VDM++仕様に対する境界値分析を用いたテストケース自動生成

宮崎大学 工学研究科
片山 徹郎 研究室
立山博基 片山徹郎

目次

1.背景
2.テストケース自動生成ツールBWDM
3.現在の修正箇所
4.今後の課題

JaSST'16 Kyusyu 2016 11/4(Fri)

ソフトウェアの高品質化のために

- 従来のソフトウェア開発の仕様記述段階において、自然言語(Ex. 日本語、英語)の持つ曖昧さにより、仕様書に曖昧な箇所が含まれてしまう。
- 厳密な仕様を作成するには？
  - 形式手法(VDM++, SPIN)
- ソフトウェアテストも必須
  - ソフトウェアの品質のために言わずもがな大事
  - テストケース設計・テストの実施に手間と時間が掛かり、納期とのトレードオフになりがち

テストケース自動生成ツールBWDM

```
class Mix

functions

混在仕様 : nat * int -> seq of char
  混在仕様(arg1, arg2) ==
    if(arg1 mod 2 = 0) then
      if(0 <= arg2) then
        "arg1:偶数、arg2:正の数"
      else
        "arg1:偶数、arg2:負の数"
    else
      if(arg2 < 0) then
        "arg1:奇数、arg2:負の数"
      else
        "arg1:奇数、arg2:正の数";
```

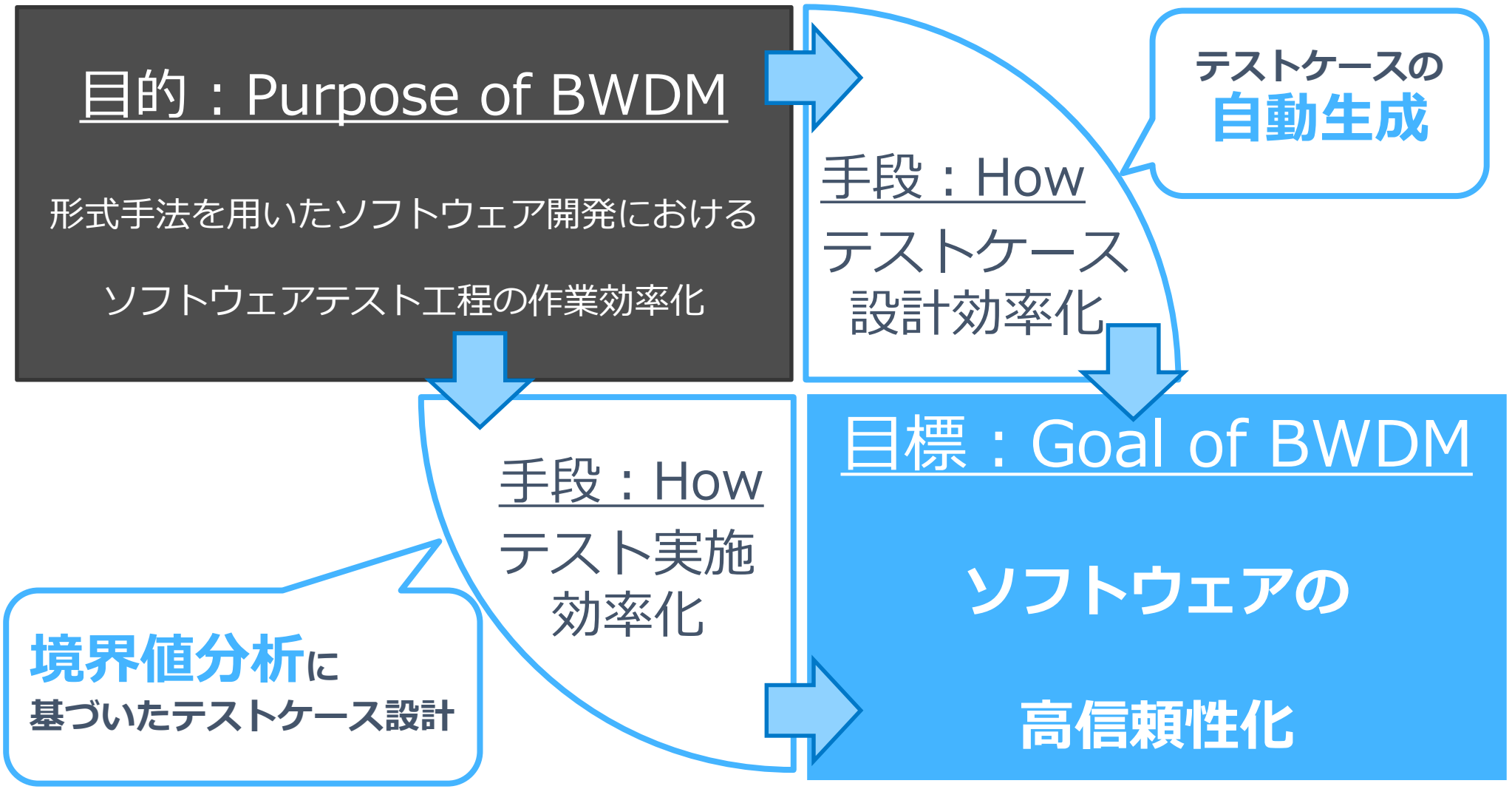
テストケースNo.	入力データ		-->	期待出力データ
No.1	natMin-1	intMin-1	-->	Undefined Action
No.2	natMin-1	intMin	-->	Undefined Action
No.3	natMin-1	intMax	-->	Undefined Action
No.4	natMin-1	intMax+1	-->	Undefined Action
No.5	natMin-1	0	-->	Undefined Action
No.6	natMin-1	-1	-->	Undefined Action
No.7	natMin	intMin-1	-->	Undefined Action
No.8	natMin	intMin	-->	arg1:偶数、arg2:負の数
No.9	natMin	intMax	-->	arg1:偶数、arg2:正の数
No.10	natMin	intMax+1	-->	Undefined Action
No.11	natMin	0	-->	arg1:偶数、arg2:正の数
No.12	natMin	-1	-->	arg1:偶数、arg2:負の数
No.13	natMax	intMin-1	-->	Undefined Action
No.14	natMax	intMin	-->	arg1:奇数、arg2:負の数
No.15	natMax	intMax	-->	arg1:奇数、arg2:正の数
No.16	natMax	intMax+1	-->	Undefined Action
No.17	natMax	0	-->	arg1:奇数、arg2:正の数
No.18	natMax	-1	-->	arg1:奇数、arg2:負の数
No.19	natMax+1	intMin-1	-->	Undefined Action

