# 参照に重点を置いたオブジェクト指向プログラミング入門教育の工夫

土肥紳一[†1]

オブジェクト指向プログラミング入門教育を担当する中で，初学者の理解が難しい項目がある．それは，コレクションクラスの ArrayList オブジェクトとインタフェースである．これらの理解が，難しい理由を受講者の様子を観察する中で，参照の理解が十分でないことに気が付いた．本論文では，参照に重点を置いたオブジェクト指向プログラミング入門教育の工夫について述べる．

# Devises of teaching method in Object Oriented Programming Education for novice programmer focusing on reference of object

SHINICHI DOHI[†1]

There are so difficult items to learn object oriented programming education at novice programmer. It is an ArrayList object in Java collection class and interface. While observing some novice programmer in the class of the computer programming B, I found that some novice programmer do not understand enough an ArrayList object and interface. In this paper, I describe devises of teaching method in Object Oriented Programming Education for novice programmer focusing on reference of object.

## 1. はじめに

初学者にとって，オブジェクト指向プログラミングの学習は難しい．オブジェクト指向の概念を初学者に理解させるためには，初学者がオブジェクトをイメージし易い教授の工夫が必要である．その工夫の一つとして，身近な物を活用しながらその概念をイメージさせることが挙げられる．オブジェクトの関係をイメージできるようになった次の段階は，Java 言語で実装することである．先行研究では，オブジェクト指向言語を学習する上での，クラスとオブジェクトの定義を理解するためのワークベンチがある[1]．さらに，オブジェクト指向の初等教育における可視化を支援する取り組みが報告されている[2].

筆者はオブジェクト指向プログラミングの入門の科目である「コンピュータプログラミング B」を担当する中で，受講者のモチベーションの向上を目指す教授法(SIEM:ジーム)を実践している[3]．SIEM は授業毎に理解度調査を目的としたアンケート調査を実施しており，この結果から初学者にとって理解の難しい内容は，コレクションクラスの ArrayList オブジェクトとインタフェースであることが分かった[4]．引数や返却値を使ったプログラム分割の概念は，事前履修条件になっている「コンピュータプログラミング A」で学習する．受講者の様子を観察する中で，理解を妨げている本質は，オブジェクトの参照の理解が不十分であることに気が付いた．この問題を解決するための教授の工夫とその効果について述べる．

## 2. 授業について

(1) Java に関する科目構成について

本学部の Java に関する科目構成は，「コンピュータプログラミング A」「コンピュータプログラミング B」「オブジェクト指向設計」となっている．「コンピュータプログラミング A」は，手続き型の考え方の入門を学習する．授業の後半では，「コンピュータプログラミング B」の学習に繋がるように引数，返却値を活用したプログラムの分割に重点を置いている．「コンピュータプログラミング B」は，オブジェクト指向の入門を学習する．クラス図と API(Application Program Interface)仕様が与えられると，受講者がプログラムを実装できる能力を身に付ける．「オブジェクト指向設計」は，ソフトウェアの分析・設計を学び，クラス図と API 仕様を受講者自身で定義できることを目指している．受講者は，これら 3 科目の学習によって，プログラムの分析・設計から実装までを行えるようになる．

(2) 「コンピュータプログラミング B」の授業内容

2014 年春セメスターに開講した「コンピュータプログラミング B」の授業内容を，表 1 に示す．授業は現在 5 クラスに分割しており，1 クラスの受講者数は 40～50 名である．開講時間は，月曜日と水曜日の 14:30 から 16:20(途中 10 分の休憩)となっており，授業回数は 1 セメスターで 27 回となる．左の列から「授業回数」「章」「主な内容」の順に記載した．2 回目から 5 回目の授業は，2 章の内容である．この章は，専門用語がたくさん出現するため，受講者がその概念を十分に理解できる様に，授業時間に余裕を持たせている．3 章では複数のサイコロ(Dice オブジェクト)を扱うために ArrayList オブジェクトを活用する．4 章では 3 章で学習した ArrayList オブジェクトをさらに活用し，カップ(Cup オブジェクト)を作る．これらの関係を示すために，Dice クラスの API 仕様を**表 2** に，Cup クラスの API 仕様を**表 3** に，クラスの関係を**図 1** に示す.

†1 東京電機大学情報環境学部
Tokyo Denki University, School of Information Engineering

表 1「コンピュータプログラミング B」の授業内容

| 回数 | 章 | 主な内容 |
|---|---|---|
| 1 | 1 | ガイダンス，Java，サクラエディタのインストール等 |
| 2 | 2 | 非手続き型言語，オブジェクト，オブジェクトの生成等 |
| 3 | | オブジェクト図，状態・振る舞いの追加等 |
| 4 | | クラス図とソースプログラムの関係等（Student クラスの完成） |
| 5 | | クラス図とソースプログラムの関係等（Teacher クラスの完成） |
| 6 | 3 | サイコロ(Dice)オブジェクトの生成等（Dice クラスの完成） |
| 7 | | 複数の Dice オブジェクトの生成と ArrayList オブジェクトによる格納等 |
| 8 | 4 | Cup クラスの完成等 |
| 9 | | Book クラス，Bookshelf クラス等 |
| 10 | 5 | キーボードからの入力，DiceGame クラス等 |
| 11 | | 特殊なサイコロ，インタフェース等 |
| 12 | | 総合復習 |
| 13 | | 中間試験 |
| 14 | 6 | Coin クラス，Check クラス，インタフェースの活用等 |
| 15 | | CoinBox クラス等 |
| 16 | | Bookshelf(4 章の内容)，Wallet クラス等 |
| 17 | 7 | Educatee クラス等 |
| 18 | | 参照，ArrayList 等 |
| 19 | 8 | ビンゴゲーム(Ball インタフェース等) |
| 20 | | ビンゴゲーム(Box インタフェース等) |
| 21 | | ビンゴゲーム(Bingo インタフェース等) |
| 22 | | ビンゴゲーム(全体のクラス図の関係等) |
| 23 | 9 | 継承等 |
| 24 | 10 | 応用 |
| 25 | | 総合復習 |
| 26 | | 期末試験 |
| 27 | 11 | ストリームについて |

