UNIVERSIDADE FEDERAL DO PARANÁ - UFPR
Bacharelado em Ciência da Computação

SOFT
DISCIPLINA: Engenharia de Software **DATA: _____/_____/_____**
AULA NÚMERO: 15 **PROFESSOR: Andrey**

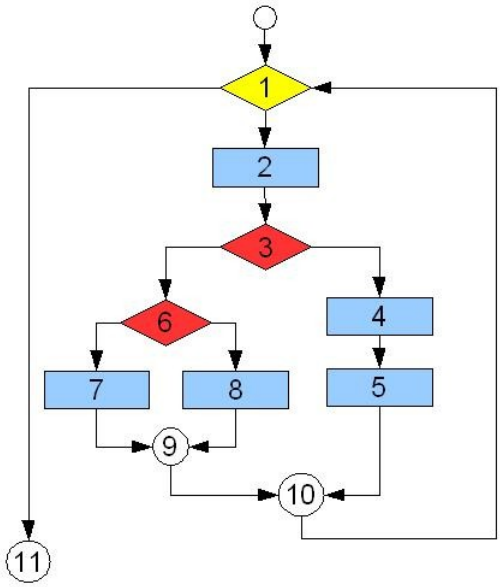
APRESENTAÇÃO

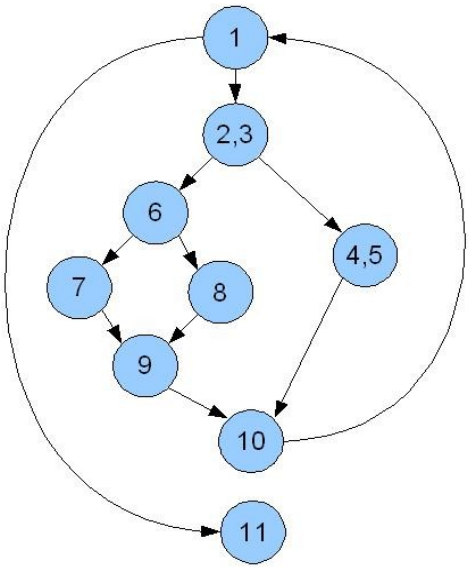
O objetivo desta aula é apresentar e discutir conceitos relacionados a Testes de Software.

DESENVOLVIMENTO

Notação Grafo de Fluxo

- O grafo de fluxo mostra o fluxo de controle
- Nós representam um ou mais processos
- Arestas representam o fluxo de controle
- Regiões do grafo são áreas limitadas pelas arestas e nós (incluindo a área fora do grafo)





Complexidade Ciclomática

Como saber quantos caminho procurar?
Complexidade Ciclomática
O número de regiões do grafo de fluxo corresponde à complexidade ciclomática
V(G) = E – N + 2
E : número de ramos do grafo
N : numero de nós do grafo
V(G) = P + 1
P : número de nós predicados do grafo
Nó predicado é o que tem duas ou mais arestas saindo dele

