Departamento de Informática **Verificação Validação Software**
Faculdade de Ciências **Exame Dois**
Universidade de Lisboa **10 de Julho de 2009**

Inicie cada grupo numa nova página. Duração: três horas.

Grupo 1. [2.5 valores]

Os príncipios de engenharia de software dizem respeito a leis, regras ou doutrinas relacionadas com sistemas de software, incidindo no processo de construção e na descrição dos seus comportamentos. Glenford Myers apresentou, no seu livro 'The Art of Software Testing", um conjunto de príncipios sobre teste de software baseado em execução. Um destes príncipios é o seguinte.

> A actividade de teste deve ser levada a acabo por um grupo independente daquele responsável pelo desenvolvimento.

Apresente argumentos a favor ou contra a existência de um grupo de teste independente. No seu argumento considere os recursos, cultura e tipo de sistemas de software.

Grupo 2. [4 valores]

> O programa calcula o troco devido pelo pagamento de um dado produto, tal como seria preparado por um empregado de balcão ou por uma máquina de venda automática. O programa recebe três tipos de grandezas—o preço do produto, o valor total das moedas para pagamento, e um vector com as moedas disponíveis para preparar o troco—e prepara um vector com as moedas que compõem o troco. O programa gera um erro quando: não for possível preparar o troco com os valores introduzidos; o valor do pagamento for insuficiente para cobrir o preço do produto. O preço do produto e o pagamento são dados em cêntimos. Cada um dos vectores tem três posições, para moedas de 1, 2 e 5 cêntimos.

a) Derive um conjunto de classes de equivalência para o programa de trocos; complemente as classes utilizando a análise dos valores fronteira. Certifique-se que identifica classes válidas e inválidas.

b) Conceba uma bateria de casos de testes utilizando as classes de equivalência e os valores fronteira identificados. Para cada caso de teste especifique a classe de equivalência coberta, os valores de entrada, os possíveis valores na saída, e o identificador de caso de teste escolhido. Mostre em forma tabular que cobriu todas as classes e todos valores fronteira.

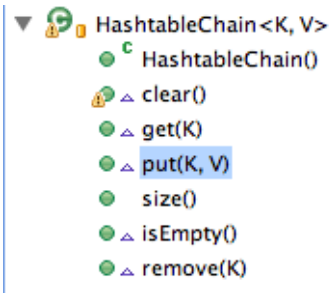
c) Identifique as condições de entrada e de saída do programa e as regras que as relacionam. Apresente por meio de uma tabela estas relações e desenhe o grafo de causa e efeito.

d) Desenvolva uma tabela de decisão e, a partir desta, uma bateria de casos de teste.

e) Relacione as duas baterias de testes obtidas.

Grupo 3. [3 valores]

Considere a seguinte classe que implementa uma tabela de dispersão com endereçamento encadeado,



e um dos seus métodos:

```
public void put(K key, V value) {
  int index = find(key);
  if (table[index] == null) {
    // Create a new linked list at table[index].
    table[index] = new LinkedList<Entry<K, V>>();
  }
  // Search the list at table[index] to find the key.
  for (Entry<K, V> nextItem : table[index]) {
  // If the search is successful, replace the old value.
    if (nextItem.key.equals(key)) {
      nextItem.value = value;
      return;
    }
  }
 // assert: key is not in the table, add new item.
  table[index].addFirst(new Entry<K, V>(key, value));
  numKeys++;
  if ((double)numKeys / table.length > LOAD_THRESHOLD)
    rehash();
}
```

a) Para o método put descreva uma bateria de testes que assegure 100% de cobertura de decisão. Apresente uma tabela que identifique cada ramo e os casos de teste que o utilizam.

b) Para o mesmo método identifique a informação de fluxo de dados para cada parâmetro, variável local ou atributo da classe, em cada expressão e instrução. Para cada uma destas variáveis apresente uma tabela identificando os caminhos definição-utilização. A partir das tabelas gere uma bateria de testes que exercitem tantos caminhos definição-utilização quantos os possíveis.

c) Relacione as duas baterias de testes obtidas.

Grupo 4. [2 valores]

Falamos de *análise estática* quando o artefacto de software é examinado manualmente, ou com um conjunto de ferramentas, mas não executado. Nos métodos de *análise dinâmica* o artefacto de software é executado utilizando um conjunto de *inputs* e o sei *output* é examinado e comparadao com o esperado.

a) Discuta o tipo de artefactos de sofware que pode ser alvo de análises dinâmicas e de análise estática.

b) Revisão é uma reunião de grupo cujo propósito é avaliar um artefacto de software ou conjunto destes. Quais os benefícios para uma organização resultantes do estabelecimento de um programa de revisão

Grupo 5. [1,5 valores]

A passagem para o nível 3 do *Test Maturity Model* é um passo importante para as organizações. Os quatro objectivos de maturidade do TMM3 são:

- Controlo e monitorização do processo de teste;
- Integração da actividade de teste no ciclo de vida do software;
- Estabelecimento dum programa de formação técnica; e
- Estabelecimento de uma organização de testes.

Descreva sumariamente os passos necessários para que uma organização atinja estes objectivos.