Técnicas de Conceção de Testes

- **Técnicas de conceção de Testes**
  - Técnicas baseadas nas especificações ou caixa-preta
  - Técnicas baseadas na estrutura ou caixa-branca
  - Técnicas baseadas na experiência

  - Escolher as técnicas de teste

Técnicas de Conceção de Testes

- **Técnicas de conceção de Testes**
  - O que é uma técnica de conceção de Teste?
    - **Sabendo:**
      - O teste exaustivo é impraticável
      - Devem usar-se subconjuntos dos casos de teste
      - Os casos de teste com maior probabilidade de encontrar defeitos
  - São necessários processos e técnicas que nos ajudem a selecionar os melhores casos de teste.

Técnicas de Conceção de Testes

- **Técnicas de conceção de Testes**
  - O que é uma técnica de conceção de Teste?
    - **O procedimento para selecionar e projetar os casos de teste**
      - Baseado no modelo estrutural ou funcional do software
      - Que obtenha sucesso na deteção de defeitos
    - **Uma maneira de determinar os melhores casos de teste**
    - **O processo de identificar as condições, casos e dados de teste.**

Técnicas de Conceção de Testes

- **Técnicas de conceção de Testes**
  - Vantagens de usar técnicas de conceção de Teste?
    - **Pessoas diferentes**
      - Probabilidades similares de encontrar defeitos
    - **Encontrar mais falhas**
      - Foco em tipos específicos de falhas
      - Testar os objetos (produtos) corretos
    - **Teste mais eficiente**
      - Encontrar falhas com menor esforço
      - Técnicas sistemáticas e mensuráveis.

Técnicas de Conceção de Testes

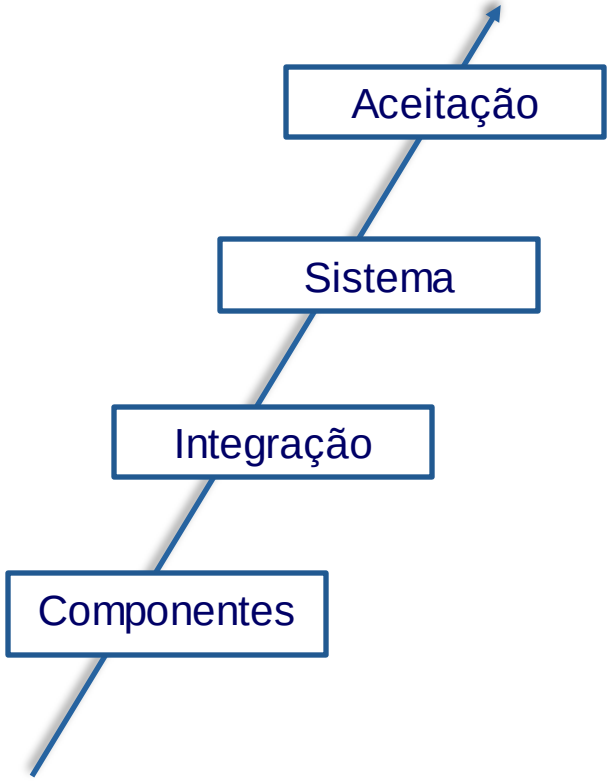
- **Técnicas de conceção de Testes**
  - 4 tipos de técnicas sistemáticas
    - **Estáticas (não-execução)**
      - Análise de documentação, análise do código fonte, etc. (já abordados)
    - **Funcionais (caixa-preta)**
      - Baseado no comportamento, nas funcionalidades do software
    - **Estruturais (caixa-branca)**
      - Baseados na estrutura do software
      - A informação sobre como o software é construído é utilizada para derivar os casos de teste
        - Por ex. o código e informações detalhadas de conceção
    - **Baseadas na experiência**
      - O conhecimento e experiência de pessoas é utilizado para derivar casos de teste.

Técnicas de Conceção de Testes

- **Técnicas de conceção de Testes**
  - Funcionais (caixa-preta)
  - Estruturais (caixa-branca)

Testes de caixa-preta
Usados em todos os níveis mas mais dominantes nos níveis mais altos.

Testes de caixa-branca
Usados predominantemente nos níveis mais baixos.