表 2 Dice クラスの API 仕様

| Dice | API仕様 |
|---|---|
| random:Random = new Random()<br>value:int | |
| Dice()<br>cast():void<br>getVAlue():int | コンストラクターです.<br>1から6の乱数を生成し，状態のvalueに格納します.<br>状態のvalueを返却します. |

表 3 Cup クラスの API 仕様

| Cup | API仕様 |
|---|---|
| arrayList:ArrayList<Dice> =<br>new ArrayList<Dice>() | |
| Cup()<br>add(dice:Dice):void<br>get(number:int):Dice<br>size():int<br>cast():void<br>getSum():int<br>getValue(number:int):int | コンストラクターです.<br>引数でDiceオブジェクトを受け取り，ArrayListにaddします.<br>引数numberで指定したオブジェクトの参照を返却します.<br>Cupオブジェクトの中のDiceオブジェクトの個数を返却します.<br>Cupオブジェクトの中のDiceオブジェクトを全て振ります.<br>Cupオブジェクトの中のDiceオブジェクトの目の合計を返却します.<br>引数numberで指定したDiceオブジェクトの値を返却します. |


図 1 クラスの関係

5 章以降は，貯金箱(CoinBox オブジェクト)，財布(Wallet オブジェクト)，本棚(Bookshelf オブジェクト)等へオブジェクトの例を変えながら，インタフェース等の概念を学習する．取り上げるオブジェクトの例は異なっているが，ArrayList オブジェクトで管理するオブジェクトが，貨幣(Coin オブジェクト)や本(Book オブジェクト)に変わっただけであり，同じ関係になっている．講義内容の詳細は，文献[5]を参照されたい．

(3) 授業の実施形態

本学部は 2001 年の開設当初から，BYOD(Bring Your Own Device)に取組んでおり，ノート PC は学生諸君の必携とし，学部の授業で必要となる有償ソフトウェアは大学が提供している．「コンピュータプログラミング B」の授業では，Java，テキストエディタ，コマンドプロンプト利用し，有償ソフトウェアは利用していない．Eclipse 等の統合環境を活用する方法もあるが，初学者にとって Java の本質でないことがたくさん発生するため，使っていない．授業のアシスタントは，原則として大学院生の TA(Teaching Assistant)が 2 名，学部生の SA(Student Assistant)が 2 名で担当する．

(4) 身近な物を活用した教授の工夫

授業では身近な物を活用しながら，オブジェクトの関係をイメージさせる工夫を取り入れている．一例であるがサイコロオブジェクトは，段ボールで大きなサイコロを作った．この様子を図 2 に示す．段ボールに蓋をあけ，オブジェクトの中に状態や振舞いがあることを解説した．

図 2 段ボールで作ったサイコロの例

大きなサイコロを活用する事によって，振舞いの動作を強調して見せることができる．例えばサイコロを振る動作は，大きなサイコロを転がすことによって実演し，Java で実装する場合は，振舞いはサイコロオブジェクトの中にあることを説明した．オブジェクトを大きく見せることが，理解度の向上に繋がると考えていた．しかし，教授者から遠く離れて着席している受講者には，その様子を十分に伝えることはできなかった．教室はグループ学習形式のレイアウトとなっており，教授者に背を向けて着席する受講者もいる．この問題の回避策として，書画カメラを使って 3 面あるスクリーンに投影することを考えたが，サイコロが大き過ぎて書画カメラで撮影できなかった．現在は，小さなサイコロを書画カメラで撮影し，拡大しながら説明している．

## 3. 理解度調査結果

筆者のクラスでは，長年 SIEM を実践しており，授業毎に理解度調査を目的としたアンケートを実施している．理解度調査は，授業毎に数個の調査項目を設け，「はい」「いいえ」の 2 択で回答を得る．なお，過去の調査には実施していない項目もあり，この場合は実在するデータのみで平均値を算出した.図 3 は 2005 年から 2014 年までを対象に，延べ 113 項目の理解度調査結果について，平均値を示したものである．横軸は，質問項目の番号を，縦軸は「はい」と回答した割合(%)である．理解度調査結果が 80%を大きく下回った場合は，次回の授業で補足説明等を行う．理解度調査結果が平均値と比較して上昇していれば，授業が上手く進行しており，逆に平均値よりも低下していれば，授業が上手く進行していないと判断できる．

