ソフトウェア開発論

単体試験局面

第2.16版

- ホワイトボックス試験

ソフトウェア開発論・スケジュール

- はじめに
- 要求分析
- 外部設計
- 内部設計
- プログラミング
- **単体試験局面**
- 結合試験
- 総合試験
- 信頼性評価
- まとめ

3つの試験局面

- 基本的な考え方：3段階の試験プロセス
  - 単体試験局面
  - 結合試験局面
  - 総合試験局面

各試験（単体、結合、総合）の仕方：共通

1. entry criteria: 作成・修正したコードの文法エラーがないこと。
2. 各試験ごとに、試験用入力データ(以下**テストデータ**)+予想結果を含む、**テストケース（複数）**を前もって用意する。
3. テストデータを流す。
4. 予想した結果と一致するかどうか確認する。
5. 問題が有れば、設計問題記録もしくはプログラム問題記録として報告
6. 設計、コードを修正し、再度2からくり返す。
7. 最後に全テストケースを流し直す。
   これを**レグレッション・テスト（回帰試験）**と呼ぶ。
8. 回帰試験で出た問題への対策が完了すれば、再度回帰試験を行う。
9. exit criteria: 問題が出てこなくなれば、各試験終了である。

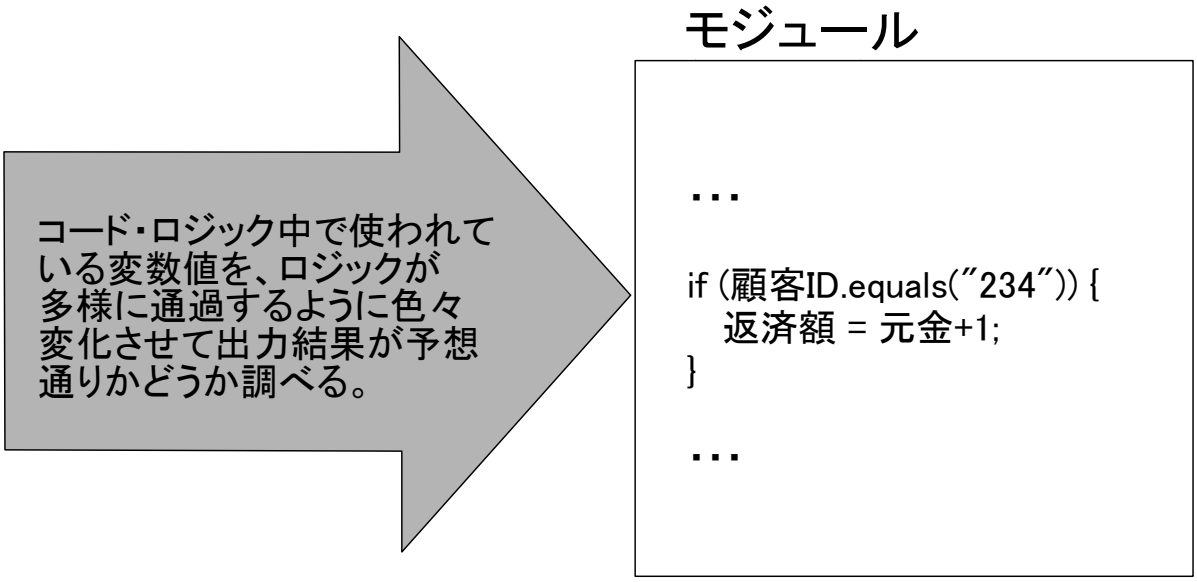
単体試験局面

- WHAT:各モジュール（関数／サブルーチン／メソッドなど）のコードが、
  その内部設計と等価であるかどうかという観点で試験する。
- HOW: ホワイトボックス試験
- WHO:プログラマーが実施する場合と、
  専門のテスターが実施する場合がある。

