



LABORATÓRIO DE PROGRAMAÇÃO II - ATIVIDADE DIRIGIDA 3

Questões:

1. Observe o seguinte programa Java:

```
package p;
public class exemplo {

    public exemplo() { }
    public static void
    main(String[] args) {
        try {
            System.out.println(1/0);
            System.out.println("M");
        }
        catch (ArithmeticException
ex2) {
            System.out.print("X");
        }
        catch (Exception ex3) {
            System.out.print("Y");
        }
        finally {
            System.out.print("Z");
        }
        System.out.print("F");
    } //Fim do método main

} //Fim da classe exemplo
```

A saída desse programa é

- (A) MXYZF
- (B) XYZ
- (C) XF
- (D) XZF
- (E) XYZF

2. Qual das afirmações a seguir faz uma apreciação correta a respeito da linguagem de programação Java?

(A) O conceito de herança múltipla é implementado nativamente.

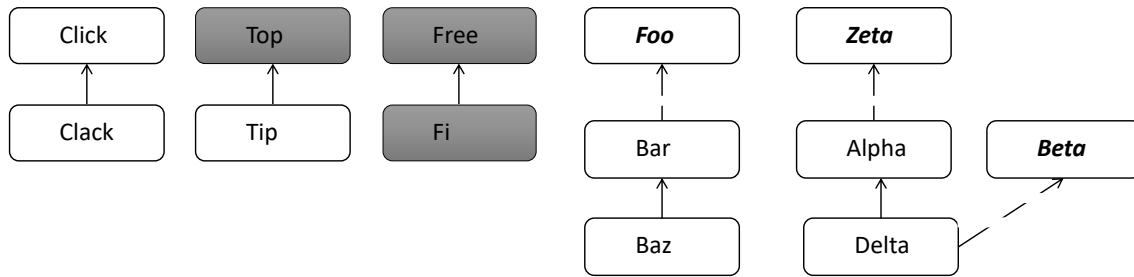
(B) Uma classe pode implementar somente uma interface ao mesmo tempo.

(C) Uma classe pode implementar uma interface ou ser subclasse de outra classe qualquer, mas não ambos simultaneamente.

(D) A construção de um método que pode levantar uma exceção, cuja instância é uma subclasse de *java.lang.RuntimeException*, não exige tratamento obrigatório por parte do programador dentro daquele método.

(E) Objetos da classe *java.lang.String* têm comportamento otimizado para permitir que seu valor seja alterado sempre que necessário, liberando imediatamente a memória usada pelo conteúdo anterior.

3. Dada a estrutura abaixo, escreva as declarações para as estruturas de classes de 1 a 5:



Exemplo:

```

1) public class Click {}
   public class Clack extends Click{}
  
```

4. Analise o seguinte programa e justifique sua execução/saída:

```

import java.io.IOException;
public class Questao04
{
    public static void main( String args[] )
    {
        try
        {
            throw new IOException();
        } // fim do try
        catch ( Exception exception ) // exceção de superclasse
        {
            exception.printStackTrace();
        } // fim do catch
        catch ( IOException ioException ) // exceção de subclasse
        {
            System.err.println( "IOException" );
        } // fim do catch
    } // fim do main
} //fim da classe
  
```

5. Marque a única alternativa FALSA sobre exceções em Java:

- Em Java, todos os objetos que podemos dar throw e catch são de classes derivadas direta ou indiretamente de Throwable.
- As exceções que são classes-base de RuntimeException não precisam ser tratadas.
- Não é natural ocorrerem exceções do tipo Error em Java. Exceções deste tipo representam erros na JVM e não devem ser capturadas.
- No try-catch, não é necessário capturar as exceções mais específicas primeiro.
- O bloco finally no try-catch-finally sempre é executado.