VDM++仕様

自動生成

テストケース

BWDM: Boundary Value & Vienna Development Method

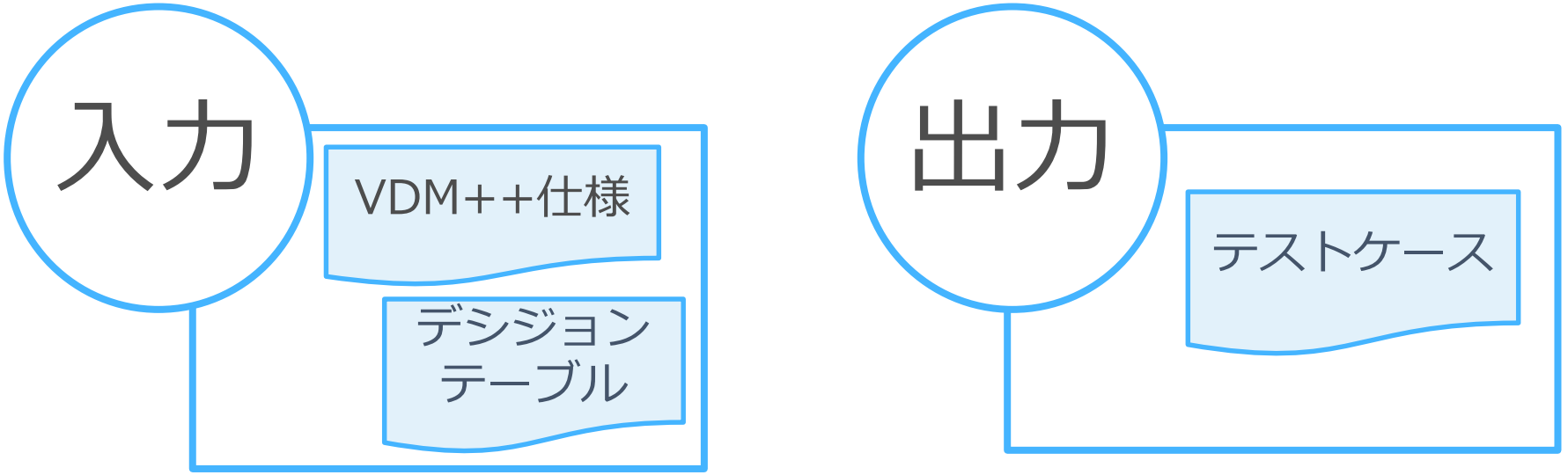

目的：Purpose of BWDM
形式手法を用いたソフトウェア開発における
ソフトウェアテスト工程の作業効率化
手段：How
テストケース
設計効率化
テストケースの
自動生成
手段：How
テスト実施
効率化
境界値分析に
基づいたテストケース設計
目標：Goal of BWDM
ソフトウェアの
高信頼性化

BWDMの使用場面

1. 仕様をVDM++で記述
2. 設計などを経てコーディング
3. BWDMで仕様からテストケース生成
4. 開発したソフトウェアをテスト

コーディングまで終えた段階で、作成した仕様からのコーディングが正しく行えているかをテストする

BWDMの入出力



VDM++仕様・・・通常のテキストファイル形式
デシジョンテーブル、テストケース・・・csvファイル形式

デシジョンテーブルは本研究室で以前開発した
デシジョンテーブル自動生成ツールで生成したものを使用する

デシジョンテーブル

クラス名：Sample						
関数名：SampleFunction						
		#1	#2	#3	#4	
Condition	a < 5	T	T	F	F	1
Condition	12 <= a	T	F	T	F	
Action	"aは5未満です"	T	T	F	F	
Action	"aは12以上です"	F	F	T	F	2
Action	"aは5以上かつ12未満です"	F	F	F	T	