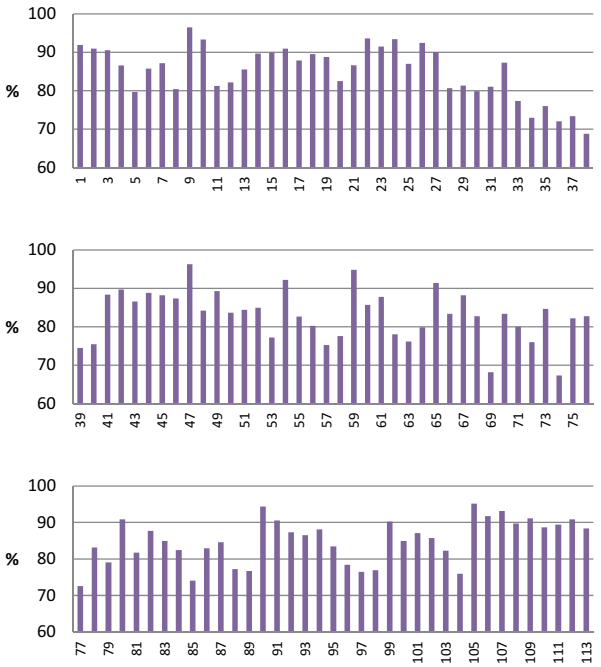



図 3　理解度調査結果(2005 年～2014 年の平均値)

表 4 は，理解度が 80%を下回る質問項目を抽出したものである．左の列から「番号」は質問項目の番号，『「はい」の割合』，「質問項目」，「章番号」，「備考」の順に示した．番号 33 から番号 40 までは，4 章の内容である．4 章は複数のサイコロ(Dice オブジェクト)を扱うために ArrayList オブジェクトを活用し，カップ(Cup オブジェクト)を作る．番号 53，57，58 は，5 章の内容である．サイコロの目の合計が偶数か奇数かを当てるゲーム(DiceGame オブジェクト)を取り上げる．番号 53 は勝敗判定の内容である．勝敗の判定は ArrayList オブジェクトが管理している Dice オブジェクトを順に参照し，目の合計を求める必要がある．合計を求める処理で繰り返しが必要になるため，初学者にとって難しい内容である．番号 57，58 はサイコロに，定義した Castable インタフェースを実装し，偶数しか出ないサイコロ(EvenDice)，奇数しか出ないサイコロ(OddDice)を同一視して扱うことを学ぶ．番号 62 から 64，69，72，74，77，79 は，6 章の内容である．Value インタフェースを定義し，これを実装する貨幣(Coin オブジェクト)，小切手(Check オブジェクト)を取り上げ，貯金箱(CoinBox オブジェクト)で扱う．その後，授業はビンゴゲームへと続く流れになっている．なお，各オブジェクトのクラス図および API 仕様等は割愛した．

表 4　理解度調査結果が 80%を下回る項目

| 番号 | はい | 質問項目 | 章 | 備考 |
|---|---|---|---|---|
| 5 | 79.6 | クラスとオブジェクトの違い | 2 | |
| 30 | 79.8 | ArrayList の get メソッド | 3 | ArrayList |
| 33 | 77.3 | Cup クラスの add メソッド | 4 | ArrayList を使って Cup オブジェクトを作成する． |
| 34 | 73.0 | Cup クラスの get メソッド | | |
| 35 | 76.0 | Cup クラスの size メソッド | | |
| 36 | 72.0 | Cup クラスの cast メソッド | | |
| 37 | 73.3 | Cup クラスの getSum メソッド | | |
| 38 | 68.8 | Cup クラスの getValue メソッド | | |
| 39 | 74.5 | 引数 | | |
| 40 | 75.4 | 返却値 | | |
| 53 | 77.2 | DiceGame クラスの judge メソッド | 5 | |
| 57 | 75.2 | インタフェースの抽出 | | インタフェースを実装したオブジェクトを ArrayList で扱う．貯金箱，本箱，財布を取り上げる． |
| 58 | 77.5 | インタフェースの実装方法 | | |
| 62 | 78.0 | 小切手クラス(Check) | 6 | |
| 63 | 76.1 | 小切手クラス(Check)に Value インタフェースを実装する方法 | | |
| 64 | 79.9 | プログラム B6_4 の ArrayList オブジェクトに Coin50,Coin100,Check オブジェクトが入る理由 | | |
| 69 | 68.1 | CoinBox クラスの remove メソッド | | |
| 72 | 75.9 | CoinBox クラスの print メソッド | | |
| 74 | 67.3 | Bookshelf クラスの get メソッド | | |
| 77 | 72.6 | Wallet クラスの remove メソッド | 4,6 | |
| 79 | 79.1 | Wallet クラスの getSum メソッド | | |
| 85 | 74.0 | シーケンス図 | 7 | UML，参照の理解を深める． |
| 88 | 77.2 | EducateeList クラス | | |
| 89 | 76.7 | EducateeList の API 仕様 | | |
| 96 | 78.4 | ConcreteBox クラス | 8 | インタフェースを実装したオブジェクトを ArrayList で扱う． |
| 97 | 76.5 | 演習 8-6 のオブジェクト図 | | |
| 98 | 76.9 | プログラム B8_2 | | |
| 104 | 76.0 | Ball インタフェースの実装によって，ビー玉も容易に扱えること | | |

## 4. 躓きの観察

机間指導を行う中で，受講者がどのような点で躓いているのかを観察した．

(1) クラスが実行されることの誤解

先に学習する「コンピュータプログラミング A」の授業では，引数と返却値を活用したプログラム分割を学習する．この授業では main メソッドを含むクラスメソッドでプログラムを学習するため，プログラムの実行はクラスの中が実行されることを理解している．オブジェクト指向では，new によってオブジェクトが生成されるが，生成されたオブジェクトの中に状態や振舞いがあることが理解されにくく，メソッドはクラスを記述したプログラム自体が，サブ

ルーチンや関数の様に実行される錯覚に陥り易い.

