JSiSE Research Report
vol.32,no.1(2017-5)

オブジェクト指向プログラミングの利便性に着目した学習手法

竹川 夏実*，仲林 清**
*千葉工業大学大学院，**千葉工業大学

A Learning Method for Object-oriented Programming Focusing on its Advantages

Natsumi Takekawa*，Kiyoshi Nakabayashi**
*Graduate School of Chiba Institute of Technology **Chiba Institute of Technology

オブジェクト指向プログラミング（以下 OOP）はプログラムの機能拡張に優れる手法だが，苦手意識を持つ学習者が多いとされている．OOP を正しく利用するには，まず OOP の利便性を理解した上で，基礎概念と利便性の結びつきを意識することが有効ではないかと考えられる．そこで，OOP を手続き型プログラミングと比較させ，OOP の利便性を認識させる学習手法を提案した．大学 3 年生を対象に，プログラミングに関する基本知識を与えたのち，手続き型と OOP の両方で同じ課題を解かせた．その後，プログラム内で利用されていた基礎概念，および利便性について問う記述問題で，学習者の理解度を測った．実験の前後でアンケートを実施し，OOP の基礎概念や利便性に関する理解度の変化を調査した．

キーワード: オブジェクト指向プログラミング，手続き型との比較学習，利便性の理解

1. はじめに

プログラミング手法の一つに，オブジェクト指向プログラミング（以下 OOP）がある．OOP は，複雑な構造のプログラムや後に機能拡張が必要となるプログラムを記述する場合に有効とされる(1)．OOP を正しく利用するには，まず「クラス」，「継承」，「多態性」などの OOP の基礎概念を理解する必要がある．しかし，OOP の基礎概念は抽象的で理解が難しく，そのため OOP そのものに苦手意識を持つ学習者が多数存在する．

このような問題点を解決するための代表的な指導方法を以下に示す．1 つ目は OOP の基礎概念を現実社会の物体に投影させ，抽象的な基礎概念の具体的なイメージを掴ませようとする手法である．LEGO®を用いて，OOP の基礎概念を現実社会に投影しながら，学習者に理解させる学習手法が提案されている(2)．しかしこの手法は，現実社会と実際のプログラミングとの結びつきが浅いために，実際のプログラミングで OOP の基礎概念の役割を理解させることは困難である．2 つ目は，プログラムの開発環境で，プログラムの動きをリアルタイムで視覚化する手法である(3-6)．各基礎概念がプログラム中でどのような作用を及ぼしているのかを視覚化機能を利用し，学習者に確認させる．この手法では，プログラムの動作や基礎概念のふるまいを理解することはできる．しかし，なぜそのような基礎概念が必要なのか，それによってどのような利点が生じるのかを理解させることはできず，結果として OOP を利用する意義や意味を意識させることは難しい．

そこで本研究では，課題を通して学習者に OOP の利便性である「拡張性」，「保守性」を実感させ，これらが OOP の基礎概念によって生じていることを理解させる，というアプローチをとる．これによって，「クラス」，「継承」，「多態性」などの OOP の基礎概念を，人に説明ができるようになることを目的とする．OOP の利便性を実感させるために，OOP と手続き型で同じ課題を解かせて，プログラミングの手間や複雑さを比較させる．

以下，第 2 章では本研究の学習目標とそれに対する評価基準を示す．第三章では具体的な学習教材とプログラミング手法について説明する．第 4 章では実験結果を示し，第 5 章で考察を，第 6 章で今後の課題を述べる．

2. 学習目標

1章で述べたように本研究では，OOPの利便性が基礎概念によって生じているということを実感させるという学習方針を取る．ここで，学習者に理解してもらいたいOOPの基礎概念とその利便性を具体的に挙げ，それらの利点に関する学習目標を設定した．

まず，基礎概念の学習目標について説明する．本研究で着目した基礎概念は「クラス」,「継承」,「多態性」の3点である．この3点は，プログラミングの観点から説明すると,プログラムの書き換えの手間を省く「拡張性」と，それによってバグやデバッグの手間が減少する「保守性」を実現させる．次に，この3つの基礎概念の利点を，大きく2点ずつ挙げながら，具体的に説明する．

1）クラスを用いることの利点は，構造体の機能に加えメソッドを同時に持てる点と，継承や多態性の概念を実現することができる点である．前者は，データと処理をまとめてオブジェクトとして処理をすることで，データの処理が手続き型よりも容易になる利点である．後者は前者の利点から，継承や多態性の概念を実現させることができる，ということを示す．

2）継承を用いることの利点は，共通したデータやメソッドをサブクラスで引き継げる点と，オーバーライドとの組み合わせで多態性を実現できるという点である．前者は，スーパークラスで汎化したデータや処理をサブクラスで再定義せず利用できるという利点である．後者はクラスの利点に関連して，オーバーライドの概念と継承を組み合わせ多態性を実現し，開発効率を向上させるという利点である．

