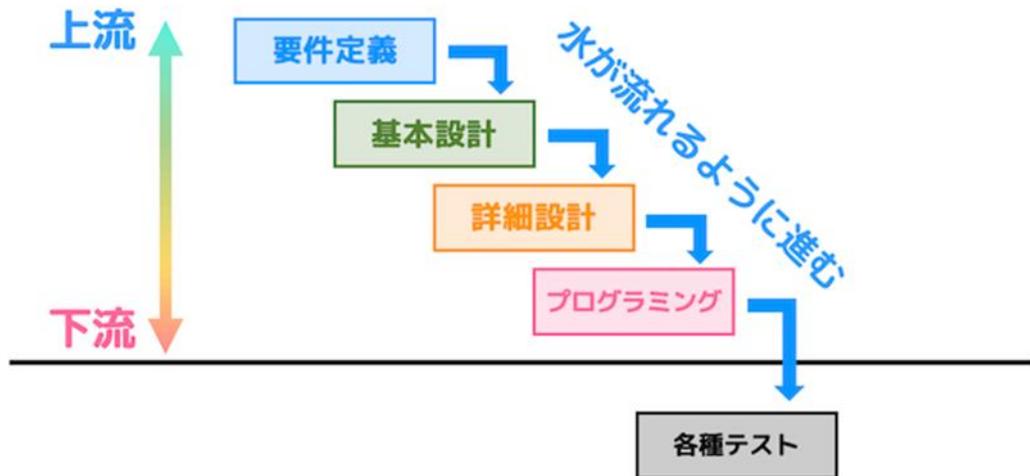


★前回の課題の解答

<Q>ウォーターフォールモデルのシステム開発工程の作業内容①～⑥を実施する順番に並べなさい。

[作業内容]

- ① 現状の問題点を調査・分析し，対象システムへの要求を定義する。 **要件定義**
- ② システムとして必要な機能をプログラムに分割し，処理の流れを明確にする。 **内部設計**
- ③ 詳細な処理手順を設計し，コーディングする。 **プログラミング**
- ④ テストを行う。 **各種テスト**
- ⑤ 各プログラム内の構造設計を行う。 **詳細設計（プログラム設計）**
- ⑥ システムの要求仕様を基に，システムとして必要な機能を定義する。 **外部設計**



答え：①⑥②⑤③④

⑤-1 モジュール分割 (module splitting)

- ・ 1つのプログラムを複数のユニットに分割する事

① モジュール分割のメリット

- a) モジュール単位で作成、テストを行えるため、並行して複数の作業を行うことができる
- b) モジュール単位で問題が起こった位置の特定、デバッグが行えるため、保守性に優れたプログラムを作ることができる

② モジュール分割の方法

a) データの流りに着目する方法

(1) STS分割法

プログラムを入力処理(Source)、メイン処理(Transform)、出力処理(Sink)の3つに分割する方法

STSは、Source(源泉), Transform(変換), Sink(吸収)の3つの頭文字

(2) トランザクション分割法 (TR分割法)

プログラムを処理内容ごとにトランザクションと呼ばれるグループに分け、トランザクションごとにモジュールを分割する方法

⑤-1 モジュール分割 (module splitting)

(3) 共通機能分割法

プログラム中にある共通な機能を1つにまとめて、共通モジュールとして独立、分割する方法

b) データの構造に注目する分割手法

(1) ジャクソン法

入力、出力両方のデータ構造に着目して分割する方法

(2) ワーニエ法

入力のデータ構造のみに注目して分割する方法

⑤-2 モジュールの独立度 (Module independence)

- ・モジュール分割を行う際は、それぞれのモジュールの設計を切り離して行うために、①それぞれのモジュールが明確である、②入出力データが明確である、必要があります。

そこでモジュールの独立性を測る尺度として、(A)モジュール強度、(B)モジュール結合度、の2つが使われるようになりました。

(A)モジュール強度

モジュール内にある機能の関連の強さを表す基準となるもの。具体的に言うと、モジュールがどれくらい単一の機能に特化して設計されているかの度合いを表す。強度は1⇒7で弱くなり、独立性も1⇒7で低くなる

