高信頼システム08

ソフトウェアの高信頼化02

テストに関して

7月22日（月）13：00から

筆記用具，授業資料，電卓，自筆ノートのみ使用可.

ブラックボックステスト

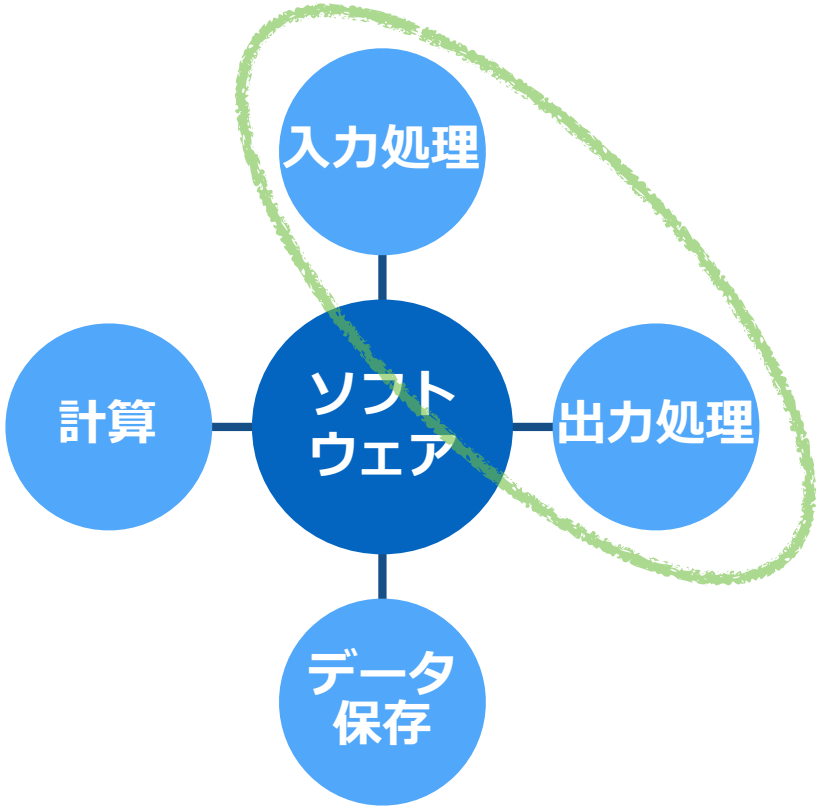
プログラムのソースコードを利用せずに（見ずに）テストを行う.

大規模ソフトウェアのテストに有効.
ほとんどのソフトウェアがこの手法でテストされている

- **同値分割法（同値クラステスト）**
- **境界分析法（境界値テスト）**
- **ディシジョンテーブル**
- **状態遷移テスト**

ソフトウェアの振る舞い

同値・境界値テストの対象部分



例：プリントダイアログ


プリンタ：
HP OfficeJet Pro 6970
プリセット：
デフォルト設定
部数：
1
ページ：
すべて
高信頼システム08
ソフトウェアの高信頼化02
0を入れると．．．
“部数”フィールドの値が範囲外です。
有効な値にリセット中...
OK
スライド
グリッド
配布資料
アウトライン
1/5
スライドレイアウト：
ページ余白を使用
発表者ノートを含める
スライド番号を含める
名前と日付を含める
オプション：
スライドの背景をプリント
ビルドの各段階をプリント
スキップしたスライドを含める
低品質イメージをプリント
PDF
詳細を隠す
キャンセル
プリント