Técnicas de Conceção de Testes

- **Técnicas de conceção de Testes**
  - Técnicas Baseadas nas Especificações ou **Caixa-Preta**

Técnicas de Conceção de Testes

- **Técnicas de conceção de Testes**
  - Técnicas Baseadas nas Especificações ou **Caixa-Preta**
  - Objetivos de aprendizagem:
    - **Escrever casos de teste com base em modelos de software fornecidos utilizando:**
      - partição por equivalências
      - análise de valor fronteira
      - tabelas de decisão
      - diagramas/tabelas de transição de estados
    - **Explicar o objetivo principal de cada uma das quatro técnicas de teste, qual o nível e tipo de teste que pode utilizar cada técnica, e de que forma podemos medir a cobertura.**

Técnicas de Conceção de Testes

- **Técnicas de conceção de Testes**
  - Técnicas Baseadas nas Especificações ou **Caixa-Preta**
  - Objetivo
    - **Determinar se os requisitos foram total ou parcialmente satisfeitos pelo produto**
    - **Avaliam o comportamento externo do software.**
  - Como
    - **são fornecidas entradas e avaliadas as saídas geradas para verificar se estão em conformidade com o especificado**
    - **os detalhes de implementação não são considerados**
    - **o software é avaliado segundo o ponto de vista do utilizador**

Técnicas de Conceção de Testes

- **Baseadas nas Especificações ou Caixa-Preta**
  - Particionar por equivalências
  - Análise de Valor Fronteira
  - Testes baseados em Tabelas de Decisão
  - Testes de Transição de Estados
  - Testes de Casos de Uso.

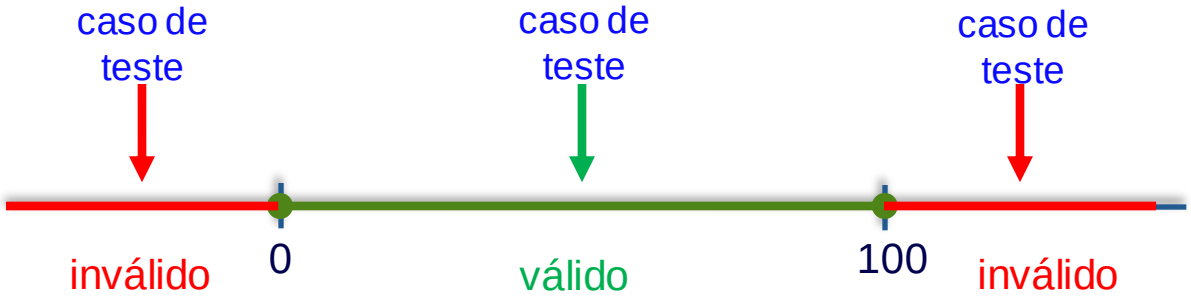
Técnicas de Conceção de Testes

- **Baseadas nas Especificações ou Caixa-Preta**
  - Particionar por equivalências (grupos ou classes)
    - **Adequada ao teste de valores típicos de entradas possíveis**
    - **Divisão das entradas em partições (grupos, classes)**
    - **Cada partição com características comuns**
    - **Um teste efetuado para um valor da partição é equivalente ao teste para qualquer outro valor nessa partição**
    - **Devem ser desenhados testes para cobrir todas as partições.**

Técnicas de Conceção de Testes

- **Baseadas nas Especificações ou Caixa-Preta**
  - Particionar por equivalências (grupos ou classes)

Uma função que apenas deve aceitar valores no intervalo [0, 100]



- **Pressupostos:**
  - Se funciona para um caso de teste então funciona para todos os restantes da mesma partição
  - É melhor testar um de cada partição do que testar apenas todos de uma mesma partição.

Técnicas de Conceção de Testes

- **Baseadas nas Especificações ou Caixa-Preta**
  - Particionar por equivalências (grupos ou classes)
    - **Como fazer**
      1) Definir as partições (classes de equivalência)
         1) Identificar as condições de entrada (i.e.. Intervalos de entrada válidos e inválidos)
         2) Identificar as partições propriamente ditas (são definidas a partir das condições de entrada)
         3) Confirmar se todos os elementos de uma partição podem ser tratados da mesma forma
            1) Se não for possível, subdividir essa partição
      2) Identificar os casos de teste
         1) Numerar as partições válidas e inválidas
         2) Para cada partição inválida
            1) Pelo menor um caso de teste que cubra essa situação
         3) Para cada partição válida
            1) Pelo menor um caso de teste que cubra essa situação. Escolher casos de teste que possam cobrir várias partições.