(2) オブジェクトの状態を直接アクセス

オブジェクトの状態は，通常，非公開(private)にしてプログラムを記述する．受講者は，変数の型のうち基本データ型だけを教わっている．受講者は，カプセル化の概念を十分に理解していないため，セッターやゲッターを活用することに慣れていない．オブジェクトの中の状態を直接アクセスすることを考えてしまう．

(3) オブジェクトの入れ子

複数のオブジェクトを組み合わせて新しくオブジェクトを定義する場合，オブジェクトが入れ子になる構造が出現する．受講者は，入れ子になっているオブジェクトの振舞いを，サブルーチンや関数と捉えて直接アクセスすることを考えてしまう．

(4) クラス図の改変

クラス図は，プログラムの設計図である．受講者の都合に合わせ，状態や振舞いを追加し，クラス図の仕様を改変しがちである．既存の振舞いを利用すれば良い場合でも，同じような内容のプログラムを記述する受講者も出てくる.

(5) 2 つの立場

オブジェクト指向では，オブジェクトを利用する立場と，オブジェクトを実装する立場の 2 つが存在する．受講者の中には 2 つの立場が交錯し，プログラムが組めなくなる人がいる．オブジェクトを利用する立場では，オブジェクトの実装は気にする必要は無い．しかし，オブジェクトを実装する立場では，仕様を満足するようにプログラムを記述しなければならないが，利用を考えてしまう人がいる.

(6) API 仕様の改変

API 仕様は，振舞いを定義するものである．クラス図と同様に，受講者の都合に合わせて振舞いの仕様を変更する人がいる.

(7) コンパイルエラーに誘発

授業では，クラス毎にファイルを保存するように指導している．複数のオブジェクトを使ったプログラムを作成する場合，コンパイルエラーが発生すると，受講者は見当違いのファイルを修正し始め，さらに状況を悪化させ，プログラムの破壊が始まる.

(8) 引数と返却値

引数，返却値の概念が理解できていない受講者がおり，オブジェクト指向を理解できない.

(9) オブジェクトを利用するプログラム

クラス図, API 仕様を満足するプログラムを完成した後，コンパイルエラーが無くなると，そのままプログラムを実行しようとする.「コンピュータプログラミング A」では，main メソッドが存在するプログラムを扱ってきたため，コンパイルが終了するとプログラムの実行を行えた. しかし，オブジェクト指向ではオブジェクトを利用するプログラムが別に必要となる.

## 5. 参照に重点を置いた代入の理解

参照に重点を置いた代入の理解の工夫を行った．以下にその内容を述べる.

(1) 小物の活用

クラスとオブジェクトの理解は，初学者にとって重要である. 受講者の理解を助けることを目的に,「たい焼きの型」と「たい焼き」の絵を見せ，クラスとオブジェクトの関係を説明したこともある．受講者は教わった時は理解できたような気になるが，断片的な理解に過ぎないことがうかがえる．試行錯誤を繰り返す中で，付箋紙，組紐，粘土，粘土の型，粘着テープ，鋏を使う方法を取り入れた．活用した小物を図 4 に示す. この中で粘土と型はセットになっており，100 円ショップで入手した.

図 4 活用した小物

(2) クラスとオブジェクトの理解

Java はオブジェクトを new によって生成する．通常，生成したオブジェクトは, クラス名と同じ型の変数を定義し，この変数に代入する.「コンピュータプログラミング A」では基本データ型しか学習していないため，変数自体にオブジェクトが格納される誤解が生じる．オブジェクトの代入は，オブジェクトへの参照を格納している．この理解を助ける工夫を図 5 に示す.

まず new Student("太郎")によってオブジェクトが生成される．この様子は，①粘土の型に粘土を「ニュー」と言いながら流し込み，必要に応じて付属のローラーで粘土を成形する．②粘土を型から取り出すと，型通りにオブジェクトが作られることを見せる．この時，型がクラスで粘土がオブジェクトであることを強調する. ③変数宣言の Student student の部分は，付箋紙を半分に折り student と記載し，変数に見立てる．半分に折った付箋紙の間に組紐を挟み，内側で組紐と付箋紙を粘着テープで固定する．④=の部分は，粘土を組紐が指し示すように配置する．この時，=はオブジェクトを参照していることを強調する．一連の様子

を書画カメラで見せることによって，変数にはオブジェクト自体が格納されているわけでなく，組紐が示すオブジェクトの参照を格納していることを分かり易く説明した．

Student student = new Student(“太郎”);

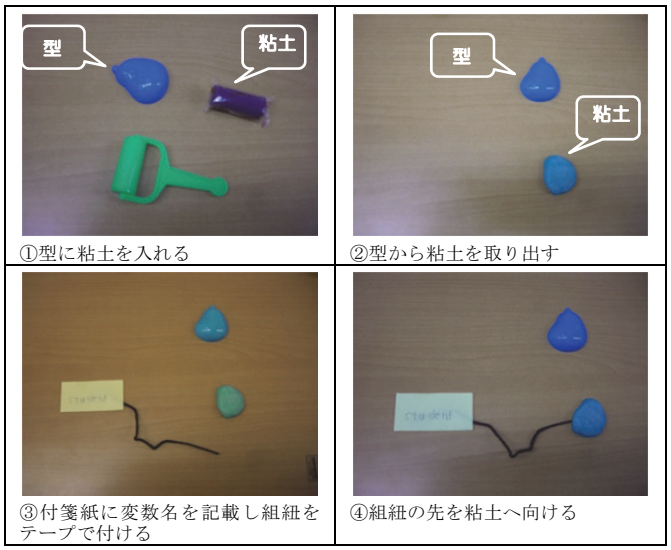



図 5　オブジェクトの生成

(2)複数のオブジェクト

複数のオブジェクトを扱うためには，通常，オブジェクトの個数に応じた変数を宣言し，オブジェクト毎に対応する変数へ代入する．図 6 の左側の図は，2 つの変数が個々にオブジェクトを参照していることを示している．図 6 の右側の図は，変数同士の代入を示している．この代入の結果，2 つの変数が組紐で同じオブジェクトを参照していることを示すことができる．