同値分割法

入力領域を「同値クラス」（部分集合）に分割，
部分集合に入る入力値を等価（同値）とみなし，
テストケースを削減する方法

例

＜要求仕様＞

入力 A ：1から999まで入力可能

入力 B ：1から499まで入力可能

テストの際にどのような入力でテストを行うか？

テストケースを作ってみる：有効同値と無効同値

入力範囲を同値に分割する

入力A：

無効同値（～0）	**有効同値（1～999）**	無効同値（1000～）

入力B：

無効同値（～0）	**有効同値（1～499）**	無効同値（499～）

非常に強いテストケース

有効同値と無効同値をしっかりカバーできるように

① A=0　B=0

0はチェックした方が良い

② A=-20　B=-20

③ A=0　B=250

④ A=-5 B=750

⑤ A=500 B=250

⑥ A=500 B=0

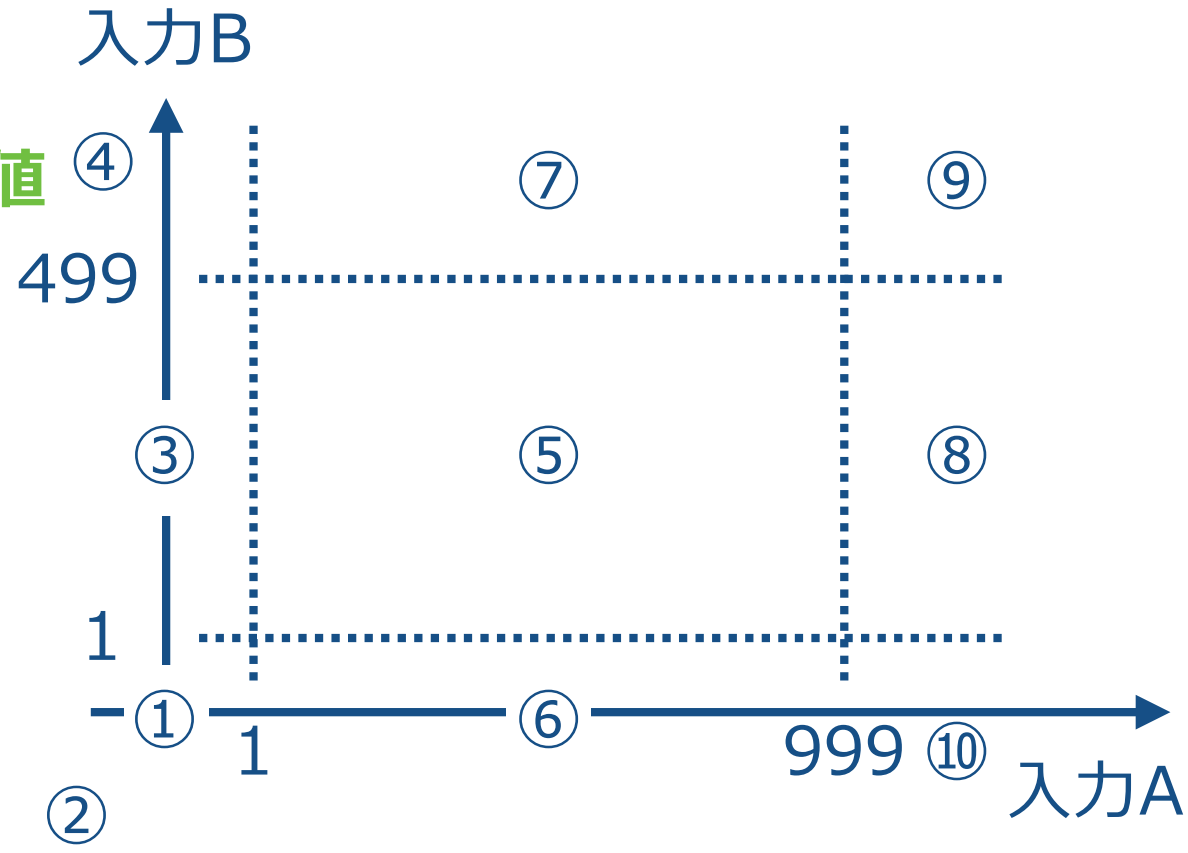
⑦ A=500　B=550

⑧ A=1100 B=250

⑨ A=1100 B=550

有効同値

A: 1～999, B: 1～499
であれば良い



実践的なテストケース

テストケースを減らすには

できるだけ少ない同値で全ての範囲を網羅できるように

① A=0　B=0

② A=-20　B=-20

③ A=0　B=250

④ A=-5 B=750

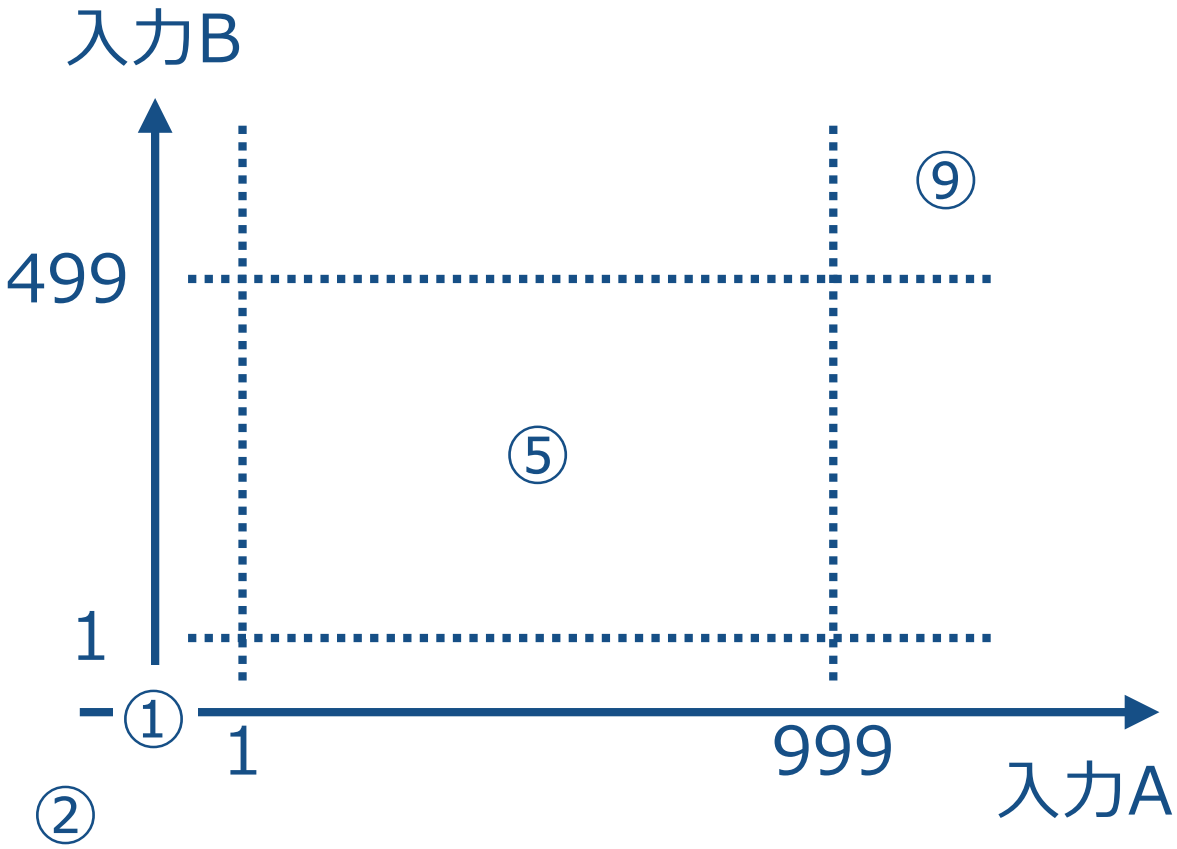
⑤ A=500 B=250

⑥ A=500 B=0