Técnicas de Conceção de Testes

- **Baseadas nas Especificações ou Caixa-Preta**
  - Particionar por equivalências (grupos ou classes)
    - **Exemplo**
      - Suponha-se que se pretende testar um programa que é responsável por verificar se a sintaxe de um comando está correta. A sintaxe a verificar é:
        - OPER OP1 OP2
      - Requisitos/Restrições:
        - <OPER>
          - provém de uma lista de 5 operações válidas
            Lista = {SOM, SUB, DIV, MUL, EXP}
          - É sempre representado por 3 caracteres maiúsculos
        - <OP1> e <OP2>
          - Número inteiro não negativo

Técnicas de Conceção de Testes

- **Baseadas nas Especificações ou Caixa-Preta**
  - Particionar por equivalências (grupos ou classes)
    - **Exemplo**
      1) Definir as partições: identificar entradas

Condições de entrada	Tipo de condição	Part válidas	P. inválidas

Técnicas de Conceção de Testes

- **Baseadas nas Especificações ou Caixa-Preta**
  - Particionar por equivalências (grupos ou classes)
    - **Exemplo**
      1) Definir as partições: **partições podem ser tratados da mesma forma**

Condições de entrada	Tipo de condição	Part válidas	P. inválidas
OPER maiúsculas	Booleano	sim	não
OPER dimensão	Booleano	sim	não
OPER Válido (da lista)	Booleano	sim	não
OPER = SUB	Da lista	sim	não
OPER = … (lista)	Da lista	sim	não
…	…	…	…
OP1	int	≥ 0	< 0
OP2	int	≥ 0	< 0
Após OPER vêm OP1	Booleano	sim	Não
…	…	…	…

Técnicas de Conceção de Testes

- **Baseadas nas Especificações ou Caixa-Preta**
  - Particionar por equivalências (grupos ou classes)
    - **Exemplo**

      2) Identificar os casos de teste: numerar as partições válidas e inv.

Condições de entrada	Tipo de condição	Part válidas	P. inválidas
OPER maiúsculas	Booleano	sim	não
OPER dimensão	Booleano	sim	não
OPER Válido (da lista)	Booleano	sim	não
OPER = SUB	Da lista	sim	não
OPER = … (lista)	Da lista	sim	não
…	…	…	…
OP1	int	≥ 0	< 0
OP2	int	≥ 0	< 0
Após OPER vêm OP1	Booleano	sim	Não
…	…	…	…

Técnicas de Conceção de Testes

- **Baseadas nas Especificações ou Caixa-Preta**
  - Particionar por equivalências (grupos ou classes)
    - **Exemplo**

2) Identificar os casos de teste: numerar as partições válidas e inv.

Condições de entrada	Tipo de condição	Part válidas	P. inválidas
OPER maiúsculas	Booleano	Sim (1)	não (2)
OPER dimensão	Booleano	sim (3)	não (4)
OPER Válido (da lista)	Booleano	sim (5)	não (6)
OPER = SUB	Da lista	sim (7)	não (8)
OPER = … (lista)	Da lista	sim (9)	não (10)
…	…	…	…
OP1	int	≥ 0 (11)	< 0 (12)
OP2	int	≥ 0 (13)	< 0 (14)
Após OPER vêm OP1	Booleano	sim (15)	Não (16)
…	…	…	…

Técnicas de Conceção de Testes

- **Baseadas nas Especificações ou Caixa-Preta**
  - Particionar por equivalências (grupos ou classes)
    - **Exemplo**

2) Identificar os casos de teste

- Para as partições inválidas: um caso de teste para cada

- Sub 20 10 //Grupo inv. 2
- soma 10 20 //Grupo inv. 2, 4
- MLT 2 3 //Grupo inv. 6
- SUB -2 -4 //Grupo inv. 12, 14
- SUB 2 -5 //Grupo inv. 14
- SUB -5 6 //Grupo inv. 12
- //semelhante para as restantes OPER válidas
- SUB MUL 12 //Grupo inv. 16
- …

Técnicas de Conceção de Testes

- **Baseadas nas Especificações ou Caixa-Preta**
  - Particionar por equivalências (grupos ou classes)
    - **Exemplo**

2) Identificar os casos de teste

- Para as partições válidas: cobrir as partições com número menor casos de teste

- SOM 10 15 //Grupo 1, 3, 5, 9, 11, 13, 15
- …
- //semelhante para as restantes OPER válidas
- …
- //sendo possível aumentar e diversificar os casos de teste

Técnicas de Conceção de Testes

- **Baseadas nas Especificações ou Caixa-Preta**
  - Particionar por equivalências (grupos ou classes)
    - **EXERCÍCIO**
      - Uma função que aceita as medidas dos 3 lados de uma figura e retorna o tipo de triângulo.
      - Da especificação da função consta o seguinte:
        - A saída da função é um inteiro que representa o tipo de triângulo:
          - 1 - Equilátero
          - 2 - Isósceles
          - 3 – Escaleno
          - 4 – Não é um triângulo
        - As medidas de cada lado são valores decimais]0,1].

        int triangulo(double lado1, double lado2, double lado3)

      - Desenhe os casos de teste usando a técnica de partição por equivalência.