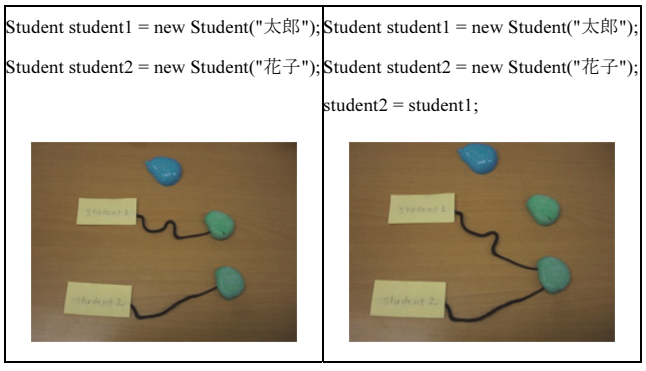



図 6　オブジェクトの代入

(3)null pointer exception

null pointer exception は，オブジェクトを何も参照しない状態で，オブジェクトの振舞いを実行しようとすると発生する．new を忘れる等，初学者が陥り易い例である．このことは，変数がオブジェクトを何も参照していない状況を見せることで容易に説明できる．この様子を図 7 に示す．左側の図は，new を忘れており，右側の図は代入を忘れている例である．

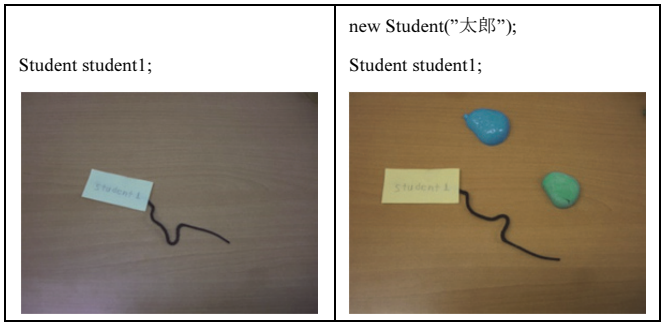



図 7　null pointer exception の例

## 6. 参照に重点を置いた ArrayList の理解

表 4 に示した通り，初学者にとって理解が難しいのは ArrayList オブジェクトである．授業では ArrayList オブジェクトを活用する例として，複数のサイコロを取り上げている．サイコロの他に，サイコロを入れる入れ物として蓋の付いた箱を使った．この箱が，ArrayList オブジェクトを意味している．

(1)　add メソッドの説明

add メソッドは，箱にサイコロを入れる操作と一致しており，受講者にイメージさせ易い．ArrayList オブジェクトは，サイコロを参照する情報が引数で渡され，サイコロ自体が渡されているわけでないことを説明する．以下のプログラムは，引数に new を指定することで，サイコロを参照する情報が，引数で渡されていることを知る良い例である．

```
ArrayList<Dice> diceBox = new ArrayList<Dice>();//①
diceBox.add(new Dice());  //②
diceBox.add(new Dice());  //③
diceBox.add(new Dice());  //④
```

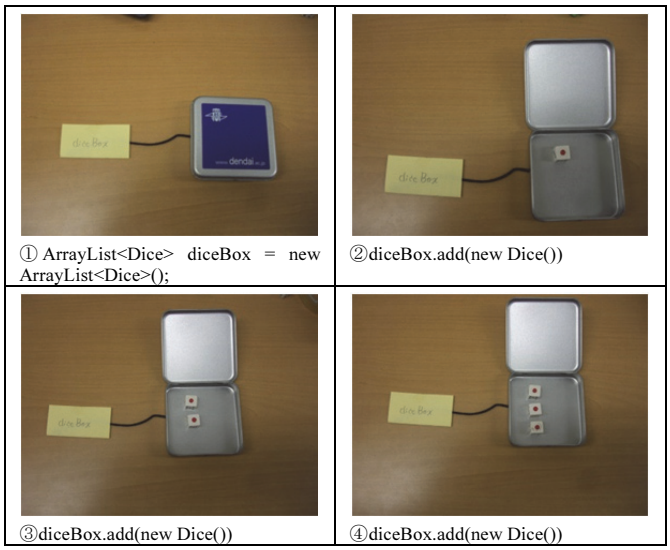


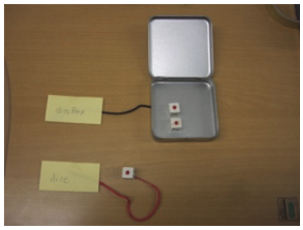

図 8　add の振舞い

推移の様子を図 8 に示す．①は ArrayList オブジェクトを生成している．②は ArrayList オブジェクトの内部の様子を示すために，蓋を開け，サイコロが入ったことを示している．③と④はさらにサイコロを追加した様子を示した．この例では，上から下に向かって順にサイコロを増やして説明した．

