DSI 03/04 — Ficha Prática - *Diag. de Estado / Diag. de Actividade*

Stack

Considere uma classe `StackBilhete` que implementa os seguintes métodos:

```
public void push(Bilhete b);
public void pop();
public Bilhete top();
```

com o comportamento esperado de uma Stack.

Escreva um **Diagrama de Estado** que descreva o percurso de um bilhete na Stack.

Encomendas

Considere a seguinte interface Java:

```
public interface Encomenda {
    public void disponível(Vector linhas);
    public void disponível();
    public void envia();
    public void cancela();
    public void adiciona(Vector linhas);
}
```

Uma classe que implemente a interface Encomenda deverá ter o seguinte comportamento:

Após ser criada uma encomenda fica pendente a aguardar que todas as suas linhas sejam satisfeitas.

Existem duas formas de registar que uma (ou mais) linha(s) da encomenda já está/estão satisfeita(s). O método `disponível(Vector linhas)` assinala a satisfação das linhas passadas como parãmetro. O método `disponível()` assinala a satisfação de todas as linhas ainda por satisfazer. Após estar totalmente satisfeita a encomenda passa a poder ser enviada.

O envio da encomenda é registado utilizando o método `envia()`. Após ser enviada a encomenda fica registada como expedida.

Em qualquer momento antes do envio é possível cancelar a encomenda (método `cancela()`). Nesse caso ela fica registada como cancelada.

O método `adiciona(Vector linhas)` pode ser utilizado em qualquer encomenda que não cancelada ou expedida para lhe adicionar novas linhas.

Escreva um **Diagrama de Estado** que represente o comportamento descrito.

Torniquete

Como parte da preparação para o Euro 2004 foi-lhe pedido que desenvolvesse o software de controlo dos torniquetes de validação de acessos aos estádios. Os torniquetes possuem braços metálicos que podem impedir ou permitir a passagem de pessoas conforme estejam bloqueados ou não. Possuem também um leitor de bilhetes para validação dos bilhetes para os jogos.

A solução que vai propor tem o seguinte modo de funcionamento:

Ao ser ligado o torniquete fica bloqueado. Quando é inserido um bilhete válido o bilhete é retido e o torniquete é desbloqueado, ficando nesse estado até que alguém passe, altura em que bloqueia novamente. Quando é inserido um bilhete inválido, faz soar um alarme e o bilhete é devolvido. O sistema mantém o alarme até o bilhete ser retirado do leitor, altura em que volta a estar bloqueado. Enquanto está desbloqueado o torniquete não aceita bilhetes.

O sistema possuirá também um modo de teste para permitir a técnicos analisar o funcionamento do torniquete. Quando em modo de teste os técnicos podem realizar dois tipos de testes: testar os braços e testar o leitor de bilhetes. Para testar os braços o técnico pode bloquear e desbloquear o torniquete. Para testar o leitor de bilhetes o técnico insere bilhetes (que poderão ser válidos ou inválidos) que serão sempre devolvidos. No caso do bilhete ser inválido o sistema soa o alarme até o técnico retirar o bilhete. Os dois tipos de teste podem ser executados em paralelo.

O modo de teste pode ser activado em qualquer altura. Quando sai de modo de teste o sistema fica na situação de bloqueado. O sistema pode ser desligado em qualquer altura.

Escreva um **Diagrama de Estado** que represente a solução proposta.

Ordenação

Considere o seguinte extracto de código Java:

```
public class Lista {
    // variáveis de instância
    private Vector lista;

    // métodos de instância
    public void ordena() {
      int i=lista.size()-1, j, min;
      Comparable c,
      Object o;

      while(i>0) {
        j=0;
        while (j<i) {
          c = (Comparable)lista.get(j);
          o = lista.get(i);
          if (c.compareTo(o)==1) {
            lista.set(i, c);
            lista.set(j, o);
          }
          j++;
        }
        i--;
      }
    }
    ...
}
```

Escreva um **Diagrama de Actividade** que descreva o algoritmo do método `ordena`.

Viagens

Escreva um **Diagrama de Estado** que represente um processo de autorização de viagens tal como descrito de seguida:

> Um formulário de autorização de viagens é normalmente utilizado nas empresas para aprovação de despesas de viagenm dos seus funcionários. O formulário de autorização de viagem passa por vários estados durante o processo de aprovação. Normalmente o funcionário preenche um formulário vazio e envia-o ao director da sua unidade para ser assinado.
>
> Se a quantia for pequena (menos de EUR300.00), o director de serviço assina o formulário e envia-o para a contabilidade para ser processado. Ao receber o formulário a contabilidade emite um cheque a favor do funcionário. Depois de o cheque ser levantado o formulário é arquivado. Se o cheque não for levantado no prazo de 90 dias, o formulário perde a validade.
>
> Se a quantia for elevada (EUR300.00 ou mais), o director de serviço assina o formulário e envia-o para o director financeiro para aprovação. O director financeiro assina o cheque e envia-o para a contabilidade para ser processado.
>
> Obviamente tanto o director de serviço como o director financeiro podem rejeitar a autorização de viagem se não considerarem as despesas aceitáveis. Neste caso o funcionario pode alterar o formulário de modo a incluir mais informação sobre a despesa, ou decidir pagar ele mesmo.