Licenciaturas em Engenharia Informática

Modelação e Design

AULAS LABORATORIAIS

FICHA 6 - DIAGRAMAS DE SEQUÊNCIA

1. Considere os códigos (em Java) representados na Figura 1:

a) Relativamente ao código da Figura 1a) construa o diagrama de sequência correspondente à invocação da operação "doOne".

b) Relativamente ao código da Figura 1b) construa o diagrama de sequência correspondente à invocação da operação "doX".

```
public class A {
    private B myB = new B();

    public void doOne() {
        myB.doTwo();
        myB.doThree();
    }
}


class B {
    public void doThree() {
        // ...
    }

    public void doTwo() {
        // ...
    }
}
```

Figura 1a

```
public class Register {
    public void doX()   {
        // ...
        clear();
        // ...
    }

    public void clear() {
        // ...
    }
}
```

Figura 1b

2. Considere o código (em Java) representado na Figura 2. Construa o diagrama de sequência correspondente à invocação da operação "run". Assuam que classe Vector possui um método *add* que permite adicionar uma string que recebe por parâmetro e um método remove que recebe uma string e devolve um valor booleano.

3. Relativamente à seguinte situação:

- Um texto é constituído por várias linhas de texto.
- A fonte de um texto é representada pelo nome da fonte (por exemplo, Arial) e pelo seu tamanho (por exemplo, 12pt).
- A formatação de um texto é representada pela sua fonte, a informação sobre a cor e sobre o alinhamento (esquerda, direita, centrado ou alinhado).
- Um texto pode conter formatações diferentes em zonas diferentes. Uma zona é definida simplesmente através de um ponto inicial e final no texto (por exemplo).

a) Considere que a classe "Texto" tem uma operação que devolve a Formatação num determinado ponto do texto. Construa o diagrama de sequência correspondente.

b) Considere que a classe "Texto" tem uma operação que permite modificar a formatação entre dois pontos de texto. Construa o diagrama de sequência correspondente.

c) Considere que a classe "Texto" tem uma operação que permite acrescentar, ou inserir, uma linha de texto com um determinado formato. Construa o diagrama de sequência correspondente.

```
import java.util.Vector;

public class Driver {
    private StringContainer b = null;

    public static void main(String[] args){
        Driver d = new Driver();
        d.run();
    }

    public void run() {
        b = new StringContainer();
        b.add("One");
        b.add("Two");
        b.remove("One");
    }
}

class StringContainer {
    private Vector v = null;

    public void add(String s) {
        init();
        v.add(s);
    }

    public boolean remove(String s) {
        init();
        return v.remove(s);
    }

    private void init() {
        if (v == null)
            v = new Vector();
    }
}
```

Figura 2

4. Dadas as seguintes situações:

a) Um veículo tem um proprietário. Um proprietário pode ter vários veículos. Considere que na classe "Proprietario" está definida uma operação para imprimir a informação (por exemplo, a matrícula) de todas as viaturas de que é proprietário. Construa o diagrama de sequência correspondente.

b) Uma pessoa tem um nome e um grupo de amigos. Considere que na classe "Pessoa" está definida uma operação para listar todos os amigos, e todos os amigos dos seus amigos, e todos

os amigos dos amigos dos seus amigos (e assim sucessivamente) dessa pessoa. Construa o diagrama de sequência correspondente.

c) Uma corrida consiste num conjunto de veículos que irão competir numa pista. Os veículos podem ser carros ou motas. Cada veículo tem um piloto. A pista é constituída por uma sequência de segmentos interligados. Cada um destes segmentos tem um determinado comprimento e um ângulo de viragem. A corrida tem uma determinada duração, medida em "número de voltas". Considere que na classe "Corrida" está definida uma operação que devolve a distância total dessa corrida. Elabore dois diagramas de sequência diferentes para essa operação.

5. Considere uma aplicação para o envio de correio eletrónico. Quando o utilizador seleciona a opção "Send/Receive", a primeira operação a executar pelo programa é enviar todas as mensagens contidas na pasta "Outbox" (através do método *enviaMensagens*). Após o envio das mensagens, com sucesso, estas mensagens são colocadas na pasta "SentMessages". De seguida, verifica se, no servidor, existe correio novo (através do método v*erificaNovoCorreio*) e, dependendo da resposta, irá transferir estas novas mensagens para a pasta "InBox".

Represente os objetos adequados e o diagrama de sequência correspondente.