3

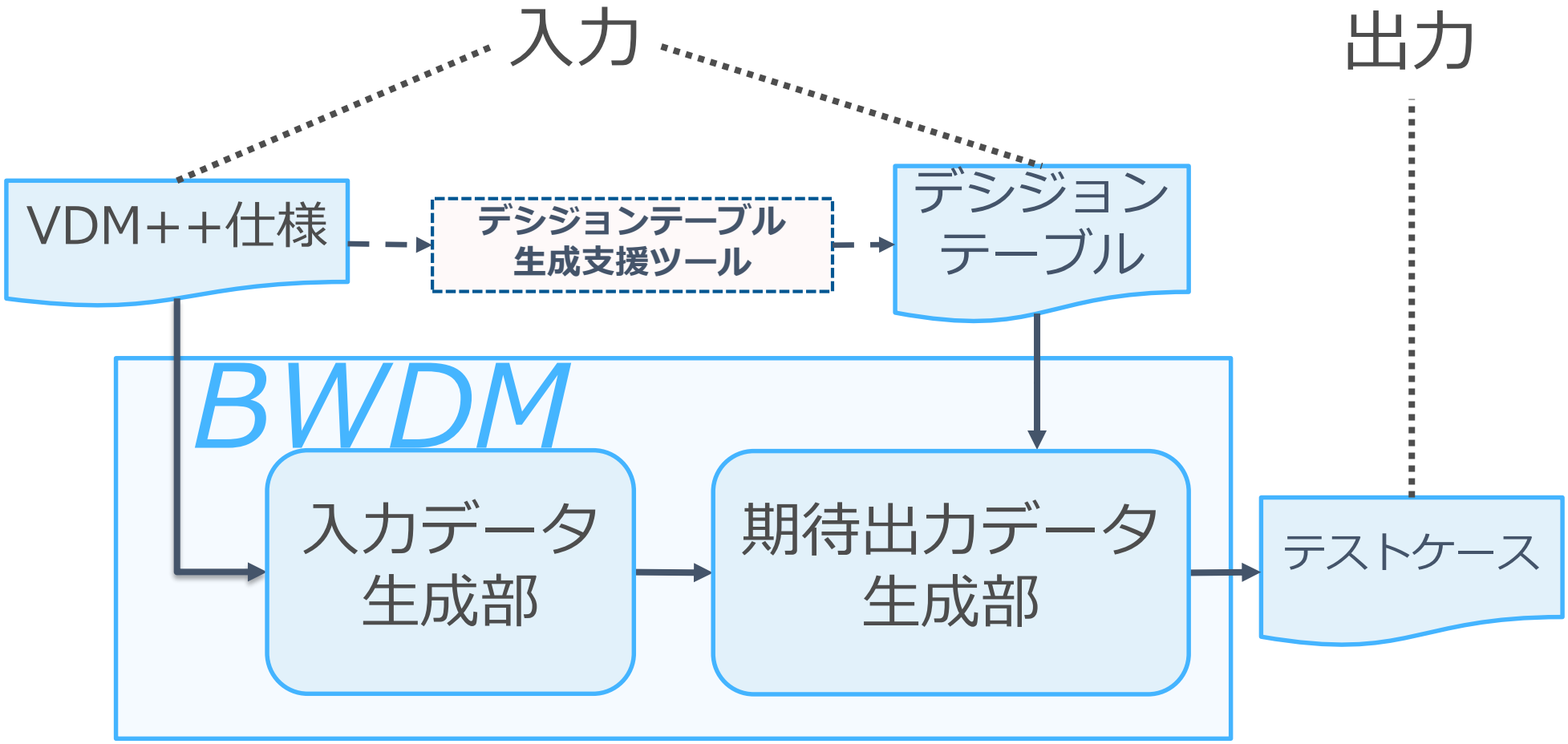
ソフトウェアの入力に対する出力を表形式にまとめたもの

1. 条件部 - ソフトウェア内の条件文とそれらの真偽値が取りうる組み合わせを記述
2. 動作部 - ソフトウェア内の動作と条件部の真偽値に対する動作を記述
3. 規則部 - 条件部と動作部により、ソフトウェアの振る舞いを表す

