Nome: Nº

1º Teste

Programação Orientada por objetos
2014/15
Aulas Teórica de 16 de Abril de 2015
Departamento de Informática
Universidade Nova de Lisboa

(duração 1h30 minutos)

– Sem consulta –

Leia com atenção a informação constante desta página, enquanto espera a indicação do docente para começar a resolução do teste.

Este enunciado é composto por:
Uma página de Rosto (esta)
14 páginas de enunciado.
1 boletim/folha de respostas

O teste é composto por perguntas de resposta múltipla.

Atenção:

Resposta múltipla para seleção de apenas uma alínea
Uma resposta errada desconta um quarto do valor dessa pergunta na cotação total do teste
(VALOR DA PERGUNTA) / 4

Todas as perguntas de resposta múltipla devem ser respondidas no boletim de resposta fornecido em conjunto com este enunciado assinalando com uma bola a cheio a opção pretendida. (para evitar enganos, sugere-se o uso de lápis e/ou a folha do enunciado, assinalando-se no fim a resposta final a caneta preta ou azul).

Se o boletim não tiver o número de aluno nem estiver assinado pelo próprio não será considerado para avaliação.

A resolução final no boletim tem de ser feita a caneta azul ou preta (qualquer outra cor levará a anulação).

No fim de 1h30 minutos de teste o docente **recolherá o enunciado e boletim de respostas**.

Boa Sorte!

I - Conceitos POO

1. A noção de objecto é uma das noções cruciais à Programação Orientada a Objectos. Um objecto é (escolha a afirmação verdadeira):

[A] Um conjunto de operações definidas como invocáveis a partir do exterior da entidade e outras que são declaradas como apenas acessíveis de outras internas

[B] Uma garantia de que todas as entidades do mesmo tipo são iguais tanto em estrutura e comportamento

[C] Uma entidade ou módulo de computação básico e único com: identidade única, estado interno e conjunto de operações (únicas a aceder directamente o estado interno do objecto)

[D] Um conceito de programação que permite vencer os problemas de programação provocados pela abstração, encapsulamento e modularidade

[E] Nenhuma das anteriores

II - Java (Diversos)

2. Considere o seguinte trecho de código relativo a uma classe ***BankAccount***:

BankAccount.java

```
public class BankAccount {

        private int amount;    //dinheiro disponível na Conta Bancária

          … // Resto variáveis e construtor

        public void mistery(double amount) {

                this.amount = this.amount - amount;
                this.amount += amount;

          }
          ... // Outros métodos
}
```

O que faz o método ***mistery***?

[A] Apenas se amount for 0.00, a variável de instância (ou de estado) amount não é alterada.

[B] Coloca dinheiro na conta bancária variável de instância (ou de estado) amount, ficando com um valor na variável de instânica amount superior ao que tinha antes de executar o método.

[C] Retira dinheiro da conta bancária representado pela variável de instância (ou de estado) amount, ficando com um valor na variável amount inferior ao que tinha antes de executar o método.

[D] O valor da variável de instância (ou de estado) amount antes e depois do método ser executado mantem-se igual.

[E] Nenhuma das anteriores.

3. Considere a seguinte implementação da class Square:

Square.java

```
public class Square {
        private int sideLength;
        private int area;

        public Square(int length) {
                sideLength = length;
                area = sideLength * sideLength;
        }

        public void grow(){
                sideLength++;
        }

        public int getarea() {
                return area;
        }
}
```

[A] A classe pode retornar áreas do retângulo erradas.

[B] A classe dará erros no eclipse e uma possível main não conseguirá testá-la.

[C] Ambos o construtor e o método grow estão a alterar a variável sideLength do rectângulo, algo que não é permitido em Java.

[D] A classe está bem implementada.

[E] Nenhuma das anteriores.

