

オブジェクト指向における段階的開発への誘導を目指す 初級 Java 演習支援の課題検討

Problem Examples of Applied Java Programming Exercise for Stepwise Development in Object-Oriented Style

辻 健人^{*1}, 玄馬 史也^{*1}, 富永 浩之^{*1}
Kento TSUJI^{*1}, Fumiya GEMBA^{*1}, Hiroyuki TOMINAGA^{*1}

^{*1}香川大学工学部

^{*1}Faculty of Engineering, Kagawa University
Email: s12t241@stmail.eng.kagawa-u.ac.jp

あらまし：初級 Java 演習では，オブジェクト指向の基本概念を習得し，各オブジェクトの機能を実装していく．次に，概念の抽象化に基づいて，オブジェクト間の振舞いを忠実に記述する．本研究では，オブジェクト群の責務を意識した開発手法を目的とする演習を提案する．課題としての仕様の提示だけでなく，各種のテストを用意し，段階的詳細化を体感させる．さらに，試練テストを導入し，責務の分担の明確化を検証する設計手法を学ぶ．本論では，本演習で題材とする課題について検討する．

キーワード：オブジェクト指向，応用 Java プログラミング，スパイラル開発，テスト手法

1. はじめに

大学情報系学科では，初年次の初級プログラミング演習として，手続型の C 言語を採用することが多い．ここで，基礎的な文法事項，データ構造と基本的なアルゴリズム，フローチャートの設計，要件定義とウォーターフォール開発を習得する．

続いて，Java 言語などで，オブジェクト指向プログラミングに進む．その入門的なコーディング段階では，文法的な学習項目の比重が大きい．設計と言っても，構造化フローチャートの延長線で捉えたり，メソッドの外部仕様で留まってしまうことが多い．これは，オブジェクトの即物的な描写に過ぎない．そうした初級プログラミングから脱却するには，オブジェクト指向に沿った設計を理解し，スパイラル開発に基づいた実装を習得する必要がある．

2. 本提案の OOP 開発演習の教育目標

本研究では，初級 Java のプログラミング演習に続いて，オブジェクト指向による設計と実装を実践的に行う演習に焦点を当てる⁽¹⁾．そのための演習支援システムを開発する．ここでは，オブジェクトの記述の段階的な進展を，図 1 のように捉える．特に，内部状態を明確にした概念的なオブジェクトの抽象化と，メッセージ通信によるオブジェクト間の振舞いを意識した記述に着目する．さらに，オブジェクトのロールから責務を抽出することも重要である．

本演習では，上記のモデルに基づき，3 つの教育目標を掲げる．1 つは，オブジェクトの責務を意識したモデリング設計に馴染ませる．2 つは，オブジェクト指向の一般的な語法に沿ったコーディングを習慣付けさせる．3 つは，段階的なテストを中心にした開発手順を体感させる．対象者は，大学情報系学科の 3 年次程度とし，3～4 人のグループで，チーム開発を体験させる．

本研究では，教育効果の高い問題設定や必要機能

を議論し，演習支援システムを開発する．まずは，プログラムの機能要件の外部評価である入出力サンプルでの実行結果の確認を実装する．次に，その拡張として，非機能要件の内部評価に，システムによる自動的な検証や支援を導入していく．

3. 本提案の OOP 開発演習の出題構成と意図

本提案の演習では，与えられた課題の解答コードをチームで作成する．概略的な設計，部分的な実装，全体的な完成，機能の拡張，品質の向上と進めていく．事前に，教師は，満たすべき仕様や入出力サンプルを構築し，模範コードを作成しておく．設問に従い，メソッド単位で適当な箇所を幾つか隠蔽し，穴空きの出題コードとする．ただし，1 つのプログラムを共同で完成させるというより，チームの一員として，他のメンバとの協力関係の下で，自己の分担を意識させることを重視する．

演習は，前半の外部評価と後半の内部評価に分ける．前半では，与えられた仕様に従い，設問ごとに基礎的な内容のコーディングに取り組む．プログラムの外部評価として，入出力データによる実行結果の照合で成果を判定する．後半では，コードの品質も意識し，リファクタリングを行ったり，コードの修正や拡張に取り組む．プログラムの内部評価として，各種のテストで判定する．

4. 課題の進行と各種のテスト

前半の外部評価は，個別と集結の 2 種類の実行テストで行う．個別テストでは，メンバ各自が提出した解答コードで，模範コードの該当部分を置き換えて差し込み，実行する．集結テストでは，チーム全体の解答コードを模範コード上で集結し，実行する．

各メンバがコーディング作業を進め，個別テストに至る．個別テストでは，メソッドやクラス単位でコードを提出し，模範コードと結合して実行することで入出力の検証を行う．

集結テストでは、メンバが解答したメソッドやクラスの中で、使用するものを決定し、1つのプログラムとする。個別テストでエラーが生じなくとも、全体の整合性が満たされなければエラーが生じたり、実行結果が正しくない状態が発生したりする。

