ソフトウェア設計法及び演習
ソフトウェア工学概論及び演習

大山 勝徳
日本大学 工学部

連絡（再掲）

- 休講
  - 対象: 1組，休講日: 7/13（月）
  - 対象: 2組，休講日: 7/6（月）
- 補講
  - 対象: 1組，補講日: 7/27（月），1,2限，113教室
  - 対象: 2組，補講日: 7/27（月），1,2限，122教室

- 設計演習2のレビュー
- 構造化プログラミング
  - 詳細設計
    - モジュールの外部設計
    - モジュールの論理設計
  - 構造化定理
  - Java言語による構造化プログラミング
  - デシジョンテーブル

- 設計演習2のレビュー
- 構造化プログラミング
  - 詳細設計
    - モジュールの外部設計
    - モジュールの論理設計
  - 構造化定理
  - Java言語による構造化プログラミング
  - デシジョンテーブル

レビュー

- レビュー
  - 仕様書や設計書，プログラムなどを，開発者とは別の人が内容を検討し，結果をフィードバックする工程
  - 検討項目
    - 仕様や要求を満たしているか
    - 誤りや不具合の有無
    - 冗長性の有無

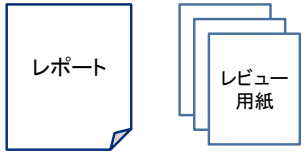
思い込みによる検討漏れを防ぐなど，「開発者とは別の人」が実施することが重要

レビュー方法（設計演習1と同じ手順）

1. レポート提出者（設計者）は，レビュー用紙を3枚受け取る
2. レビュー1
   1. 設計者は，共同作業者以外の人にレビューを依頼し，レポートとレビュー用紙1枚を渡す
   2. レビューアはレビューを行ない，評価を記載したレビュー用紙とレポートを設計者に返す
3. レビュー2，レビュー3を行なう
4. 設計者は，レポートとレビュー3枚をまとめ，提出する

レビューと配点

- レポート・レビューの配点に関して
  レポートの点＋α↓1 ±α↓2
  - レポートの点: レポートそのものの点数
    - 評価全体の20%
  - α↓1 ：レポートの加点
  - α↓2 ：レビューの加点/減点
    - レビューの点はレビューアに付く



- 設計演習2のレビュー
- 構造化プログラミング
  - 詳細設計
    - モジュールの外部設計
    - モジュールの論理設計
  - 構造化定理
  - Java言語による構造化プログラミング
  - デシジョンテーブル

詳細設計（モジュールの外部設計）

- 詳細設計
  - 構造化設計（モジュール分割）の後の工程
  - 各モジュールをプログラム化する作業
- モジュールの外部設計
  - 呼び出すモジュールのモジュール名やパラメータなど（外部特性）の定義
    - モジュール名
    - 機能
    - パラメータリスト
    - 入力変数／出力変数
    - 外部効果

詳細設計（モジュールの外部設計）

モジュール外部特性		
特性	説明	プログラム特性
モジュール名	暗証番号確認	checkpw
機能	入力された暗証番号を確認し，エラーであれば，メッセージを出力し，再入力を促す	
インタフェース（パラメータリスト）	入力：暗証番号 出力：誤り回数	Checkpw(pwno, wrongno)
入出力変数	暗証番号：英数字，8桁 誤り回数：整数	Pwno char (8), Wrongno int
外部効果	入力：暗証番号 出力：エラーメッセージ	

図 モジュールの外部特性

p.191

詳細設計（モジュールの論理設計）

- モジュールの論理（アルゴリズム）設計
  - プログラミング作業のこと
  （1）アルゴリズムの作成
  （2）データの定義
  - 理解しやすいアルゴリズムでモジュールのコードを表現するために**構造化プログラミング**

構造化の目的：
・プログラムの構造を明確化する
（誤りの発見を容易にする）
・プログラムの変更を容易にする

- 設計演習2のレビュー
- 構造化プログラミング
  - 詳細設計
    - モジュールの外部設計
    - モジュールの論理設計
  - 構造化定理
  - Java言語による構造化プログラミング
  - デシジョンテーブル

構造化プログラミングの提唱

- Dijkstraの主張（GOTO論争，1968年）
  - プログラムをわかりにくくしているのは不用意に使うGOTO命令である
  - プログラムは基本的な3つの構造単位（順次，選択，繰返し）にそって書けば，GOTO命令なしに書くことができる
  - GOTOがなくなれば，プログラムは上から下へ自然に読んで行けるため，わかりやすくなる

構造化プログラミング

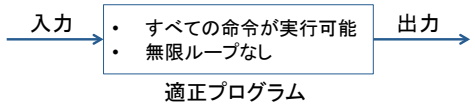
- 構造化プログラミングの原理
  - 上から下へ自然に読めるプログラムコード
    - 今日ではgoto文を使用しない
  - 構造化定理
    - 適正プログラム
    - 制御構造の標準化
  - 段階的詳細化
    - 一度に詳細化せず，段階的な詳細化を行なう

構造化定理

- 構造化定理の要点（1/2）

p.192

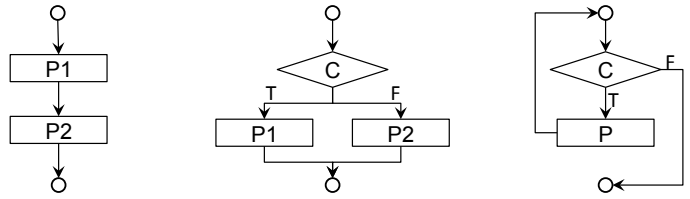
  - 適正プログラム
    - 1つの入り口と1つの出口のみのプログラム制御
    - すべての命令が実行可能（到達不能な行を作らない）
    - 無限ループなし