4. Considere as seguintes classes:

BankAccount.java

```
public class BankAccount {
        private int balance;

        public BankAccount (int balance_aux) {
             balance=balance_aux;
        }

        … //outros métodos
}
```

BankRobber.java

```
public class BankRobber {
    public static void main(String[] args) {
        BankAccount momsSavings = new BankAccount(1000);
        momsSavings.balance = 0;
    }
}
```

[A] O código da main resulta em roubar a bank account da mãe mas poderia estar melhor implementado.

[B] O construtor da classe BankAccount está mal implementado.

[C] O código da main não resulta.

[D] O código da main resulta em roubar a bank account da mãe e está bem implementado.

[E] Nenhuma das anteriores.

5. Assumindo que a variável s1 e s2 do tipo String tem “POO B” e “teste”, respectivamente, como posso escrever exatamente a seguinte mensagem: “POO B - teste01”?

[A] System.out.println(s1 + s2 + “1”);

[B] System.out.println(s1 + s2 + 01);

[C] System.out.println(s1 + s2 + “01”);

[D] System.out.println(s1 + “ - “ + s2 + “01”);

[E] System.out.println(s2 + s1 + “01”);

6. Suponha que in é um objecto do tipo Scanner que lê do System.in, e que o seu programa chama:

```
String name = in.next();
```

Qual o valor de name se o utilizador escrever “João Q. Public”?

[A] “Public”

[B] null

[C] “João Q. Public”

[D] “Q. Public”

[E] “João”

7. Qual o output ao executar o seguinte trecho de código?

```
for (int i = 1; i < 5; i++)
        System.out.println(i);
```

[A] 1
2
3
4

[B] 1
2
3
4
5

[C] 12345

[D] 1234

[E] 1 2 3 4 5

8. Qual o output esperado para a execução do seguinte código?

```
String s = "B";
int i = 20;
if (i/2 == 10) {
        s += "OO";
        if ( i/4 == 2 || i/4 == 3 )
            s += "P";
         else
            s += "K";
} else
       s = "POO";
System.out.println(s);
```

[A] "B"

[B] "BOOP"

[C] "POO"

[D] "BOOK"

[E] "BOO"

9. Sejam s1 e s2 duas strings tais que:

```
String s1 = “Teste”;
String s2 = “POO B”;
```

Como se verifica se as duas strings têm o mesmo conteúdo?

[A] s1 == s2

[B] s1 != s2

[C] s1.equals(s2)

[D] s1.length() == s2.length()

[E] !s1.equals(s2)

10) Considere o seguinte trecho de código:

Test.java

```
...
Circle c1 = new Circle(0, 0, 5);
Circle c2 = c1;
Circle c3 = new Circle(0, 0, 5);

if (c1 == c2)
        System.out.print(“A”);
else System.out.print(“B”);

if (c1.equals(c3))
        System.out.print(“C”);
else System.out.print(“D”);
...
```

Qual o output de executar este mesmo trecho de código?

[A] AC

[B] BC

[C] AD

[D] ABCD

[E] BD

II - Interpretador de comandos

11) Considere o seguinte interpretador de comandos:

Main.java

```
public class Main {

    //Opções do menu
    private static final String CMD_INSTALAR = "I";
    private static final String CMD_DESINSTALAR = "D";
    private static final String CMD_MOVE = "M";
    private static final String CMD_SAIR = "S";


    public static String processCommand(Scanner in) {

        String option;
        System.out.println("I – instala equipamento");
        System.out.println("D – desinstala equipamento");
        System.out.println("M – move o equipamento ");
        System.out.println("S - sair");
        option = in.nextLine();
        return option;
   }

   // Outros métodos relevantes
   (...)

   public static void main(String[] args) {
       Scanner in = new Scanner(System.in);
       String command = processCommand(in);

       while (!command.toUpperCase().equals(CMD_SAIR)) {
           switch (command.toUpperCase()) {

               case CMD_INSTALAR:
                   (...) // invocação do método respectivo
               break;

               case CMD_DESINSTALAR:
                   (...) // invocação do método respectivo
               break;

               case CMD_MOVE:
                   (...) // invocação do método respectivo
               break;
           }
       }
       System.out.println("Terminado.");
       in.close();
   }
}
```

Qual o problema com este código quando executado?