3）多態性を用いることの利点は，メソッドの呼び出し文の書き換えが減る，もしくは不要になる点，それに関連しメインメソッドの内の書き換えが不要になる点である．前者は，多態性を実現させる継承・オーバーライドの利用で，既存のメソッドの使いまわしが可能になるという利点である．後者は，入力データが追加されても，メインプログラムの指示の変更・追加が不要で済むという利点である．

次に，「拡張性」,「保守性」の 2 つの利便性の学習目標について説明する．ちなみに，OOP の 2 つの利便性の特徴は以下の通りである．

（ⅰ）拡張性 ・・・ 仕様追加があったときのプログラムの拡張のしやすさを示す．

（ⅱ）保守性 ・・・ 拡張性に加えてデバッグのしやすさを示す．拡張性によって生じる利点は，プログラムを拡張する際に,OOP の概念のおかげでソースの書き換えが減るということと，それによって書き換え箇所が新しく定義した箇所のみに絞られることである．保守性による利点は,OOP の概念で先に述べた拡張性が実現され，その結果バグの量が軽減することと，また書き換え箇所も限定的になるため開発者側の作業量が減ることである．

以上の基礎概念・利便性それぞれの利点を学習者が理解することを学習目標とし，まとめたものを表1に示す．

表 1 学習目標

1.クラスを用いる利点について説明できる
1-1. 構造体に対してメソッドを持つことができる 2-2. 継承ができる，多態性を実現するのに必要な念である
2．継承を用いる利点について説明できる
2-1. 共通したデータ，メソッドをサブクラスで引き継げる 2-2. オーバーライドを用いることで多態性を実現できる
3．多態性を用いる利点について説明できる
3-1. メソッドの呼び出し文の書き換えが不要,作業量が減る 3-2. メインメソッド内の書き換えがなくなる
4．拡張性について説明できる
4-1. OOP の概念で書き換えの量が減る 4-2. 書き換えが新しく定義した部分のみになり作業量が減る
5．保守性について説明できる
5-1. OOP の概念で書き換え量が減少しバグの量が軽減する 5-2. OOP の概念で書き換えを行う部分が新しく定義した部分のみになり，バグが見つかる箇所が限定的になる

3. 学習手法

3.1 基本的な方針

本研究では，学習者に手続き型と OOP でプログラムを比較させるという手法をとる.OOP の基礎概念を利用しない手続き型を比較対象にすることで，OOP の利便性を実感させやすくするというのが狙いである．

プログラミングを行うことで生じる手続き型と

OOP の差異は以下の通りである. 事前に作成したプログラムは手続き型・OOP の両手法で同一のデータ構造を取っているが，処理方法が異なっている．手続き型では，主に if 文の分岐処理でデータを分別し，計算や出力等の処理を行っていく．つまり，新しい種類のデータを追加していくほど分岐が増加し，ソースコードが複雑化していく．一方 OOP では，クラスを利用してデータの分別をし，計算・出力等の処理もそのクラス内で行うためソースコードが複雑化しない. これは，クラスをはじめとした継承・多態性などの OOP の基礎概念がプログラム中で利用されているからであり，その結果として「修正箇所が限定的になる」ことや，「バグが発生しても発見しやすくなる」という OOP の利便性が実現される.

3.2 学習課題

作成したサンプルプログラムは，ショッピングカートの実装をテーマとしたものである．このプログラムは，顧客 1 人 1 人が買い物かごを取り，好きな商品を入れ，レジで精算するという一連の処理を行うものである．本研究で取り上げた「ショッピング」とは，amazon のようなショッピングサイトを想定し，さまざまな品種の商品が購入できるものと定義している.

プログラムの機能要件は 2 点ある．1 点目は，「商品種別ごとの割引率を変えること」である．2 点目は，「割引計算を商品種別ごとのクラスのメソッドのオーバーライドで実現させること」である．この 2 点を実現する際に, OOP と手続き型とでどのような差異が出るかを学習者に確かめてもらう.

また，学習者に提示する前のプログラムで実装している商品は 1 種類のみ（本）であり，これに続き様々な種類の商品を学習者に追加させる課題を出題する．出題した課題については，次節で詳しく説明する.

3.3 学習の組み立て

表 2 で示す実験の流れに沿って，出題した課題と目的を説明する.

1）事前アンケートでは，学習者の事前の OOP と手続き型に関する理解度を測り，その理解度にどのくらい差があるのかを確認した.