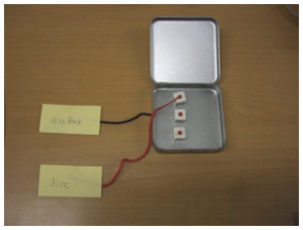
(2) get の説明

物を使って get メソッドを説明する場合，箱の中からサイコロを取り出して見せることが，受講者にイメージさせ易い．しかし，箱から取り出すことは，get メソッドでは無く remove メソッドであり，矛盾が生じる．Java ではオブジェクトは全て参照で管理されており，サイコロや箱などの物を使った説明の限界がここにある．

get メソッドで返却されるものは，サイコロ自体ではなく，その参照であることを強調することによって，ArrayList オブジェクトは，参照情報を管理していることを説明する．授業では，この参照を他の参照と区別するために，赤色の組紐を使って説明した．この様子を図 9 に示す．



Dice dice = diceBox.get(0);
誤った解釈

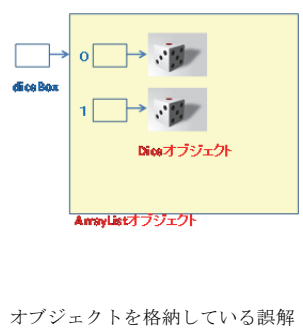


Dice dice = diceBox.get(0);
正しい解釈

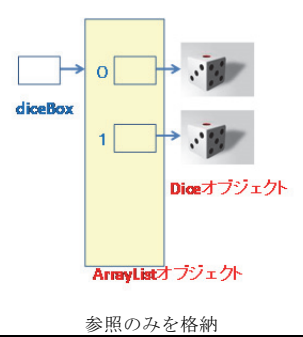
図 9 get の振舞い

(3) ArrayList オブジェクトは参照を管理

ArrayList オブジェクトは，オブジェクト自体を格納しているわけではない．この点は，蓋の付いた箱で説明すると矛盾が生じる．先に，変数への代入を取り上げたが，変数にはオブジェクトへの参照が格納されており，ArrayList オブジェクトは，この変数が沢山集まったオブジェクトであることを説明するように工夫した．授業で使用した資料を図 10 に示す．




オブジェクトを格納している誤解




参照のみを格納

図 10 ArrayList オブジェクトの様子

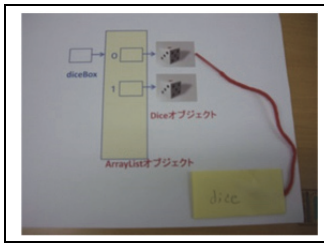
図 10 の左側の図は，ArrayList オブジェクトの中に Dice オブジェクトが存在することを示したものである．箱の例と関連付けるため，サイコロが内包されている様子を示す目的で，大きな四角でサイコロを囲んでいる．しかし，図 10 に示した右側の図のように，ArrayList オブジェクトは Dice オブジェクトの参照情報のみを管理していることを説明し，正しい理解へと導いた．

(4) サイコロを振る

ArrayList オブジェクトの中で管理しているサイコロを全て振る場合は，繰返しを使う．具体的には，ArrayList オブジェクトの中にあるサイコロを，順番に振る．これを行うプログラムの例を以下に示す．

```
for(int i = 0; i < diceBox.size(); i++){
    Dice dice = diceBox.get(i);
    dice.cast();
}
```

1 回目の繰返しで変数 dice は，diceBox.get(0)によって最初に add した Dice オブジェクトを参照できるようになる．この様子を図 11 の左側に示す．



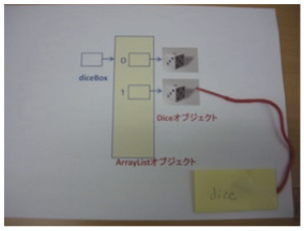




図 11 参照先が順に切り替わる様子

この状態で Dice オブジェクトの cast を実行すると，サイコロが振られたことになる．1 回目の繰り返しが終了すると 2 回目の繰り返しが始まり，変数 dice は，diceBox.get(1)によって 2 番目に add された Dice オブジェクトを参照できるようになる．この様子を図 11 の右側に示す．これらのことをサイコロの個数だけ繰り返すことによって，すべてのサイコロを振ることができる．サイコロの個数は，size メソッドで分かる．参照の概念に重点を置くことによって，組紐を順に参照先を切り替えられていることを説明した．

(5) オブジェクトの入れ子

続く授業では，ArrayList オブジェクトを活用しながら Cup オブジェクトを実装し，さらに Cup オブジェクトを活用しながら DiceGame オブジェクトへ進む．先の説明では，ArrayList オブジェクトを蓋付きの缶に見立てて物を見せてきた．この延長線上で Cup オブジェクトを考える場合，ArrayList オブジェクトを大きな円筒形の缶に入れることになる．オブジェクトの入れ子が進むと，身近な物での説明は困難になる．この様子を図 12 に示す．

図 12　オブジェクトの入れ子

入れ子になったオブジェクトの参照の関係は，あらかじめその様子を印刷した用紙を準備しておき，要点のみ付箋紙と組紐を活用しながら，説明を行った．この用紙を図 13 に示す．