  - 制御構造の標準化
    - 上から下へ自然に読めるプログラムコードの実現

制御構造の標準化

- 構造化定理の要点（2/2）
  - 基本制御構造（順次，選択，繰返し）



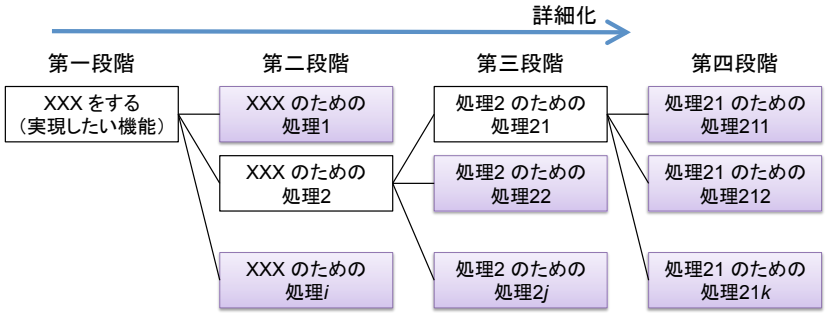

```
P1;
P2;
P3;
```

```
if (C) {
  P1;
} else {
  P2;
}
```

```
while (C) {
  P;
}
```

段階的詳細化

- プログラムの設計時に，処理手順の記述を概要から徐々に詳細化する
  - 最も詳細化された記述はプログラムコードに相当



段階的詳細化

- 良い詳細化の条件
  - 各命令ごとに独立して詳細化できる
  - 各段階で，詳細を見ずに内容を理解できる
  - 各段階で，詳細さのレベルが統一されている
  - 不要な詳細まで表現されていない

処理の流れを「段階的」にバランス良く「詳細化」する

- 設計演習2のレビュー
- 構造化プログラミング
  - 詳細設計
    - モジュールの外部設計
    - モジュールの論理設計
  - 構造化定理
  - Java言語による構造化プログラミング
  - デシジョンテーブル

Java言語による構造化プログラミング

- クラスの外部設計と論理設計 p.203
  - クラスの外部設計後，各メソッドについて構造化プログラミングを実施できる
  - 特に，Javaは構造化プログラミングとオブジェクト指向プログラミングの両方に適している

（1）選択構造
- if文，switch文（多分岐選択）

（2）繰返し構造
- for文，while文，do-while文
- 繰返しの中断: break, continue

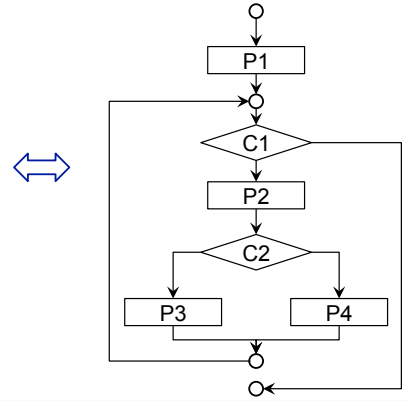
Java言語による構造化プログラミング

- 基本制御構造の組み合わせ
  - どのようなアルゴリズムでも記述できる

```
戻り値の型 メソッド名 (パラメータ) {

 P1;
 while ( C1 ) {
   P2;
   if ( C2 ) {
     P3;
   } else {
     P4;
   }
 }

}
```



繰り返し構造の中にある選択構造

- 設計演習2のレビュー
- 構造化プログラミング
  - 詳細設計
    - モジュールの外部設計
    - モジュールの論理設計
  - 構造化定理
  - Java言語による構造化プログラミング
  - デシジョンテーブル

デシジョンテーブルとは

- プログラムで判定する必要のある条件とそのときの選択処理の内容（行動）の関係表
  - 条件表題欄と条件記入欄
    - 判定すべき条件の組み合わせを表す
  - 行動表題欄と行動記入欄
    - 条件記入欄の条件に従って実行すべき行動を表す

条件表題欄	条件記入欄
行動表題欄	行動記入欄

デシジョンテーブルの構成

デシジョンテーブルの具体例

p.211

(1) システムの立ち上げ時である.	Y	N	N	N	N	N
(2) システムの終了時である.	N	Y	N	N	N	N
(3) 要求が預け入れである.	N	N	Y	N	N	N
(4) 要求が払戻しである.	N	N	N	Y	N	N
(5) 要求が残高照会である.	N	N	N	N	Y	N
(6) 要求が記帳である.	N	N	N	N	N	N
a. 初期処理を行う.	X	-	-	-	-	-
b. 要求を読取る.	-	-	X	X	X	X
c. 預け入れ処理を行う.	-	-	X	-	-	-
d. 払戻し処理を行う.	-	-	-	X	-	-
e. 残高照会処理を行う.	-	-	-	-	X	-
f. 記帳処理を行う.	-	-	-	-	-	X
g. 終了処理を行う.	-	X	-	-	-	-

(3)から(6)に共通したbの行動

ATMシステムのデシジョンテーブル

まとめ

- 設計演習2のレビュー
- 構造化プログラミング
  - 詳細設計
    - モジュールの外部設計
    - モジュールの論理設計
  - 構造化定理
  - Java言語による構造化プログラミング
  - デシジョンテーブル