⑦ A=500　B=550

⑧ A=1100 B=250

⑨ A=1100 B=550



実践的なテストケース2

① A=0　B=0

② A=-20　B=-20

③ A=0　B=250

④ A=-5 B=750

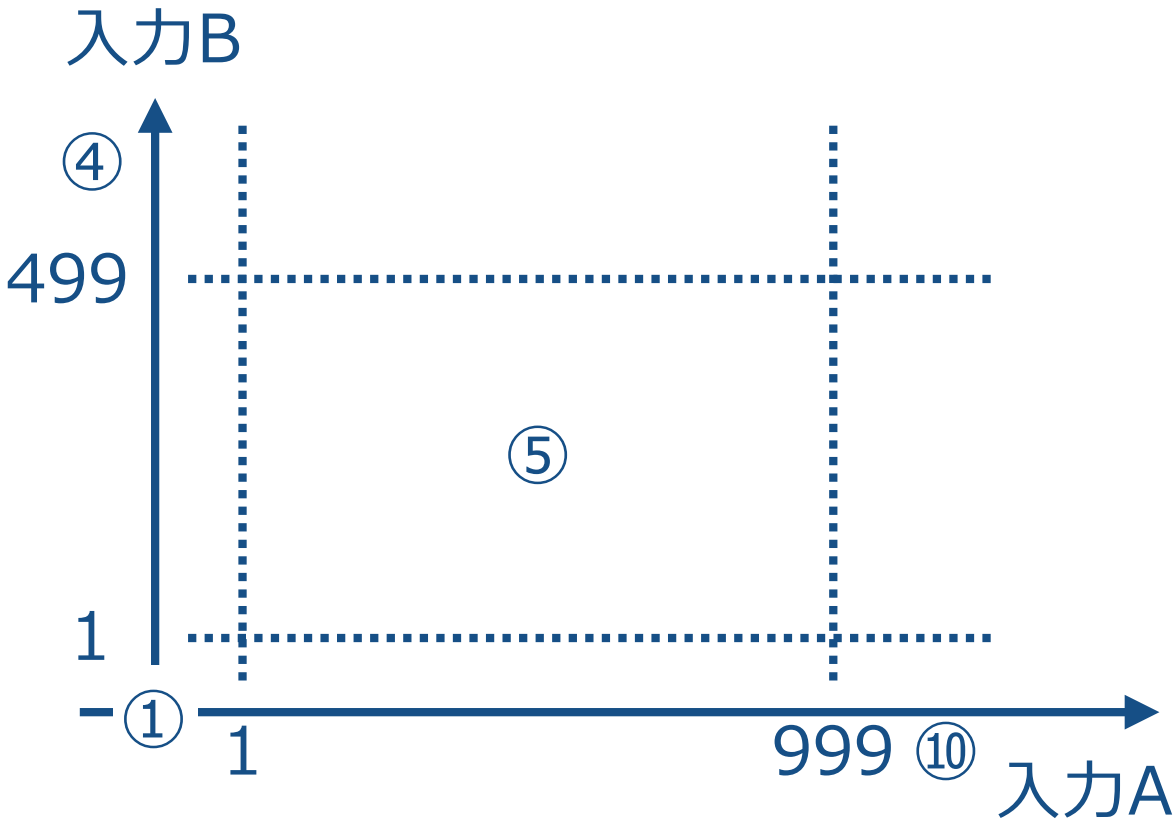
⑤ A=500 B=250

⑥ A=500 B=0

⑦ A=500　B=550

⑧ A=1100 B=250

⑨ A=1100 B=550

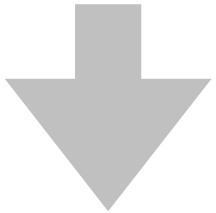


同値クラステストの課題

有効同値が1個に対して，無効同値9個もある

実際のプログラムは複雑

→ 無効同値のテストケースを作って実行するのに時間がかかる.



