UAlg FCT
UNIVERSIDADE DO ALGARVE
FACULDADE DE CIÊNCIAS E TECNOLOGIA

Algoritmos e Estruturas de Dados

Exercícios

Capítulo 1. Fundamentos da Linguagem Java

1.1 Linguagem de Programação Java

1.1.1 Considere o seguinte código Java:

```
String string1 = "hello";
String string2 = string1;
string1 = "world";
System.out.println(string1);
System.out.println(string2);
```

Indique o que é impresso no ecrã quando o código é executado.

1.1.2 Descarrege os ficheiros ContaBancaria.Java e SimpleMain.Java disponibilizados na tutoria.

a)Crie um projeto no seu IDE (IntelliJ ou Eclipse) com ambos os ficheiros dentro da mesma package. Crie uma *command line application* para teste, e execute o método *Main*.

b) Leia com atenção os comentários do código fornecido.

c) Desenhe um diagrama de caixas e ponteiros, ilustrando as zonas de memória do *Stack* e do *Heap*, e os objetos computacionais que são criados, para as primeiras 3 linhas de código do método *Main*.

d) Faça algumas alterações ao método Main fornecido, por exemplo criando novas contas, e fazendo mais operações de levantamento e depósito, e imprima o resultado no ecrã.

e) Acrescente um novo construtor à classe ContaBancária que permite criar uma nova conta sem especificar um NIB. Nestes casos, o NIB é determinado como sendo o próximo número inteiro que ainda não tenha sido usado anteriormente. Sugestão: use um campo estático interno para guardar o último NIB utilizado.

f) Experimente a criação de novas contas bancárias usando o construtor da alínea anterior, e imprima o NIB das contas criadas.

1.1.3 Considere o seguinte programa, tendo em conta que a classe Conta se encontra implementada:

Foram criados 4 objectos da classe Conta no Heap. Indique quais ficam marcados para remoção

```
public static ContaBancaria copiaConta(Conta c)
{
    ContaBancaria c1 = new ContaBancaria(c.getNIB(),c.getSaldo()); //Obj1
    ContaBancaria c2 = new ContaBancaria(c1.getNIB(),c1.getSaldo());//Obj2

    return c1;
}

public static void main(String[] args) {

    ContaBancaria c1;
    ContaBancaria c2;
    ContaBancaria c3;

    c1 = new ContaBancaria(2039958123,200); //Obj 3
    c2 = copiaConta(c1);
    c1 = new ContaBancaria(1231488393,150); //Obj 4
    c3 = c1;
    c1 = null;
    ...
}
```

no fim do código apresentado (assumindo que o Garbage Collector é activado nessa altura).

1.1.4 Considere um programa em Java com classe Main, e com um método main que recebe um argumento, imprime a String "Hello World!" para o ecrã, e imprime na linha seguinte o argumento recebido.

a) Implemente e execute o programa usando o IntelliJ ou o Eclipse..
b) Compile e execute o programa usando a linha de comandos.

1.1.5 Considere a função que calcula o factorial de um número inteiro n recebido. Lembre-se que n! = n x (n-1)!

a) Implemente a função de forma recursiva.
b) Implemente a função de forma iterativa.

1.1.6 Implemente uma função filtraLista, que dada uma lista de números inteiros, devolve uma lista apenas com os números inferiores a 10.

1.1.7 Implemente uma função filtraLista, que recebe uma lista de objectos genéricos do tipo *T*, e um predicado (i.e. uma função que retorna *true* ou *false*) com um argumento genérico do tipo *T*, e devolve uma lista apenas com os elementos da lista que satisfazem o predicado.

1.1.8 Considere o problema de ser reordenar de forma aleatória (baralhar) os elementos de um array de objectos. A forma mais simples de resolver este problema é escolher um elemento de forma aleatória, removê-lo para outro array, reajustar o array, e voltar a repetir o processo até não restarem elementos. No entanto esta abordagem não é nada

eficiente. Para resolver este problema Sattori propôs um algoritmo que vai usando o lado direito do array (o fim deste) para ir guardando os elementos que vão sendo removidos das restantes posições do array. O programa seguinte contém uma implementação do algoritmo de baralhar de Sattolo.

```
public static void sattoloShuffle(Object[] a) {
    int n = a.length;
    for (int i = n; i > 1; i--) {
        // choose index uniformly in [0, i-1[
        int r = (int) (Math.random() * (i-1));
        Object swap = a[r-1];
        a[r-1] = a[i-1];
        a[i-1] = swap;
    }
}

public static void main(String[] args) {
    String[] a =
{"isto","é","uma","lista","de","palavras","a","baralhar"};
    sattoloShuffle(a);
    for (int i = 0; i < a.length; i++)
        System.out.println(a[i]);
}
```

Infelizmente, o aluno que fez o código cometeu um erro, e o algoritmo não funciona. Utilize o mecanismo de depuração (debugger) do Eclipse para avançar passo a passo e tentar perceber porque é o que algoritmo não funciona. Corriga o problema.

1.1.9 Implemente um programa Java que recebe o nome de um ficheiro como argumento, abre um ficheiro csv (em que as colunas estão separados pelo caracter “,”), e imprime a 2.ª coluna de cada linha, para todas as linhas.

1.1.10 Considere o seguinte código:

```
public class Point2D {
    public int x;
    public int y;

    public Point2D(int x, int y)
    {
        this.x = x;
        this.y = y;
    }

    public int getX() { return this.x; }
    public int getY(){ return this.y; }
}


public class Main {

    public static void main(String[] args) {
        Point2D p = new Point2D(0,0);
        p.x = 4;
        p.y = 5;
    }
}
```

A implementação acima respeita a abstração de dados? Justifique.