後半の内部評価では、実装の適切さを確認するため、検証テストに挑戦する。チームの解答コードと試練コードを結合し、試練プログラムを実行して評価する。試練コードは、テスト駆動開発における、テストコードおよびその拡張である。正誤だけではなく、非機能要件としての処理速度や応答性など、パフォーマンスも測定する。例えば、カプセル化について、オブジェクト変数へのアクセス修飾子を正しく設定しているかどうか、試練コードからの攻撃的なアクセスで確認する。また、継承について、下位クラスが、与えられた上位クラスから正しく派生されているかどうか確認する。

5. 部分整列を題材とする設問構成

本演習で扱う課題として、整列算法を取り上げ、演習の進行に沿った設問を議論する。整列算法は、C言語のプログラミングとして幅広く習得しており、オブジェクト指向におけるデータの抽象化の導入に向いている。ここで、仕様の変更や拡張に伴うプログラムの修正も念頭に置いた設計と実装を誘導する。

(1) データ構造の主要クラスの概略的な設計

まず、数値データのクラス NumData を用意する。2つのデータが正順かどうかを判定する比較メソッド isOrder() を実装する。次に、データ列のクラス Sequence を用意する。データの比較メソッドによって、整列済みかどうかの内部状態を判定するメソッド isSorted() を用意する。また、格納順で標準端末に出力するメソッド output() を実装する。i番目とj番目のデータを交換するメソッド swap() を実装する。

(2) データ構造の生成

対象のデータ列を生成する抽象クラス SequenceGenerator の下に、幾つかの下位クラスを用意する。NormalGenerator クラスは、0,1,2,...,n の整列済みのデータ列を生成する。InputGenerator クラスは、標準端末からのデータ列の入力である。RandomGenerator クラスは、ある範囲の乱数によるデータの格納である。同じデータが幾つか格納されることもある。ShuffleGenerator クラスは、整列済みのデータ列の攪拌による乱列の生成である。

(3) 仕様に沿った関連クラスの部分的な実装

SequenceGenerator の下位クラスを実装する。InputGenerator では、メソッド input() の処理を実装する。標準端末から文字列として与えられたデータを、NumData クラスのインスタンスに格納し、それをコレクションに push していく。

(4) データの汎化と抽象クラス

数値データのクラスを汎化して、抽象クラス Data を用意する。データとしては、実数値、文字列、複

数値のベクトルなども扱う。比較メソッド isOrder() をデータに合わせてオーバーライドする。文字列ならば辞書式順序、ベクトルならば総和や二乗平均での大小を用いる。また、SequenceGenerator の下位クラスも必要な修正を行う。

(5) 特化クラスでの整列算法の具体実装

整列処理の抽象クラス Sort クラスを用意し、整列メソッド sort() を定義する。このメソッドを実行後は、Sequence の内部状態は整列済みになる。Sort クラスの特化クラスとして、単純挿入法 InsertSort とクイックソート QuickSort を実装する。それぞれ、メソッド sort() をオーバーライドする。クイックソートでは、基準値 pivot を与えて大小の部分列に振り分けるメソッド partition() が必要である。

(6) 機能の改良と拡張

機能の改良として、整列の高速化を考える。算法としては、データ数が多いときクイックソートを用い、データ数が少なくなったら単純挿入法に切り替えることにする。また、機能の拡張として、部分整列を考える。入力として与えられた N 個のデータ列に対し、大小関係を比較し、上位 R 個のみを高速に整列して出力する。クイックソートを用いる場合、左右の振分けに必要な部分だけに適用する。

6. おわりに

初級 Java 演習では、オブジェクト指向の基本概念を習得し、各オブジェクトの機能を実装していく。

次に、概念の抽象化に基づいて、オブジェクト間の振舞いを忠実に記述する。本研究では、オブジェクト群の責務を意識した開発手法を目的とする演習を提案する。各種のテストを用意し、段階的詳細化を体感させる。さらに、検証テストを導入し、責務の分担の明確化を検証する設計手法を学ぶ。本論では、整列処理を題材として、課題を検討した。今後は、責務クラスのリファクタリングなどを考慮したボートゲーム課題なども考える。品質の向上や試練テストに関わる設問も具体化する。

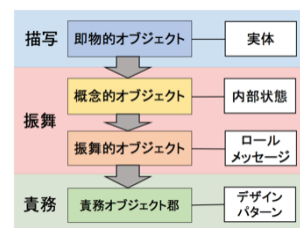


図1 オブジェクトの記述の段階的な進展

参考文献

- (1) 辻健人, 玄馬史也, 富永浩之: “オブジェクト指向の入門的な開発スタイルを意識させる初級 Java プログラミングのグループ演習の支援システムの GUI 検討”, 情報処理学会 第 79 回全国大会, Vol.79, No.2ZC-04, pp.649-650 (2017).