[A] Entra num ciclo infinito apenas no caso de não se inserir logo no início do programa a opção sair

[B] Entra sempre num ciclo infinito

[C] Dá erro pois falta a opção default no switch

[D] Não pode ter um break no último case do switch

[E] Nenhuma das anteriores

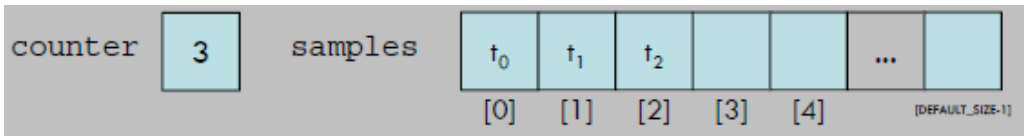
Grupo II - Vectores

12) Com respeito ao exercício do WeatherStation apresentado nas aulas teóricas da disciplina, considere o seguinte excerto da classe WeatherStation:

WeatherStation.java

```
…..

private int counter;
private double [] samples;

public void insertAt (double temp, int pos) {
        for (int i=counter-1; i >= pos; i--)
                samples[i+1] = samples[i];

        samples[pos] = temp;
        counter++;
  }

  public void removeAt(int pos){
        for(int i=pos; i < counter-1; i++)
                samples[i] = samples[i+1];

        counter--;
  }

....
```

Analise os dois métodos presentes acima e responda à seguinte questão:



Tendo em consideração o estado inicial apresentado acima das variáveis counter e samples, indique qual o resultado das mesmas após ser invocado os seguintes métodos pela ordem apresentada:

```
insertAt(20, 2);
removeAt(3);
```

[A] counter = 4 samples = (t0, 20, t1, t2, ⋯)

[B] counter = 4 samples = (t0, t1, t2, 20, ⋯)

[C] counter = 3 samples = (t0, t1, t2, ⋯)

[D] counter = 4 samples = (t0, t1 , 20 , t2 , ⋯)

[E] counter = 3 samples = (t0, t1, 20, ⋯)

13) Considere a classe:

MyClass.Java

```
public class MyClass {
        private static final int DIM = 6;
        private int [] tt1;

        public MyClass(){
                ...
        }

        public void Init(boolean ctx){
                if(ctx){
                        for(int i=1; i<DIM;i++)
                                tt1[i]=i*(i-1);
                }
                else{
                        for(int i=DIM; i>=1; i--)
                                tt1[i]=i%2;
                }

        }


}
```

Qual o código a incluir no construtor de MyClass para inicializar o vector tt1 com DIM posições?

[A] tt1 = new int[DIM];

[B] tt1 = int[DIM];

[C] tt1 = DIM*new int

[D] tt1 = new int(DIM);

[E] Nenhuma das anteriores

14) Admitindo que o construtor mencionado na pergunta anterior(13) está correcto, e tendendo à classe MyClass, podemos afirma que após a execução da seguinte instrução:

```
MyClass t1 = new MyClass();
```

o conteúdo do vector tt1 em t1 é:

[A] tt1 = (DIM,DIM,DIM,DIM,DIM,DIM)

[B] tt1 = (0,0,0,0,0,0)

[C] tt1 = (NULL,NULL,NULL,NULL,NULL,NULL)

[D] Os valores no vector são indefinidos

[E] Nenhuma das anteriores

15) Atendendo à classe MyClass mencionada na pergunta 13, sabendo que se carregou o objecto t1 com o vector tt1=(0,0,0,0,0,0), e que se executa a seguinte instrução:

```
t1.Init(true);
```

qual será agora o novo conteúdo do vector tt1 em t1?