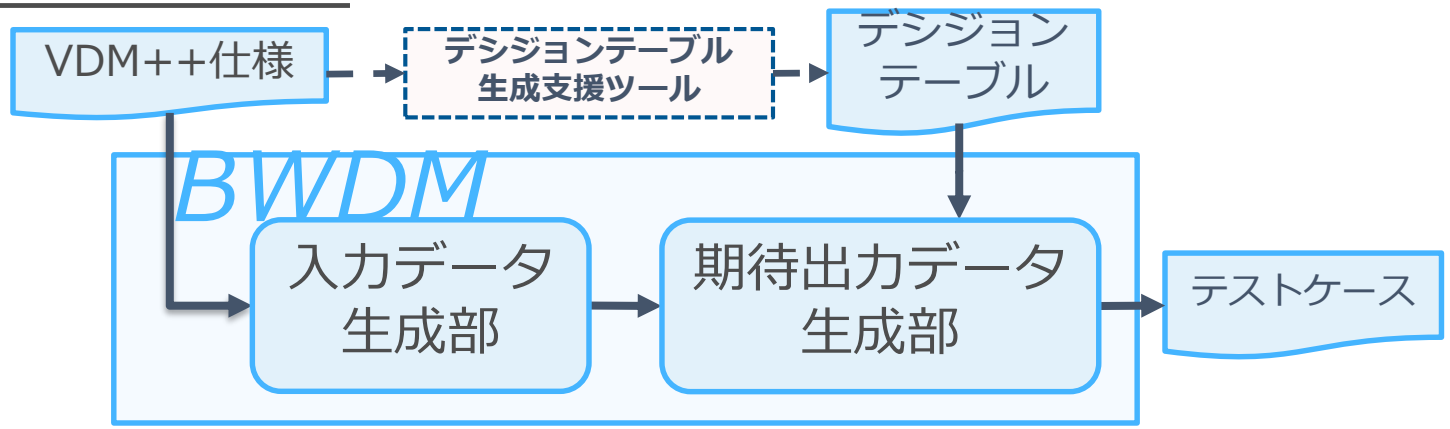
BWDM内において、

生成した入力データに対する期待出力データの導出に用いる

入出力とBWDM


入力
出力
VDM++仕様
デシジョンテーブル
生成支援ツール
デシジョン
テーブル
BWDM
入力データ
生成部
期待出力データ
生成部
テストケース

BWDMの処理



入力データ生成部

- VDM++仕様の構文解析、条件式抽出
- 境界値分析
- 入力データ生成

期待出力データ生成部

- デシジョンテーブル読み込み
- 期待出力データ生成
- テストケース生成・出力

テストケース自動生成ツールBWDM

```
class Mix

functions

混在仕様 : nat * int -> seq of char
  混在仕様(arg1, arg2) ==
    if(arg1 mod 2 = 0) then
      if(0 <= arg2) then
        "arg1:偶数、arg2:正の数"
      else
        "arg1:偶数、arg2:負の数"
    else
      if(arg2 < 0) then
        "arg1:奇数、arg2:負の数"
      else
        "arg1:奇数、arg2:正の数";
```

テストケースNo.	入力データ		-->	期待出力データ
No.1	natMin-1	intMin-1	-->	Undefined Action
No.2	natMin-1	intMin	-->	Undefined Action
No.3	natMin-1	intMax	-->	Undefined Action
No.4	natMin-1	intMax+1	-->	Undefined Action
No.5	natMin-1	0	-->	Undefined Action
No.6	natMin-1	-1	-->	Undefined Action
No.7	natMin	intMin-1	-->	Undefined Action
No.8	natMin	intMin	-->	arg1:偶数、arg2:負の数
No.9	natMin	intMax	-->	arg1:偶数、arg2:正の数
No.10	natMin	intMax+1	-->	Undefined Action
No.11	natMin	0	-->	arg1:偶数、arg2:正の数
No.12	natMin	-1	-->	arg1:偶数、arg2:負の数
No.13	natMax	intMin-1	-->	Undefined Action
No.14	natMax	intMin	-->	arg1:奇数、arg2:負の数
No.15	natMax	intMax	-->	arg1:奇数、arg2:正の数
No.16	natMax	intMax+1	-->	Undefined Action
No.17	natMax	0	-->	arg1:奇数、arg2:正の数
No.18	natMax	-1	-->	arg1:奇数、arg2:負の数
No.19	natMax+1	intMin-1	-->	Undefined Action

VDM++仕様

自動生成

テストケース

BWDM: Boundary Value & Vienna Development Method

今後の課題

適用範囲が狭い

- 「剰余式を含みネストした条件式」からのテストケース生成を行えない（現在改良中）
- 両辺が変数である条件式に現在未対応
- 条件式は不等式と剰余式以外に未対応
- 実際のソースコードに近い記述でないとテストデータを出せない

GUI未実装

- 現在はCUI環境でのみ実行可能であるため使いずらい

柔軟なテストデータ生成が可能でない

- 境界値分析とテストデータ生成においてオプション指定等ができない

剰余式を含みネストした条件式

if(a mod 5 = 0) then
  if(a > 92) then
    “95, 100, 105, ...”
  else
    “..., 80, 85, 90”
else
  “others”

現状抽出するテストデータ：4, 5, 6, 92, 93

剰余式を含みネストした条件式

```
if(a mod 5 = 0) then
  if(a > 92) then
    “95, 100, 105, ...”      ・・・ 未実行
  else
    “..., 80, 85, 90”        ・・・ 5の場合実行
else
  “others”                   ・・・4, 6, 92, 93の場合実行
```

現状抽出するテストデータ：4, 5, 6, 92, 93

3行目が未実行となり、
テストケース自動生成処理として不十分である