実際の現場では，ある程度，無効同値のテストを省略する必要がある

でも，プログラムがフリーズしたり，データが破壊されるのは避けなくてはいけない

実はプログラムに対する仮定があります

想定されているプログラム

```
if (0 < a && a <= 999 && 0 < b && b <= 499)
    //正しい値が入力された時の処理
else
    //間違った値が入力された時の処理
```

想定されていないプログラム (同値が別の命令文で処理されている)

```
if (0 < a && a <= 100 && 0 < b && b <= 499)
    //正しい値が入力された時の処理
    （ここに間違いがあったら有効同値では検出できない）
else if (100 < a && a <= 999 && 0 < b && b <= 499)
    //正しい値が入力された時の処理
else
    //間違った値が入力された時の処理
```

境界値テスト

使用条件の境界となる値と，その隣の値に対してテストを行う方法

同値分割法とセットで使われる.

バグは「境界」に潜んでいる可能性が高い

無効同値と有効同値の境目

- 境界を表す条件の誤解
- コーディング時の条件の記述誤り

境界を表す条件の誤解

境界を表す仕様の記述方法は多様な表現がある：

「以上」，「以下」，「未満」，「より小さい」，
「～を超える」，「～を下回る」，「～まで」，
「～と同じになった場合」などなど.

例

料金を処理するプログラムの仕様例

- 7歳まで無料
- 7歳以上，20歳未満は子供料金
- 21歳以上は大人料金

曖昧な点はないですか？

仕様の曖昧な点を確認することが重要！！

改善の例

- 7歳無料
- 7歳以上，20歳は子供料金
- 21歳以上は大人料金

コーディング時の条件の記述誤り

- 0歳以上，7歳 **未満は** 無料
- 7歳以上，20歳 **以下** は子供料金
- 21歳以上は大人料金

```
if 0<= 年齢 and 年齢<7 then
    料金=無料
else if 7<=年齢 and 年齢<=20 then
    料金＝子供料金
else if 21<=年齢 then
    料金＝大人料金
```

エラーの4タイプ

◉タイプ1：等号や不等号の間違い

```
if 0<=年齢 and 年齢<=7 then
    料金=無料
else if 7<=年齢 and 年齢<=20 then
    ・・・・・・
```

◉タイプ2：数字の書き間違いなど

```
if 0<= 年齢 and 年齢<=8 then
    料金=無料
else if 7<=年齢 and 年齢<=20 then
    ・・・・・・
```

エラーの4タイプ

◉タイプ3：境界がない

```
if 0<= 年齢 and 年齢<=7 then
    料金=無料
// else if 7<=年齢 and 年齢<=20 then
    . . . . . .
```

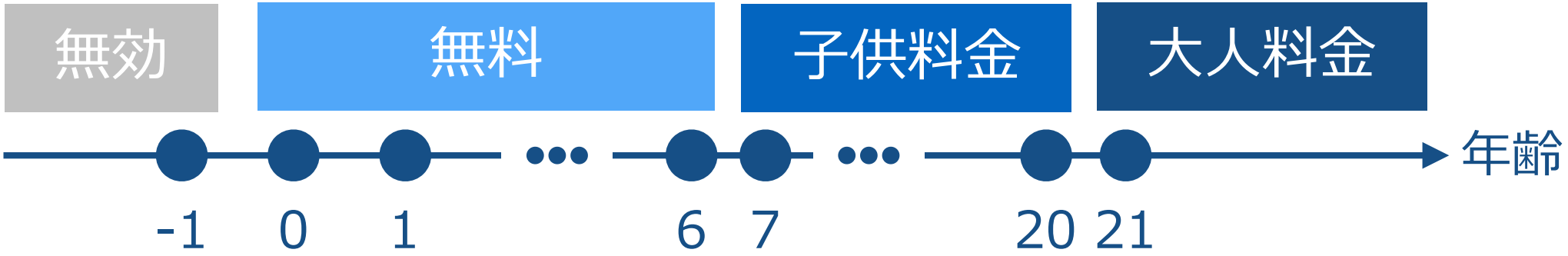
この行以降がコメントアウト

◉タイプ4：余分な境界

```
if 0<= 年齢 and 年齢<7 then
    料金=無料
else if 7<=年齢 and 年齢<=20 then
    料金＝子供料金
else if 21<=年齢 and 年齢 <= 65 then
    料金＝大人料金
```

