Escreva um programa que define uma classe SomeClass, que dispara uma Exception no construtor quando algum argumento inválido é passado. Seu programa deve tentar criar um objeto do tipo SomeClass e capturar a exceção que é disparada do construtor.

**37** [1] Determine se as afirmações abaixo são verdadeiras ou falsas. Se falsa, explique.

a) Você deve explicitamente criar os objetos stream System.in, System.out e System.err.

b) Ao ler dados de um arquivo usando a classe Scanner, se você quiser ler os dados do arquivo várias vezes, o arquivo deve ser fechado e reaberto para ler a partir do início do arquivo.

c) O método exists da classe File retorna verdadeiro se o nome especificado como o argumento para o construtor do arquivo é um arquivo ou diretório.

d) Arquivos binários são legíveis em um editor de texto.

e) Um caminho absoluto contém todos os diretórios, começando com o diretório raiz que levam para um arquivo ou diretório específico.

f) A classe Formatter contém o método printf, que permite que dados formatados sejam escritos na tela ou em um arquivo.

**38** [1] Encontre o erro de cada trecho de código e corrija.

a) Assuma que `conta`, `companhia` e `quantia` estejam declarados.

```
ObjectOutputStream outputStream;
outputStream.writeInt(conta);
outputStream.writeChars(companhia);
outputStream.writeDouble(quantia);
```

b) O seguinte trecho de código deve ler um registro do arquivo "conta.txt". A variável `inConta` deve ser usada para se referir a esse arquivo.

```
Scanner inConta = new Scanner( new File( "conta.txt" ) );
RegistroContas contas = ( RegistroContas) inConta.readObject();
```

**39** [5] Considere o arquivo vazio `C:/empty.txt`. O que acontece quando estes código são compilados e executados?

a)

```
void test() throws IOException {
BufferedReader reader = new BufferedReader(
          new FileReader("C:/empty.txt"));
System.out.println(reader.getLine());
}
```

b)
```
void test() {
FileWriter writer = new FileWriter("/fun.log");
writer.write("Hello!");
writer.close();
}
```

c)
```
void test() throws IOException {
for (int index = 1; index <= 2; index++) {
PrintWriter writer = new PrintWriter("/apa");
writer.print("apa");
writer.close();
}
}
```

**40** Crie um programa que lê todos os arquivos de texto de um diretório e escreve o conteúdo em apenas um único arquivo.

**41** O que é e para que serve a serialização de um objeto? Como podemos serializar e desserializar objetos? Exemplifique.

# Referências


[1] Paul Deitel, Harvey Deitel, *Java: How to Program*. Deitel, 9a Edição.

[2] *Java: The java tutorials.* http://docs.oracle.com/javase/tutorial/java/javaOO/QandE/nested-questions.html

[3] *Blog: Questões Comentadas para Certificação Java SCJP.* http://scjpquestoes.blogspot.com.br/

[4] *Primeira Lista de Exercícios* http://www.lac.inpe.br/~rafael.santos/Docs/IntroPOOJava/

[5] *Lista de Exercícios de POO* http://www.ic.unicamp.br/~fusberti/

[6] *Questões de Certificação SCJP.* http://www.slideshare.net/silveiraneto/questes-de-certificao-scjp

[7] *SCJP Flow Control and exception handling* www.javaprogrammingworld.com/scjp-flow-control-and-exception-handling/