1. 機能的強度

1つの機能を実行するためのモジュール、シンプルでわかりやすく、もっとも独立性が高い

2. 情動的強度

同じのデータを扱う複数の機能をまとめたモジュール、

3. 連絡的強度

複数の機能が順番に実行されるモジュール、各機能が同一データを扱っている

4. 手順的強度

複数の機能が順番に実行されるモジュール、各機能が別々のデータを扱っている

5. 時間的強度

初期設定処理、終了処理など同じタイミングで実行される複数の処理をまとめたモジュール

6. 論理的強度

関連のある複数の機能をまとめたモジュール、初期パラメータでどの機能を実行するかを選択できる

7. 暗号的強度

関連のない複数の機能をまとめたモジュール、

⑤-2 モジュールの独立度 (Module independence)

(B)モジュールの結合度

モジュール間の関連の度合いを表す基準となるのがモジュール結合度。具体的には、モジュール同士がどれだけ独立に、または関連して設計されたかの度合いを表す。

結合度	名称	内容
強い ↑	内部結合	他モジュールの内容を直接参照
	共通結合	複数モジュールが共通領域を使用
	外部結合	複数モジュールが共通データを使用
	制御結合	他モジュールのパラメータを渡す
↓ 弱い	スタンプ結合	データ構造を決めるパラメータを渡す
	データ結合	データパラメータのみの受け渡し

- ・ 変数を渡す際に引数で渡すのか、それとも外部宣言されたデータ（グローバル変数、大域変数）を参照するか
- ・ 引数として渡したのは単一のデータ(int型, double型)なのか構造体として渡したか
- ・ 制御パラメータを用いてモジュールの実行順序を制御するか

<Q>モジュール強度が最も高いものはどれか。

- ア：あるデータを対象として逐次的に複数の機能を実行するモジュール
- イ：異なる入力媒体からのデータを処理するモジュール
- ウ：単一の機能を実行するモジュール
- エ：特定の時点で必要とされる作業のすべてを含んでいるモジュール

※強度を決める基準は、

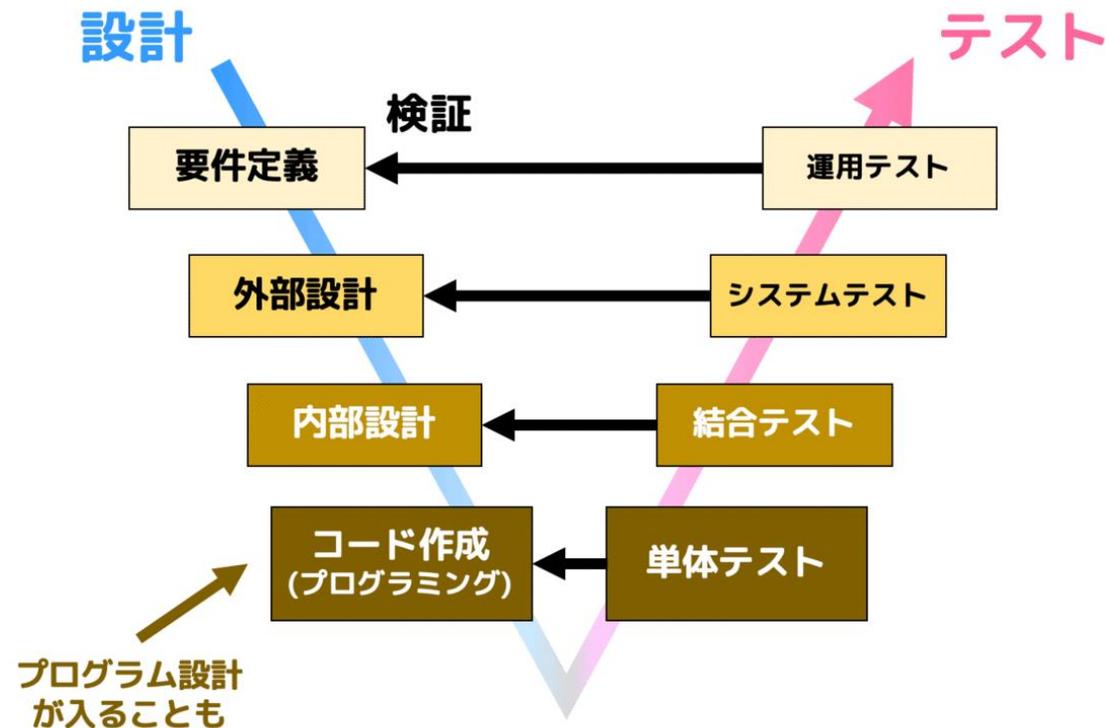
- ・単一の機能をモジュールにしたのか、複数の機能をモジュールにしたのか
- ・複数の機能をモジュールにしている場合、順番が関係あるのかないのか
- ・モジュールの入力、出力データは共通なのか機能ごとに独立しているか