[A] tt1 = (0,0,0,12,72,320)

[B] tt1 = (0,0,2,6,12,20)

[C] tt1 = (0,2,6,12,20,30)

[D] tt1 = (0,1,0,1,0,1)

[E] Nenhuma das anteriores

16) Considere o seguinte código:

```
public void myfunc(int [] tt1, int dim,int a, int b){
        for(int i=a; i<b; i++){
            for (int j=i+1; j<=b; j++){
                if(tt1[i]>tt1[j]){
                    int temp = tt1[i];
                    tt1[i]=tt1[j];
                    tt1[j]=temp;
                }
            }
        }
    }
```

Admitindo que é chamada a função anterior myfunc com argumentos de entrada tt1=(5,4,3,2,1,0), a=2, b=5 e dim=6
Qual será o novo valor de tt1?

[A] tt1 = (0,1,2,3,4,5)

[B] tt1 = (5,4,0,1,2,3)

[C] tt1 = (5,0,1,2,3,4)

[D] tt1 = (0,0,0,0,0,0)

[E] Nenhuma das anteriores

17) Considere o seguinte código:

```
public int myfunc(int [] tt1, int dim){
        int num1=tt1[0];
        for(int i=1;i<dim-1;i++){
            if(tt1[i]>num1)
                num1=tt1[i];
        }
        return num1;
    }
```

Admitindo que é chamada a função anterior myfunc com argumentos de entrada tt1=(0,1,2,3,4,5) e dim=6 qual o valor retornado?

[A] 0

[B] 4

[C] A função nunca retorna

[D] 5

[E] Nenhuma das anteriores

18) Considere o seguinte código:

```
public int myfunc(int [] tt1, int dim, int num){
                        int i=0;
                        int val=0;
                        while(i<dim){
                                if(tt1[i]>num)
                                        val+=tt1[i];
                                i++;
                        }
                        return val;
        }
```

Admitindo que é chamada a função anterior myfunc com argumentos de entrada tt1=(0,1,2,3,4,5), dim=6 e num=10 qual o valor retornado?

[A] 1

[B] 32

[C] 0

[D] A função tem erros de compilação

[E] 12

Grupo III - Iteradores

19) Considere o iterador desenvolvido para uma classe **ContactBook** que mantém uma colecção de **Contact**:

ContactIterator.java

```
public class ContactIterator {

     private Contact[] contacts;
     private int counter;
     private int currentContact;

    public ContactIterator(Contact[] contacts, int counter) {
            this.contacts = contacts;
            this.counter = counter;
            currentContact = 0;
    }

    public boolean hasNext() {
             return (currentContact < counter);
    }

   public Contact next() {
            currentContact++;
             return contacts[currentContact];
   }
}
```

Qual o problema com o código anterior?

[A] o método next salta sempre a posição 0 do vetor

[B] o construtor não pode receber o vetor contacts como argumento

[C] o método hasNext precisa de mais condições para verificar se há próximo elemento

[D] é necessário um outro construtor sem argumentos

[E] Nenhuma das anteriores

20) Ainda sobre o ContactBook, assumindo que o iterador está bem implementando, o que podemos afirmar em relação a correr o trecho de código seguinte na respectiva main:

Main.java

```
...
ContactIterator cit = cBook.iterator();
            while (cit.hasNext()) {
                        Contact c1 = cit.next();
                        Contact c2 = cit.next();
                        System.out.println(c1);
                        System.out.println(c2);
            }
...
```

Escolha a opção correta para todas as situações possíveis (i.e., considerando o caso de não ter contactos, ter 1 contacto ou com muitos contactos na agenda):

[A] o programa não consegue correr

[B] o programa corre mas por vezes não funciona

[C] o java não permite invocar duas vezes seguidas o método next do iterador

[D] o programa corre e disponibiliza sempre output

[E] Nenhuma das anteriores