UNIVERSIDADE DE ÉVORA
ESCOLA DE CIÊNCIAS E TECNOLOGIA

Licenciatura em Engenharia Informática e outras

Disciplina: Programação II

Teste 2

2017.04.07

1. (8v) Considere que temos as seguintes declarações:

```
int m, n, i = 3, j = 4, k = 5;
float v, w, x = 34.5f, y = 12.25f;
```

Partindo do princípio que cada instrução é a única a ser executada após as declarações, indique o valor que é colocado em cada variável abaixo, ou explique porque é que a afetação é inválida, caso seja esse o caso.

(a) `v = x / i;`
(b) `w = Math.ceil(y) % k;`
(c) `n = (int) x / y * i / 2;`
(d) `m = n + i * j;`
(e) `n = k /(j * i) * x + y;`
(f) `i = i + 1;`
(g) `w = float(x + i);`
(h) `x = x / i / y /·j;`

2. (3v) Usando instruções `while` e evitando a instrução `for`, apresente código equivalente a:

```
for (i=0; i<10; ++i) {
    Extra.terrestre (i);
}
```

3. (2v) Repita o exercício anterior usando as instruções `if` e `do..while`.

4. Considere as classes:

```
abstract class Forma {
    double area () { return 0; }
    double perimetro () { return 0; }
    String toString ();
};

class Quadrado extends Forma {
    double lado;
    void Quadrado (double lado) { this.lado = lado; }
    double area () { return lado * lado; }
    double perimetro () { return 4 * lado; }
    String toString () { return "Quadradro␣de␣lado␣" + lado; }
};

class Circulo extends Forma {
    double raio;
    void Circulo (double raio) { this.raio = raio; }
    double area () { return Math.PI * raio * raio; }
    double perimetro () { return 2 * Math.PI * raio; }
```

```
20      String toString () { return "Circulo de raio " + raio; }
21  };
```

(a) (1v) Explicite a relação entre as 3 classes mencionadas, i.e. herança, características das classses, etc. Para isso, desenhe um diagrama com estas classes.

(b) (2v) Defina uma classe `Retangulo` com o posicionamento nesta hierarquia e comportamento esperados.
O construtor a definir deverá aceitar dois parâmetros do tipo `double`: um para a largura e outro para a altura. Inclua definições para os métodos `area()`, `perimetro()` e `toString()` apropriados para um retângulo.

(c) (1v) Se quisesse definir o `Quadrado` em termos do `Retangulo`, o que é que faria?

(d) (1v) Considere que `f` é uma `Coleção` de formas, e que pretendemos saber qual delas a que tem a maior área. Para isso definimos um método `maiorForma()`. Este método poderá recorrer a todas mensagens que se podem enviar a objetos da classe `Forma`.
Sabe-se que a classe de `f` permite aceder a todas as formas que integram a coleção, recorrendo ao método **`nextForma()`**, que retorna cada uma das formas, em chamadas sucessivas.
Pretendemos que `maiorForma()` retorne a `Forma` do seu conjunto que tiver **maior área**, por exemplo, se tivermos:

```
1  Coleção f;
2  Forma maior;
3  f.acrescenta (new Quadrado (3.0)); // quadrado de lado 3
4  f.acrescenta (new Circulo (2.0));  // circulo de raio 2
5  f.acrescenta (new Retangulo (1.5, 2.5)); // retangulo de 1.5 x 2.5
6  maior = f.maiorForma ();
7  System.out.println ("A maior forma é " + maior);
```

Diga qual será o output deste troço de código. Assuma que as declarações de classe estão todas feitas e que o método `toString()` foi definido para as subclasses de `Forma`.

(e) (2v) Complete a definição da função `maiorForma`.

```
1   class Coleção {
2     ...;
3     Forma nextForma ();
4
5     Forma maiorForma () {
6       Forma aForma;
7       ...; // RESPONDA AQUI
8       return aForma;
9     }
10  }
```