単体試験局面で使う
ホワイトボックス試験

- 品質保証の前提となる、均質かつ高効率なテストケースを作成する必要がある。
- 構造化プログラミングされている必要がある。

一般的なホワイトボックス試験



均質かつ効率的なホワイトボックス試験

- 基本的な考え方：コード・ロジックから
  テストデータ(複数)を、具体的な数値として作成する。
  結果は**内部設計より予想**しておく。
- 均質かつ効率的な、テストデータ作成：
  - 制御の流れに注目した作成法
    - 条件文(C)
    - 繰り返し文(R)
  - 境界値分析法
    - 配列(A)
    - ファイルの読み込み(F)

制御の流れに注目した作成法
その１－条件文(C)試験

- C0: 条件で値の変わる分岐**のみ**をテストする。

文書	コード	内部設計書

試験対象	if (顧客ID.equals(″234″)) {
	　返済額 = 元金+1;
	}
		もし、顧客ID==234なら、返済額は元金に1を足す。

C0テストデータ	顧客ID=234;元金=100;返済額=0	
予想結果		返済額==101
試験実施	予想結果と等しく、かつ2行目を通過すればOK	

- C1: 全分岐を網羅する試験を行う。
  - テストケース：　顧客ID==234の時　テストデータ　顧客ID=234;元金=100;返済額=0 予想結果　返済額==101。
    顧客ID<>234の時　テストデータ　顧客ID=233;元金=100;返済額=0 予想結果　返済額==0。
  - 試験実施：　顧客ID==234の時　予想結果と等しく、かつ2行目を通過すればOK
    顧客ID<>234の時　予想結果と等しく、かつ4行目を通過すればOK
- 注意：C0を選ぶかC1を選ぶかは、試験網羅率の選択になる。
  後の講義で、品質評価を行うので、どちらを選択したか記録しておくこと

制御の流れに注目した作成法
その２－繰り返し文(R)試験

注:　()内は内部設計書

- 試験対象コード（日歩利子計算）
  - long 返済額=元金; double r=本日の金利/36500;
  - for (int i=1; i<=返済予定日数; i++){
  - 　返済額 = (long)(返済額*(1+r));
  - }
- R0: 繰り返し文を一度は実行する試験を行う。
  - テストデータ：　元金=100;本日の金利=365;返済予定日数=1;
  - 予想結果：　返済額==101でかつ3行目を通過すればOK
- R1: ①繰り返し文を0回実行する試験、②一度は実行する試験、③最後まで実行する試験の3ケースを行う。

番号	テストデータ	内部設計書からの予想結果
①	元金=100;本日の金利=365;返済予定日数=0	返済額==100 5行目を通過する。
②	元金=100;本日の金利=365;返済予定日数=1	返済額==101 3行目を通過する。
③	仮定：最長3日間とする。（ここでは予想がめんどうなため、実際には外部設計書では3650日間） 元金=100;本日の金利=365;返済予定日数=3	返済額==103 3行目を通過する。

境界値分析法
その１－配列(A)試験

- 内部設計書:　配列を使った日歩利子計算
- 試験対象コード
  - long 返済額[ ]={元金,0,0,0,0};
  - double r=本日の金利/36500;
  - for (int i=1; i<=返済予定日数; i++){
  - 　返済額[i] = (long)(返済額[i-1]*(1+r));
  - }
- A0:配列境界値を試験する。
- A1:配列境界値(index)と隣の値を試験する。

A0	テストデータ	内部設計書からの予想結果
①	返済予定日数=1	返済額=={100,101,0,0,0}; 4行目を通る。
②	返済予定日数=4	返済額=={100,101,102,103,104}; 4行目を通る。

A1	テストデータ	予想結果
①	返済予定日数=1	返済額=={100,101,0,0,0}; 4行目を通る。
②	返済予定日数=4	返済額=={100,101,102,103,104}; 4行目を通る。
③	返済予定日数=0	返済額=={100,0,0,0,0}; 6行目を通る。
④	返済予定日数=2	返済額=={100,101,102,0,0}; 4行目を通る。
⑤	返済予定日数=3	返済額=={100,101,102,103,0}; 4行目を通る。
⑥	返済予定日数=5	invalid index