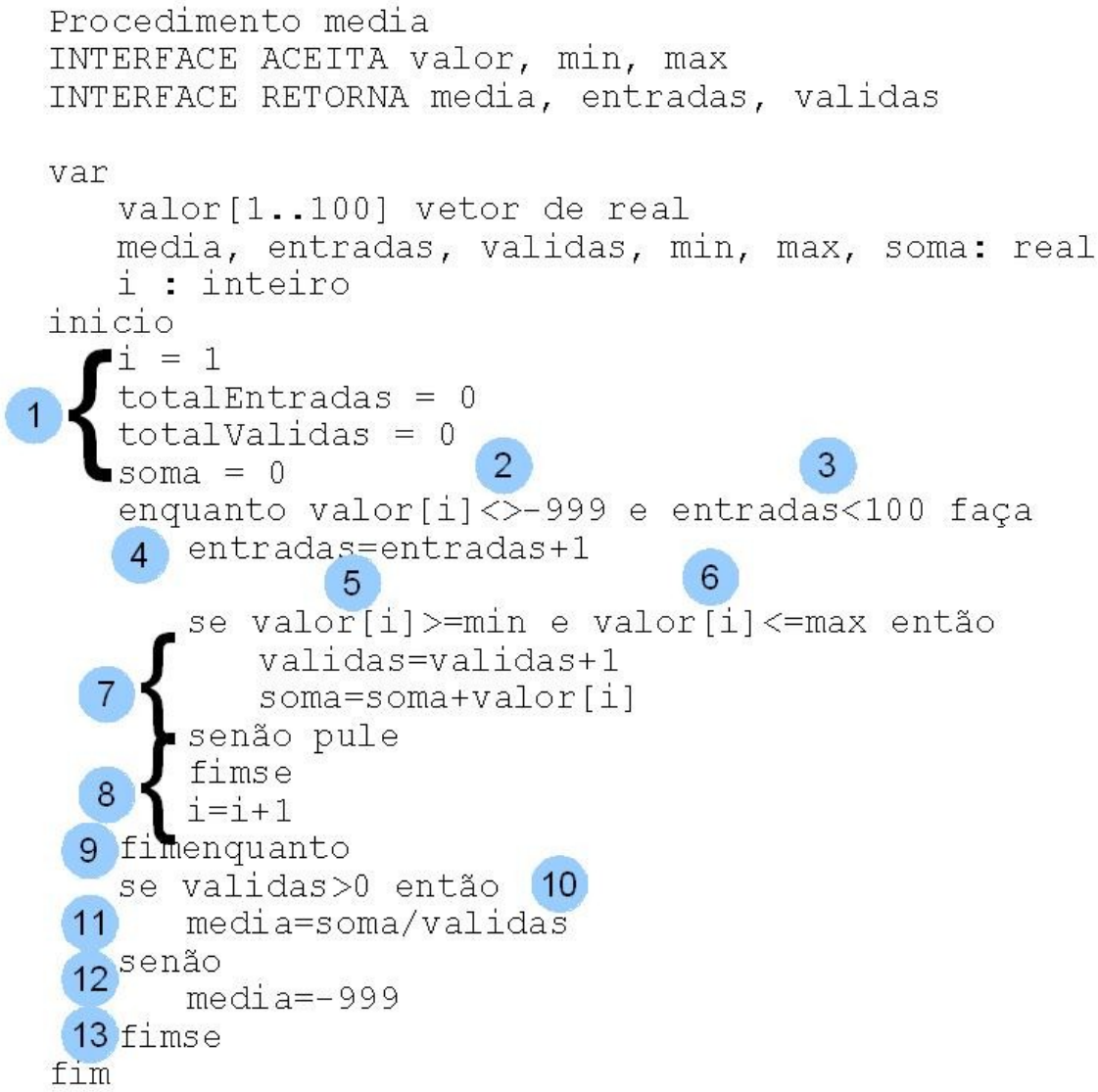
Derivação de Casos de Teste

Usando o projeto ou código como base, desenhe o grafo de fluxo correspondente
Determine a complexidade ciclomática do grafo de fluxo correspondente

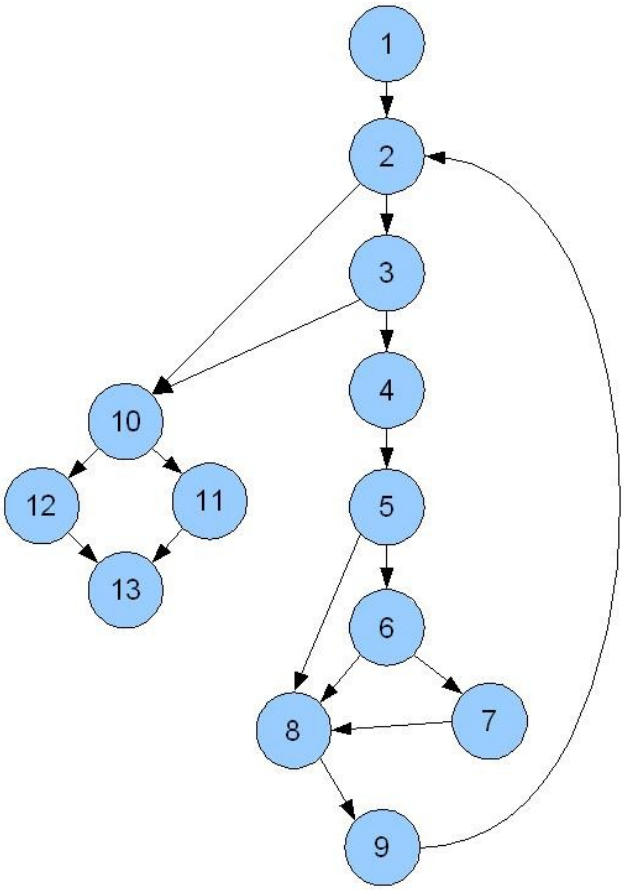
Determine um conjunto base de caminhos linearmente independentes
Prepare os casos de teste que vão forçar a execução de cada caminho do conjunto.