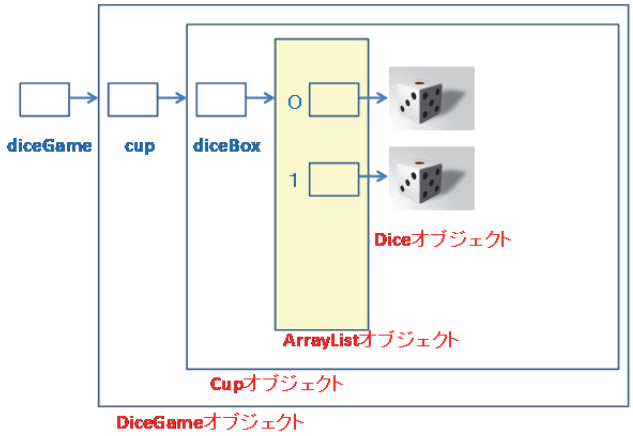



図 13　オブジェクトの複雑な関係

身近な物を活用した説明は，直感的なイメージを受講者に与えることができ効果的であるが，やがて限界が訪れる．

## 7. 教授の工夫の効果

参照に重点を置いた教授の工夫が，どの程度の効果があったのかを知るために，SIEM で得られた 2015 年と 2014 年の理解度調査結果を比較した．ただし，2015 年は現在授業が進行中であり，原稿締切の関係から 2 章と 3 章の内容を調査対象とした．

(1) 2 章の理解度の比較

2 章の理解度調査結果は，表 5 と図 14 に示した．表 5 は，左の列から順に「授業回数」「質問の番号」「質問項目」『2014 年の「はい」と回答した割合』『2015 年の「はい」と回答した割合』の順に記載した．なお，各質問項目は，紙面の都合で「は理解できましたか」の文言を省略した．「クラスとオブジェクトの違い」の理解は，時間的な推移を追跡するために，2 回目，3 回目，5 回目の授業で，番号 5，6，21 で複数回調査した．2014 年は番号 5 が 60.5%と低迷したが，2015 年は 95.0%に向上した．同様に番号 6 が 72.1%であったが，2015 年は 100.0%に向上し，理解度の顕著な向上が見られた．さらに追跡した結果，番号 21 は，2014 年と 2015 年共に 87.5%と同じ割合に収束した．

表 5 理解度調査結果(2 章の内容)

| 回数 | 番号 | 質問項目 | 2014 年 | 2015 年 |
|---|---|---|---|---|
| 2 | 1 | オブジェクトは識別可能であること | 88.4 | 100.0 |
| | 2 | オブジェクトは振舞いを持っていること | 86.0 | 95.0 |
| | 3 | オブジェクトは状態を持っていること | 83.7 | 100.0 |
| | 4 | オブジェクトの生成（インスタンス化） | 79.1 | 97.5 |
| | 5 | クラスとオブジェクトの違いは | 60.5 | 95.0 |
| 3 | 6 | クラスとオブジェクトの違いは | 72.1 | 100.0 |
| | 7 | オブジェクト（図）に状態を追加すること | 81.4 | 94.9 |
| | 8 | クラス図とソースプログラムの関係 | 69.8 | 97.4 |
| 4 | 9 | クラス図からソースプログラムを生成する方法 | 92.9 | 92.3 |
| | 10 | ソースプログラムからクラス図を作成する方法 | 92.9 | 97.6 |
| | 11 | インスタンス変数 | 76.2 | 92.7 |
| | 12 | コンストラクタ | 81.0 | 90.2 |
| | 13 | メソッド | 88.1 | 87.8 |
| 5 | 14 | インスタンス変数 | 87.5 | 87.8 |
| | 15 | コンストラクタは | 90.0 | 100.0 |
| | 16 | メソッド | 97.5 | 95.0 |
| | 17 | void | 90.0 | 100.0 |
| | 18 | private | 92.5 | 100.0 |
| | 19 | public | 90.0 | 90.0 |
| | 20 | this | 82.5 | 97.5 |
| | 21 | クラスとオブジェクトの違い | 87.5 | 87.5 |

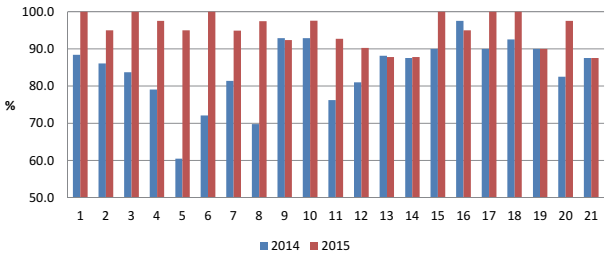



図 14　理解度調査結果の推移(2 章の内容)

図 14 の理解度のグラフからも分かるように，2015 年は理解度の立ち上がりが早いことが窺えた．2015 年の番号 6 は，一旦 100.0%になったものの，質問項目 21 では 87.5%に低下しており，少し時間が経過すると，再び分からなくなる受講者がいることも示された．

番号 8 は，クラス図とソースプログラムの関係について理解度を調査した結果である．2014 年は 69.8%であったが，2015 年は 92.4%に向上した．クラス図とソースプログラムは 1 対 1 に対応しており，クラス図が与えられるとソースプログラムを機械的に導出できる．この授業では，A4 の用紙を受講者に配布し，クラス図をフリーハンドで記述させ，これに相当するソースプログラムをクラス名，状態，振舞いの順に対応付けながら手書きさせた．もちろん，慣れて来るとクラス図を見ながら，テキストエディタへ直接ソースプログラムを機械的に導出できるようになる．

番号 11 は，インスタンス変数の理解度を調査した結果である．2014 年は 76.2%であったが，2015 年は 92.7%に向上した．粘土の型を使って生成したオブジェクトを見せながら，このオブジェクトの中にインスタンス変数が存在することを強調した効果が窺える．