Técnicas de Conceção de Testes

- **Baseadas nas Especificações ou Caixa-Preta**
  - EXERCÍCIO
    - **Desenhe os casos de teste usando a técnica de partição por equivalência para o seguinte sistema.**
    - **Descrição:**
      - A liquidação do IRC em 2016 é referente aos lucros de 2015.
      - O valor a pagar é calculado com base na tabela seguinte, considerando o tipo de entidade e a região onde está instalada.

Entidades	Continente	Madeira	Açores
Entidades residentes e estabelecimentos estáveis de entidades não residentes [1]	21%	21%	16,8%
Entidades residentes que exerçam, a título principal, atividade comercial, industrial ou agrícola, considerados como PME (nos primeiros 15.000€) [2]	17%	17%	13.6%
Entidades residentes que exerçam, a título principal, atividade comercial, industrial ou agrícola, considerados como PME (valor excedente) [3]	21%	21%	16,8%

Técnicas de Conceção de Testes

- **Desenhe os casos de teste usando a técnica de partição por equivalência para o seguinte sistema.**
  - Para determinar este valor foi desenvolvido um software, que se encontra representado no formulário seguinte em que:
    - **O campo lucro apenas deve aceitar valores inteiros não negativos**
    - **O campo residência apenas deve aceitar uma letra maiúscula do conjunto {A, C, M}**
    - **Caso o formulário seja corretamente preenchido o sistema apresenta o valor de IRC a pagar**
    - **Caso sejam preenchidos valores não válidos o sistema deve apresentar uma janela com indicação de erro no preenchimento**

Lucro referente ao ano de 2015
(apenas valores não negativos)

Residência (A=Açores, C=Continente, M=Madeira)

Tipo de entidade {1, 2, 3}

Calcular

Técnicas de Conceção de Testes

O código fornecido corresponde a um programa para realizar operações matemáticas e estatísticas. Foi desenvolvido com base nas seguintes especificações:

- O programa deve disponibilizar 3 operações matemáticas:
  - *EPrimo(int)* verifica se um número é primo. Retorna um valor booleano que confirma se o número é primo ou não. Apenas deverão ser aceites números inteiros superiores a 1. Todos os que não satisfizerem esta condição devem ser tratados como falsos primos.
  - *mMC(int n1, int n2)* deve retornar o mínimo múltiplo comum entre dois números. Apenas deverá aceitar números inteiros positivos.
  - *MDC(int n1, int n2)* deve retornar o máximo divisor comum entre dois números. Apenas deverá aceitar números inteiros positivos.
- As funções que retornem números inteiros devem ter em consideração o seguinte:
  - Quando os seus parâmetros não estiverem de acordo com as especificações deve retornar -1.

Técnicas de Conceção de Testes

- **EXERCÍCIO (Para fazerem)**
  - Escrever os casos de teste

Um programa, para realizar as funções de somatório e fatorial, foi desenvolvido com base nas seguintes especificações:

- *int calcula (int n, char op)* - deve retornar da operação fatorial ou somatório do número ***n*** consoante ***op*** = ***f*** ou ***op*** = ***s***.
- ***n*** apenas deverá aceitar números iguais ou superiores a zero.
- ***op*** pode tomar os valores ***f*** ou ***s***.
- ***Se op*** = ***f*** a função retorna o valor do fatorial de ***n*** considerando:
  - O fatorial de ´0´ é 1
  - O fatorial de n (se n ≥ 1) é dado por: 1×2×...×n
- ***Se op*** = ***s*** a função retorna o valor do somatório de ***n*** considerando:
  - O somatório de ***n*** é dado por: *1 + 2 + ... + n*.
- Para as entradas inválidas o programa deve retornar o valor -1

Considerando as especificações, escreva o conjunto de casos de teste que realizaria usando as técnicas de particionar por equivalentes e análise de valor fronteira (apresente todos os passos realizados).