<Q>モジュールの独立性を高めるには、モジュール結合度を低くする必要がある。モジュール間の情報の受渡し方法のうち、モジュール結合度が最も低いものはどれか。

- ア：共通域に定義したデータを関係するモジュールが参照する。
- イ：制御パラメタを引数として渡し、モジュールの実行順序を制御する。
- ウ：データ項目だけをモジュール間の引数として渡す。
- エ：必要なデータを外部宣言して共有する。

⑦-1 テスト手法

- ・システム開発では「ミスが減らす」のではなく、ミスを発見して、つぶすことでプログラムのバグを防いでいる。この「ミスを発見して、つぶす」こと（デバッグ）をテストと呼ぶ。テストは、①小規模（各モジュールごと）でテスト[単体テスト]、②規模を大きくしてテスト[結合テスト]、③システム全体を稼働させてテスト[システムテスト]、④利用者視点でシステムが問題なく動くかテスト[運用テスト]、という4ステップで行う。



⑦-1 テスト手法

(A) 単体テスト

・単体テストでは、モジュールが思った通りの動きをするのかをテストする。ただし、やみくもに入力（テストデータ）を決めてもあまりテストの効果はないので、「ある一定の規則」に従ってテストする入力の種類を決定する。

① ブラックボックステスト (BlackBox Test)

モジュール（プログラム）の内部をブラックボックスとして、利用者視点で行われるテスト。与えられたテストデータに対して、正しく出力が得られるかをチェックする。主に「同値分割」、「境界値分析（限界値分析）」と呼ばれる方法でテストをする入力の種類（テストケース）を作成する。

①-1 同値分割

同値分割とは、データをある一定の範囲ごとにグループ化し、グループの中で代表的な値を1つ選ぶ方法

①-2 境界値分析

境界値分析とは、データをある一定の範囲ごとにグループ化したときのグループの境目付近（境目+1 or 境目-1が多い）を重点的にチェックする方法

⑦-1 テスト手法

② ホワイトボックステスト (WhiteBox Test)

ホワイトボックステストは、モジュールの内部に注目して行うテスト。例えばプログラムの条件分岐に着目して、プログラムが正しく分岐するかを調べる。

- (a) 命令網羅 全ての命令が最低1回通るように試す
- (b) 分岐網羅 全ての分岐が最低1回通るように試す
- (c) 条件網羅 個々の条件の真・偽をそれぞれ最低1回試す
- (d) 複数条件網羅 複数ある条件の組合せをすべて試す。

<Q>以下の関数 ex2 において、①命令網羅、②分岐網羅、③条件網羅、④複数条件網羅、を満たすテストデータの例を示しなさい。ただし、テストデータの個数が最小個数になるようにすること。

```
def ex2(x, y, z):  
    if x == 3 and y <= 5:  
        print("処理1")  
    else:  
        print("処理2")  
  
    if z > 0:  
        print("処理3")
```

⑦-1 テスト手法

(B)結合テスト

- ・ 単体テストをクリアすると、クリアしたモジュールをつなぎ合わせて結合テストを行う。

① トップダウンテスト (Top Down Test)

トップダウンテストは、モジュールの上位側から順番にテストをしていく方法。まだ完成していないモジュールがある場合は、**スタブ (stub)** と呼ばれる仮のモジュールを付けてテストを行う。

② ボトムアップテスト (Bottom Up Test)

ボトムアップテストは、モジュールの下位側から順番にテストをしていく方法。まだ完成していないモジュールがある場合は、**ドライバ** と呼ばれる仮のモジュールを付けてテストを行う。

(C)信頼性成長モデル (ゴンベルツ曲線)