Exemplo

Derivar os casos de teste para um programa que calcula a média das entradas válidas, usando o método do caminho básico.



Passo 1: Desenhe o grafo de fluxo correspondente.



Passo 2: Calcule a complexidade ciclomática.
V(G) = 6 regiões
V(G) = 17 arestas – 13 nós + 2 = 6
V(G) = 5 nós predicados + 1 = 6

Passo 3: Determine um conjunto base de caminhos independentes.
Caminho 1: 1-2-10-11-13
Caminho 2: 1-2-10-12-13
Caminho 3: 1-2-3-10-11-13
Caminho 4: 1-2-3-4-5-8-9-2...
Caminho 5: 1-2-3-4-5-6-8-9-2...
Caminho 6: 1-2-3-4-5-6-7-8-9-2...

Passo 4: Prepare os casos de teste que vão forçar a execução de cada caminho
O caminho 1 só pode ser testado como parte dos caminhos 4, 5 e 6
Caminho 2: valor (i) = -999; resultados esperados: média = -999 e os outros valores com os valores iniciais.
Caminho 6: valor (i) = entrada válida; resultados esperados: média correta baseada em n valores e totais apropriados.

Teste de Estrutura de Controle

O teste do caminho básico é simples e eficaz, mas nem sempre é suficiente.
Outras variações:
Teste de condição
Teste de fluxo de dados
Teste de ciclo

Teste de Condição

Método de projeto de teste que exercita as condições booleanas de um módulo de programa
Condição Simples:
E1 <operador relacional>E2
Condição Composta:
Operadores Booleanos E, OU e NÃO
O método de teste de condição focaliza o teste de cada condição para garantir que não contém erros

Teste de Fluxo de Dados

Seleciona caminhos de teste de acordo com as definições e dos usos das variáveis do programa (potenciais usos)
DEF(S) = {X | comando S contém definição de X}
USO(S) = {X | comando S contém uso de X}
Cadeia DU (definição-uso)
A definição de X no comando S é viva no comando S' se existir um caminho entre S e S' sem outra definição de X
Cadeia DU de X: [X, S, S']
Cada cadeia DU deve ser coberta pelo menos uma vez

Teste de Ciclos

Focaliza a validade das construções dos ciclos
Ciclos: simples, concatenados, aninhados e desestruturados.
Testes para Ciclos simples:
Pule o ciclo completamente
Apenas uma passagem pelo ciclo
Duas passagens
m passagens pelo ciclo, onde m < n
n – 1, n, n+1 passagens

Testes para Ciclos aninhados:
Comece no ciclo mais interno, outros ciclos nos valores mínimos
Teste o ciclo mais interno com os outros ciclos nos valore mínimos, incluindo valores fora do intervalo e excluídos
Trabalhe em direção ao exterior passando para o ciclo seguinte com os ciclos externos em valores mínimos e os internos em valores típicos
Continue até que todos os ciclos tenham sido testados

Ferramentas Automatizadas

PokeTool (UNICAMP)

PROTEUM e PROTEUM/IM (USP)

ATIVIDADE

1. Construa os seguintes algoritmos e projete os casos de teste usando a técnica do caminho mínimo e o teste de ciclo.
   Um algoritmo que lê um número e imprime a lista dos seus divisores
   Um algoritmo que lê dois números e calcula o máximo divisor comum pelo método de Euclides.
   Um algoritmo que lê as 4 notas de um aluno e diga se ele passou por média, está em final ou reprovou.

BIBLIOGRAFIA BÁSICA

PRESSMAN, R. S.. Engenharia de Software. 6a. ed. McGraw-Hill. 2006

CARVALHO, A. M. B. R.; CHIOSSI, T. C. S.. Introdução à engenharia de software. Editora da Unicamp. 2001.

OLIVEIRA, M. Técnicas de Testes Caixa-Preta. Apresentação no II Encontro Brasileiro de Teste de Software, outubro 2007.