(2) 3 章の理解度の比較

同様に 3 章の理解度調査結果を，表 6 と図 15 に示す．2 章の結果と異なり，2014 年と 2015 年の理解度の差が全体的に縮まっていることが特徴として窺える．7 回目の授業の理解度調査では，番号 28 の ArrayList の概念，番号 29 の add，番号 30 の get メソッドに関しては前年を上回った．

表 6 理解度調査結果(3 章の内容)

| 回数 | 番号 | 質問項目 | 2014 年 | 2015 年 |
|---|---|---|---|---|
| 6 | 22 | サイコロの状態 | 95.0 | 97.5 |
| | 23 | サイコロの振舞い | 95.0 | 97.5 |
| | 24 | 乱数の生成 | 95.0 | 97.5 |
| | 25 | this の意味 | 92.5 | 92.5 |
| 7 | 26 | サイコロ(Dice)オブジェクトを複数生成する方法 | 97.2 | 100.0 |
| | 27 | サイコロ(Dice)オブジェクトを変数に代入する理由 | 91.7 | 97.4 |
| | 28 | ArrayList の概念 | 83.3 | 92.1 |
| | 29 | ArrayList の add メソッド | 91.7 | 94.7 |
| | 30 | ArrayList の get メソッド | 91.7 | 92.1 |
| | 31 | ArrayList の size メソッド | 88.9 | 86.8 |

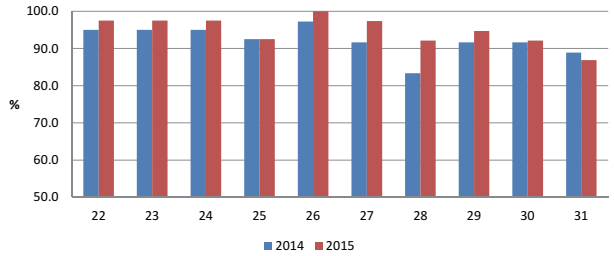



図 15 理解度調査結果の推移(3 章の内容)

番号 31 の size メソッドは，不思議な現象であるが，理解が容易であると考えられるものの，前年よりも僅かに低下する結果となった．誤差であることを願いたい．以上，参照に重点を置いた教授の工夫の効果が窺える．

(3) 受講者の反応

受講者の反応は，授業に対する要望感想から調べることにした．毎回の授業では，理解度のアンケート調査を実施しており，授業に対する要望感想は，自由記述として回答を集めている．粘土を使った授業は，2015 年 4 月 15 日(水)の授業で実施した．授業回数は，2 回目になる．この時の回答を以下に記載する．

・new だけににゅ〜
・かなり忘れてる．復習しないと
・これから頑張っていこうと思う
・すごい分かりやすいです．
・春休み中に Java と触れ合わなかったので忘れていることが多々あり思い出すまでが大変そうです．日常的にプログラミングがある生活になるといいなと思います．
・単位とれるか不安やで
・粘土が面白かったです．
・粘土の説明がとても分かりやすかったです．次はレゴブロックを使った説明をお願いします．
・粘土の表現が分かりやすかったです！
・粘土の例えとても分かりやすかったです！
・粘土を使った例えが分かりやすかったです．

粘土を使った説明は，分かり易いとの指摘が多かった．「new」と「にゅー」の語呂が良いことも，理解の一助となっていることが窺えた．

## 8. まとめ

参照に重点を置いたオブジェクト指向プログラミング入門教育の工夫を取り入れた結果，2015 年は 2014 年と比較して，理解度が向上していることが示された．特に，授業の初期の段階で理解度の顕著な向上があり，効果的である事がわかった．その後，理解度の差は縮まるものの，2015 年の大半の項目が 2014 年を上回る結果となった．粘土を活用したことにより，クラスとオブジェクトの違いを明確にできたこと，付箋紙を変数に，組紐を参照に利用したことが理解度の向上に繋がったと考えている．初学者がオブジェクト指向を学習する中で，オブジェクト指向の概念を学ぶ段階と Java 言語で実装する段階をクリアする必要がある．「コンピュータプログラミング B」の授業内容は，これらの段階を行き来しながら学習する内容となっている．さらに理解度の向上を目指すためには，こららの段階を能率よく学ぶための授業内容の工夫が今後の課題であると考えている．

本研究の一部は，科学研究費補助金（基盤研究（C）課題番号 24501214）として行っている．

## 参考文献

1) 三浦元喜，杉原太郎，オブジェクト指向言語におけるクラス定義の意味とオブジェクトの振舞いを理解するためのワークベンチ，情報処理学会，情報教育シンポジウム論文集，Vol.2011，no.4，pp.43-49，2011.

2) 大城正典，永井保夫，初等プログラミングから設計レベルまでを対象としたオブジェクト指向教育のための支援システムの提案，情報処理学会，情報教育シンポジウム論文集，Vol.2011，no.4，pp.59-66，2011.

3) 土肥紳一，宮川 治，今野紀子，SIEM によるプログラミング教育の客観的評価，情報科学技術フォーラム，情報科学技術レターズ，Vol.3，no.3，pp.347-350，2004.

4) 土肥紳一，宮川 治，今野紀子，SIEM を活用したオブジェクト指向プログラミング入門教育における教授の工夫，情報処理学会，第 77 回全国大会講演論文集(4)，pp571-572，2015.

5) 「コンピュータプログラミング B」講義ノート，http://www2.dcl.sie.dendai.ac.jp/dohi/2014/proB/