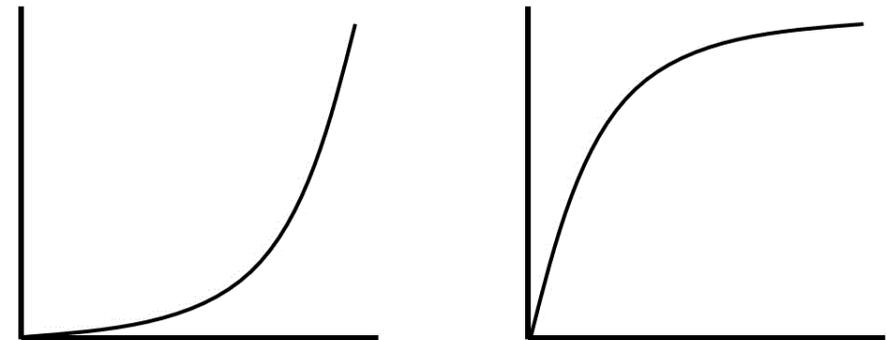
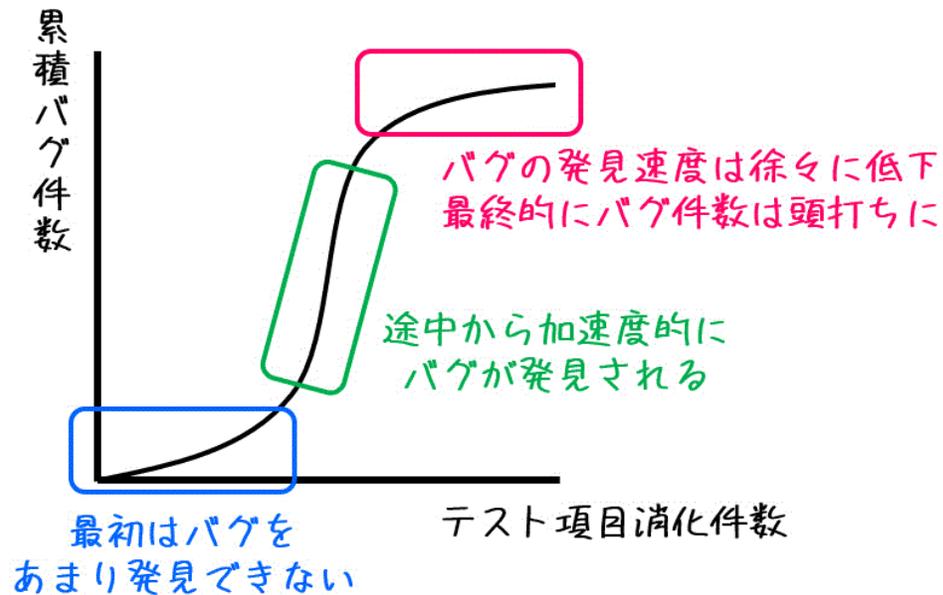
プログラムのバグを見つけ、修正して...、また見つけ...、を繰り返していくことでバグが十分に発見された「品質のよいシステムだ」と言える。「品質のよいシステムだ」と判断するために使うものが信頼性成長モデル (ゴンベルツ曲線) である。

⑦-1 テスト手法

(C)信頼性成長モデル（ゴンベルツ曲線）

プログラムのバグを見つけ、修正して...、また見つけ...、を繰り返していくことでバグが十分に発見された「品質のよいシステムだ」と言える。「品質のよいシステムだ」と判断するために使うものが信頼性成長モデル（ゴンベルツ曲線）である。

信頼性成長モデルでは、①最初はバグが緩やかに発見され、②途中から急激にバグが発見さ、③最終的にバグの発見は緩やかになり頭打ちになる



<Q> 整数 1~1,000 を有効とする入力値が、1~100 の場合は処理Aを、101~1,000 の場合は処理Bを実行する入力処理モジュールを、同値分割法と境界値分析によってテストする。次の条件でテストするとき、テストデータの最小個数は幾つか。

☆条件☆

- ① 有効同値クラスの1クラスにつき、一つの値をテストデータとする。ただし、テストする値は境界値でないものとする。
- ② 有効同値クラス、無効同値クラスの全ての境界値をテストデータとする。

ア : 5
イ : 6
ウ : 7
エ : 8