Aula 2

True or False

a) (***V***) Os testes devem estar presentes durante todo o desenvolvimento do projeto.
b) (***F***) Apesar de diferentes fases, os testes são iguais em abstração e complexidade.
c) (***V***) Teste de Unidade é concentrado nas menores unidades do software.
d) (***V***) Erros de lógica e de implementação são falhas identificáveis em um teste.
e) (***F***) Funcionando todas as partes de um código, garantimos que vai funcionar no todo
f) (***V***) A combinação dos módulos é o teste de integração, logo após o teste de unidade
g) (***F***) Um teste de integração verifica os requisitos de um sistema já pronto.
h) (***V***) O teste de sistema avalia a comunicação do sistema com outros.
i) (***V***) Um exemplo de teste de sistema é identificar o retorno do banco de dados.
j) (***F***) Os teste de aceitação são, em geral, a primeira etapa de um plano de testes.
k) (***V***) O objetivo do teste de aceitação é ver se o teste está pronto e pode ser usado.
l) (***V***) Os quatro níveis de teste podem ser automatizados e existem ferramentas para tal.

Associate the columns

Uma variável não pode ter seu conteúdo modificado após ter sido inicializado;	(0)	(***1***)	Escopo de variável.
A variável é criada e deixa de existir após o interpretador sair do bloco de execução;	(1)	(***0***)	Modificador de acesso *final*
A variável sequer chega a ser utilizada no flluxo do programa.	(2)	(***2***)	Erro de compilação.

For discussion

a) Na linguagem Java existe mais de uma estrutura de controle de fluxo (*if...else*, por exemplo). Por que aprender todas se apenas uma resolveria o problema?
Resposta pessoal. Apesar de bastante padronizada, a linguagem Java permite uma estilização de código em muitos momentos e esta diferença entre uma solução e outra tem a ver com a cultura da empresa na qual trabalha e muitas vezes com o nível de conhecimento do desenvolvedor, entre outros fatores. Essa fato se repete também nas outras linguagens. Vejamos a língua portuguesa. Podemos usar as palavras "rosto", "face" e até "cara" e optamos por uma ou outra de acordo com a cultura na qual estamos inseridos, por exemplo. Assim, usar if...else ou switch... case ou for ou while, etc, como controle de fluxo se justifica muito pela formação, costume, cultura do desenvolvedor e da sua organização.
b) O trecho de código: *if (idade<18) { entrada=false} else {entrada=true}* poderia ser reestrito como *verificaldade(idade)*. Como essa mudança facilitaria um teste de software?
Com o método verificaldade( ), tendo idade como parâmetro, podemos ter um valor booleano de retorno (false ou true) tal qual o "idade<18" pode retornar falso ou verdadeiro, porém no caso do método há claramente uma unidade separada, que

pode resultar num teste e ser guardado em um pacote para reaproveitamento em outros programas. Essa alteração é bastante positiva para aprimoramento do processo de produção.

c) Em se tratando de teste de software, pense em algumas vantagens possíveis do fato da linguagem Java ser fortemente tipada.

De fato, ao declarar uma nota como int, já sabemos que o java retornará um erro se o usuário tentar inserir uma string. Esse erro é, mesmo que ainda muito básico, um aspecto importante para garantir o cumprimento das regras do negócio.

d) Qual a motivação para o uso de uma linguagem orientada a objetos?

Se considerarmos que a orientação a objetos significa uma representação de código mais próxima do mundo real, com a criação (instanciação) e manipulação de objetos, já se pode perceber o ganho de produtividade que pode representar. É inicialmente necessário uma mudança de visão de mundo (paradigma) computacional, ou seja, pensar de uma maneira orientada a objetos e construir as soluções a partir desta perspectiva. Os objetos facilitam o reaproveitamento de código para outros programas.