境界をテストするには：On-Offポイント法※

異なる同値領域に属するデータのうち，
距離が最小となるデータを用いてテストする



良いテストケース： 年齢=-1, 0, 6, 7, 20, 21

このテストケースで，前述の四つエラータイプを検出可能か？

※ Beizer, 「ソフトウェアテスト技法」

同値クラステストのみの場合

年齢=-5, 3, 15, 30（他の値もあり得ます）

このテストケースで，前述の四つエラータイプを検出可能か？

練習問題

あるインターネットショッピングサイトで，商品を期間限定販売できる機能を追加した．次の仕様の時に境界値テストでテストする値を求めなさい．

仕様

- 販売開始：4月1日　　0時00分00秒
- 販売終了：4月5日　23時59分59秒

ディシジョンテーブル

全ての入力の組み合わせ，出力・動作を表にする

＜要求仕様＞
入力 A ：1から999まで入力可能
入力 B ：1から499まで入力可能
出力：A × B

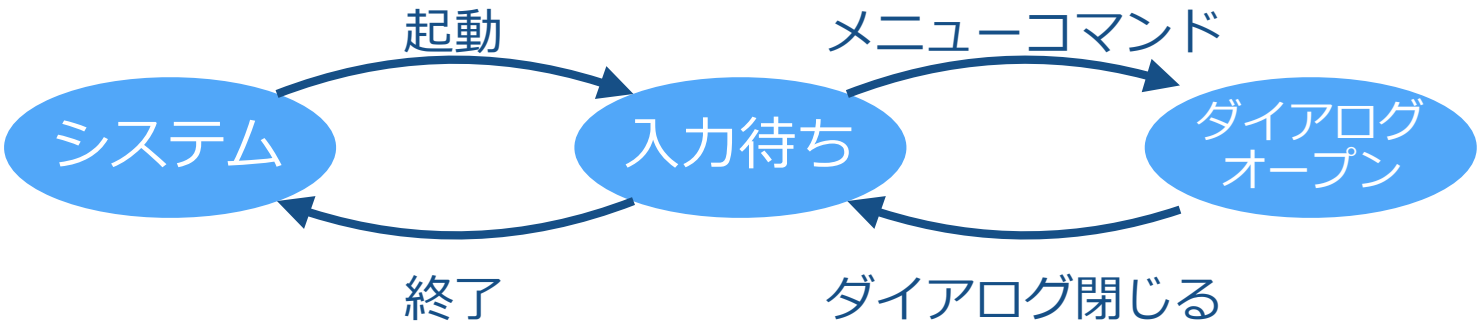
状態	ルール			
	ルール	ルール2	ルール3	ルール4
C1: A=正しい	True	True	False	False
C2: B=正しい	True	False	True	False
動作				
A1: 計算値出力	行う			
A2:入力エラー処理		行う	行う	行う

入力数が多くなると，表のサイズが膨大となる
→ 入力数が少なくて，動作が複雑なプログラムに適する

状態遷移テスト

アプリケーションは状態により挙動が異なる

例：ワープロ・エディタの状態遷移



状態 イベント	システム	入力待ち	ダイアログオープン
起動	A	NA（二つ目のインスタンスが起動しないことを確認）	NA（二つ目のインスタンスが起動しないことを確認）
メニューコマンド	NA	ファイルダイアログオープン	NA
入力	NA	入力待ち	NA
ダイアログ閉じる	NA	NA	入力待ち
アプリケーション終了	NA	システム	NA(終了しないことを確認)

A=Available, NA=Not Available