境界値分析法
その２－ファイルの読み込み(F)試験
注:　()内は内部設計書

- 試験対象コード(ファイル"ABC"を読み込み"ABD"に書き込む)

```
FileReader FI = new FileReader("ABC.txt");
BufferedReader fi = new BufferedReader(FI);
FileWriter FO = new FileWriter("ABD.txt");
String s;
while ((s= fi.readLine()) !=null ) {
    FO.write(s+"¥r¥n");
}
FI.close();
FO.close();
```

- F0: 1レコード正常に書き込まれたファイル"ABC"を用意し、実行する。また、内部設計された上限nレコード分の正常に書き込まれたファイル"ABC"を用意し、実行する。
- F1: 次の"ABC"ファイルをテストデータとして準備し、実行する。
  - "ABC"ファイルなし
  - 空(0レコード)
  - 1レコード
  - 2レコード
  - n-1レコード
  - nレコード
  - n+1レコード

F0	テストデータ	内部設計書から予想結果
①	"ABC"1レコード	"ABD"の内容が"ABC"と同じになる。
②	"ABC"nレコード	"ABD"の内容が"ABC"と同じになる。

F1	テストデータ	予想結果
①	"ABC"1レコード	"ABD"の内容が"ABC"と同じになる。
②	"ABC"nレコード	"ABD"の内容が"ABC"と同じになる。
③	"ABC"なし	空の"ABD"が出来る。
④	"ABC"0レコード	空の"ABD"が出来る。
⑤	"ABC"2レコード	"ABD"の内容が"ABC"と同じになる。
⑥	"ABC"n-1レコード	"ABD"の内容が"ABC"と同じになる。
⑦	"ABC"n+1レコード	運任せ。

なぜ構造化プログラミングか

- 入り口一個、出口一個ならば、その中の正当性を一度証明すれば、その中はブラックボックスとして良い。（正当性定理）
- したがって、関数単位にコード・インスペクションや単体試験を一度完璧にしておけば、
- その関数を利用するときには、再度関数の中身の検査は必要なくなる。
- そうすると、関数再利用のうまみが出てくる。

演習：単体試験実行・条件文

- 利子元金返済計算ソフト・条件文の単体試験用テストケースを作成してみよう。
- ステップ1：コードより、テストデータを作成する。

```
private static 元金取得結果 元金取得メソッド
(String 顧客ID){
   元金取得結果 out=new 元金取得結果();
  if (顧客ID.equals("234")) {
    out.お客様のお名前="Satoh Kazuo";
    out.元金=1000000;
  }
  return out;
}
```

- ステップ2：各テストケースごとに、内部設計書HIPOより、予想結果を作成する。

  - 内部設計書（HIPOより）
  - 顧客IDを手がかりに顧客DBより、お客さまのお名前、元金を取得
  - IF 顧客IDが顧客DBに未登録ならば、　エラーコード=1を返す。

- ステップ3：コード実行結果と、予想結果を見比べる。

演習：単体試験実行・繰り返し文

- 返済額計算メソッド・繰り返し文の
  単体試験用テストケースを作成してみよう。

1. コードより、テストデータを作成する。

```
元金取得結果 in = 元金取得メソッド(顧客ID);
 long 返済額=in.元金; double 本日の日歩=本日の金利/36500;
 for (int i=1; i<=返済予定日数; i++){
   //日歩利子計算。ただし、1円以下は切り上げ。
   返済額 =  (long)(返済額*(1+本日の日歩) +1);
 }
```

2. 内部設計書HIPOより、予想結果を作成する。

- //本日の金利、経過日数、元金より、返済額を計算
- //ただし、1円未満は切り上げ
- E=(1+B/36500)**C

データ制約の種類	ドメイン制約	導出制約
C=返済予定日数	{ 0..3650 }、整数	なし
E=返済額	{ 0..8桁 }、正の整数	E=(1+B/36500)**C 1円未満は切り上げ

3. コードを実行し、予想結果と見比べる。

おわり