Teste Tipo de “Introdução à Programação (A)”

Duração: 1:30H

Instruções importantes: Responda a todos os grupos em folhas separadas.
Identifique todas as folhas com o seu número e nome.
Antes de começar a resolver um exercício, leia o enunciado do princípio até ao fim.

I – Determine o valor que cada uma das variáveis declaradas no bloco deverá conter, no fim de cada linha, e quando a execução do mesmo terminar.

a)

```
{
    int a;
    int b;

    a = 0;
    b = 1;
    a = b + a;
    b = b – a;
}
```

b)

```
{
    int a;
    boolean b;

    a = 0;
    b = (a >= 0);
    if (b)
        b = ! b;
      else
       b = (a + 2 < 0);
    a ++ ;
}
```

c)

```
{
    int a;
    int b;
    int c;
    int m;

    a = 1;
    b = a+2;
    c = a-2;

    if(a>b) {
       if(b>c)
          m = c;
       else
          m = b;
    } else if (b<c)
            m = b;
         else
            m = c;
}
```

II – Considere que necessita de implementar uma classe para representar porta-moedas. Neste exercício, **não terá** que programar, mas sim que definir o conjunto de operações (métodos e construtores) e as variáveis e constantes que achar necessárias para implementar a memória (ou estado) de cada objecto.

Cada porta moedas guarda um certo número de moedas, que podem ser de 1 cêntimo, 10 cêntimos ou 1 Euro. O porta-moedas deve ser capaz de receber moedas dos vários tipos e permitir a consulta do seu estado (moedas e valor guardado). Deve também ser capaz de efectuar o pagamento de uma certa importância, indicada por um único valor. O pagamento só poderá ser efectuado se o porta-moedas tiver o dinheiro suficiente. Se no porta-moedas não houver moedas para fazer o pagamento certo, supõe-se que o pagamento será feito com uma importancia superior, o que dará lugar a troco, o qual será guardado de novo no porta-moedas.

Na sua solução, indique que métodos são modificadores e que métodos são de consulta. Comente a declaração de cada método com uma explicação que ajude a perceber bem a sua finalidade.

III – Considere a seguinte classe em Java.

```
class ShotGun {

static final String Str1 = "Bang!";
static final String Str2 = "Click";
static final int DEFAULT = 1;
int bullets;

ShotGun(int numBullets) {
  bullets = numBullets;
}

ShotGun() {
  bullets = DEFAULT;
}

int load() {
  return bullets;
}

void loadGun(int numBullets) {
  bullets = bullets + numBullets;
}

String shoot() {
  if(bullets > 0) {
    bullets = bullets - 1;
    return Str1;
  } else
    return Str2;
}
}
```

a) Explique qual poderá ser a funcionalidade e objectivo da classe.

b) Considere a seguinte sequência de chamada de métodos a vários objectos da classe `ShotGun`. Indique, para cada chamada que devolva um resultado, que resultado é esse.

```
ShotGun g1 = new ShotGun(2);
```

```
ShotGun g2 = new ShotGun();
g1.shoot()
g1.shoot()
g1.load()
g2.shoot()
g2.loadGun(2);
g1.shoot()
g2.shoot()
g2.loadGun(g1.load());
g1.load()
```

IV – Este exercício destina-se a resolver um pequeno problema de programação. Pretende-se desenvolver em Java uma classe `Number` cujos objectos representam números com exactamente quatro dígitos numa certa base entre 2 e 10. Quando um objecto número é criado é indicada a sua base no construtor.
Cada objecto número possui as seguintes operações:

```
void setDigit1(int digit)
```

indica o primeiro dígito do número (menos significativo)

```
void setDigit2(int digit)
```

indica o segundo dígito do número

```
void setDigit3(int digit)
```

indica o terceiro dígito do número

```
void setDigit4(int digit)
```

indica o quarto dígito do número (mais significativo)

```
int getValue()
```

indica o valor do número, na sua base

```
int getValueBase10()
```

indica o valor do número, na base 10 (decimal)

```
void increment()
```

soma 1 ao número

```
void decrement()
```

subtrai 1 ao número

```
boolean symmetric()
```

indica se o número é capicua quando escrito na sua base (ou seja, se é o mesmo número quando escrito de trás para a frente, na sua base)

Nota importante: quando um número for criado na base ***b***, pode assumir que os valores passados nos métodos setDigit é sempre entre 0 e ***b***-1.

Exemplo 1)

```
Number n = new Number(2);
n.setDigit1(0);
n.setDigit2(0);
n.setDigit3(0);
n.setDigit4(1);
n.getValue()
1000
n.getValueBase10()
8
n.increment();
n.increment();
n.getValue()
1010
n.getValueBase10()
10
n.symmetric()
false
```

Exemplo 2)

```
Number m = new Number(3);
n.setDigit1(1);
n.setDigit2(2);
n.setDigit3(0);
n.setDigit4(0);
n.getValue()
21
n.getValueBase10()
7
n.increment();
n.getValueBase10()
8
n.symmetric()
true
```

Programe em Java a classe `Number`.

****

Este teste tipo ilustra apenas a estrutura do primeiro teste e destina-se a dar uma ideia do que os alunos podem esperar.

****