UNIVERSIDADE DE AVEIRO

DEPARTAMENTO DE ELETRÓNICA, TELECOMUNICAÇÕES E INFORMÁTICA

ATP1 de Programação Orientada a Objetos

23 de março de 2018

Duração: 1h00

Nome_____ Nº mec._____

I. [7] Relativamente às perguntas 1 a 14, assinale na tabela seguinte um X na coluna "V" para as declarações que estão corretas e na "F" para as que estão incorretas. Note que estas questões têm por base a linguagem Java. Cada uma destas perguntas vale 0.5 valores e cada resposta errada desconta 0,25 valores. Questões não respondidas valem 0.

	V	F
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		

1. Devido à utilização da máquina virtual de Java (JVM), é possível executar o mesmo programa em qualquer plataforma de hardware e software que possua uma versão da JVM.
2. Existem 8 tipos de dados primitivos em Java.
3. A conversão de um valor com maior capacidade de armazenamento para um valor com menor capacidade de armazenamento é automática.
4. Os métodos da classe string do tipo `replace`, `substring`, `toLowerCase`, etc. constroem e devolvem uma String nova que inclui as modificações pretendidas.
5. Vários objetos de uma classe em Java têm o mesmo comportamento mas podem guardar dados diferentes.
6. O parâmetro de visibilidade `private` permite que um atributo ou método possa ser acedido fora da classe onde está implementado.
7. As variáveis estáticas, ou variáveis de classe, são comuns a todos os objetos dessa classe.
8. Não é possível definir múltiplos construtores numa classe, mesmo que tenham argumentos diferentes.
9. Para invocar um método estático de uma classe temos que instanciar um objeto dessa classe.
10. Na composição de classes, criamos objetos de outras classes dentro da classe nova.
11. A expressão `c1 == c2` verifica se as duas variáveis referenciam objetos com o mesmo conteúdo.
12. Uma classe em Java pode derivar de múltiplas classes.
13. A classe derivada tem acesso a todos os membros da classe base, mesmo que sejam privados.
14. Na herança de classes criamos um tipo novo que é uma extensão da classe existente, herdando a interface da classe existente, mas adicionando código novo para acrescentar mais funcionalidades.

II. [8] Considere os programas seguintes e indique o que é impresso no terminal (use as linhas vazias que se seguem aos programas). Não existem espaços no conteúdo a ser impresso.

```
public class XPTO1 {
  public static void main(String[] args) {
     String x = "00-AA-00";
     int n = 0, p;
     String k = x + ':' + 6 + 'k';
     System.out.println(k);
     String y[] = x.split("-");
     System.out.println(y.length);
     for(String s : y){
        n = n + s.length();
     }
     System.out.println(n);
     String z = new String();
     for(p = 0 ; p < x.length() ; p++){
        char c = x.charAt(p);
        if(c != '-'){
          z = z + c;
        }
     }
     System.out.println(z);
     System.out.println(p);
  }
}
```

Resultado da execução do programa:

Nome_____ Nº mec._____

```
class MyClass{
 private String x, y;
 private static int z = 10;
 public MyClass(){
        this.x = "qqcoisa";
        this.y = "qqcoisa10";
        z = z + 10;
 }
 public MyClass(String s){
        this.x = s;
        this.y = s + z;
        z = z + 10;
        System.out.println("New=" + z);
 }
 public String f(MyClass o){
        if(this.x.compareTo(o.x) < 0)
          return this.x;
        else
          return o.y;
 }
 public static int g(){
        return z;
 }
 public String toString(){
        return this.x + "-" + this.y;
 }
}
public class XPTO2 {
 public static void main(String[] args) {
  MyClass obj1 = new MyClass("poo");
  MyClass obj2 = new MyClass("prog");
  System.out.println(obj1.f(obj2));
  System.out.println(obj2.f(obj1));
  System.out.println(MyClass.g());
  System.out.println("" + obj1 + "," + obj2);
 }
}
```

Resultado da execução do programa:

_____

_____

_____

_____

_____

_____

III. [5] Considere as classes declaradas abaixo e o programa de teste (tabela à esquerda). Tenha em atenção o que foi impresso depois da execução do programa (tabela à direita) e inclua o código necessário nos espaços em branco para que seja possível a sua execução. Não acrescente métodos nem atributos.

```
class MyClass1{
   private double s;
   public MyClass1(double s){

      _____________ = s;}
   public void f(){

      this.s += _____________;}
   public double getS(){

      return ___________________;}

   public String _________________(){

      return ____________________________________; }
}
class MyClass2 extends MyClass1{
   private int x;
   public MyClass2(double s, int x){

      _________________;

      _________________ = x;}
   public MyClass2(){

      ____________________;

      ____________________ = 5;}
   public void f(){
      ____________________;

      this.x += ____________________;}

   public double _________________(){
      return this.x;}

   public String ____________________(){

      return ____________________________________________________;}
}
```

```
public class XPTO3 {
  public static void main(String[] args) {
       MyClass1 obj1 = new MyClass1(25.30);
       System.out.printf("%.1f\n", obj1.getS());
       MyClass1 obj2 = new MyClass2(10.0, 5);
       System.out.printf("%.1f\n", obj2.getS());
       MyClass2 obj3 = new MyClass2();
       System.out.printf("%.1f\n", obj3.getX());
       System.out.printf("%.1f\n", obj3.getS());
       System.out.println(obj1);
       System.out.println(obj2);
       System.out.println(obj3);
  }}
```

```
terminal> java XPTO3

25.3
10.0
5.0
5.1
MyClass1:25
MyClass2:10.0,5
MyClass2:5.1,5
```