2）課題 1 では，商品に関するクラス追加の問題を 1 つ出題し，主にクラスの概念に関して理解させることを目的とした．課題 1 の記述問題は，表 1 で示したようなクラスの利点に関して考えさせる問となっている. 表 2 では，継承の概念も理解させるとあるが，課題 1 の時点では体感させる程度のものとなっている.

3）課題 2-1 では，多態性の概念に関して意識させることを目的とした．学習者に行わせたことは，新たな品種の商品を 2 つ追加することである. 記述課題では，多態性の利点に関する問と，バグが入る箇所に関する問を設けた．前者の問では，一通り課題を終えた後に改めて多態性の利点を考えさせることで理解の定着を図ること，後者の問では，OOP の利便性の 1 つである保守性をここで実感してもらい，これが多態性という概念によって実現していることを意識させることが目的である.

4）課題 2-2 では今まで体感的に学習してきた継承の概念の理解を定着させること，さらに継承とかかわりのある多態性に関して再度理解度を確認することを目的としている．課題は，課題 2-1 で定義した品種をさらに細分化し，それに関する計算・出力の処理を OOP ならば継承，手続き型ならば if 文による分岐処理で実現させるというものである．記述課題では，継承に関する問を出題し，学習者がデータの汎化や，表 1 に示すような継承の利点が理解できているかを判断した.

5）事後アンケートでは，事前アンケートで調査した OOP の基礎概念の理解度を再度学習者に自己評価させ，その差異を確認した.

以上が実験の流れに沿った課題の組み立てである. 図 1 は，課題 1，2-1，2-2 を終え，拡張が完了した OOP のプログラムのクラス図である.

表 2 実験の流れと課題の目的

実験の流れ	目的
1）事前アンケート	学習者の事前の理解度を把握する
2）課題 1	クラスと継承の概念を理解させる
3）課題 2-1	多態性の概念を意識させる
4）課題 2-2	継承の利点や多態性に関する理解を定着させる
5）事後設問・事後アンケート	OOP の知識の理解度を主観的に評価させる，基礎概念と利便性の関係性について理解を定着させる.

表 3 課題で追加する処理の例

商品 / 処理	本 (最初から定義)	CD (課題で追加)
小計計算	値段×個数	値段×個数×0.92
情報表示	商品名 著者名 値段 個数	商品名 ｱｰﾃｨｽﾄ名 値段 個数

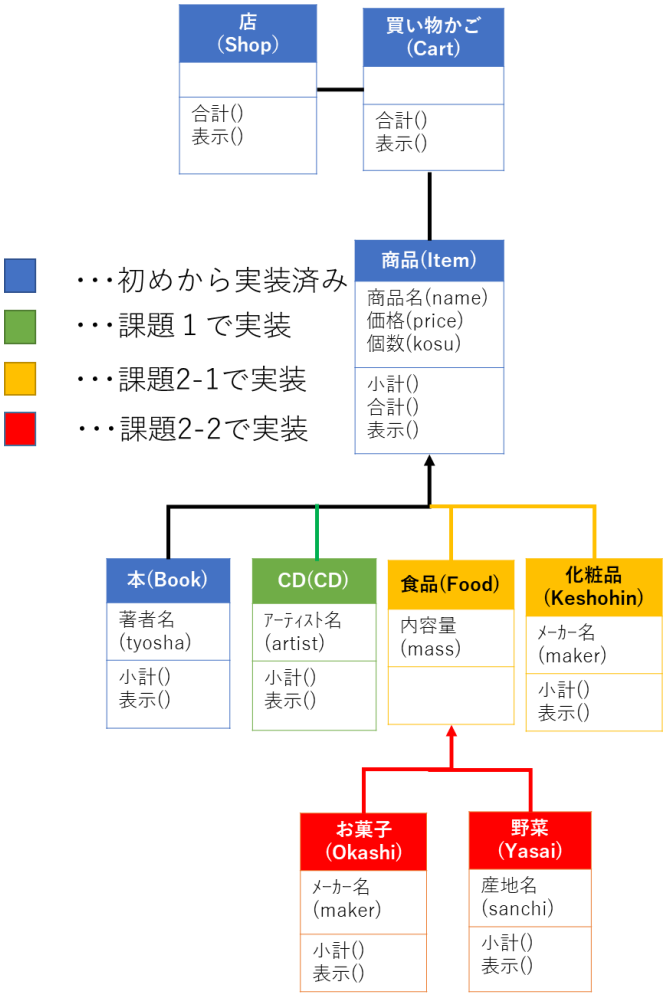

図 1　機能拡張した OOP のクラス図

4.　実験結果

4.1　実験概要

対象とした学生は，情報系学科三年生 5 名である．学習者の条件は，手続型プログラミングの経験がありOOP の経験があるが概念を詳しく理解していない学生とした．実験期間は 1 週間とし，その中で他人と相談せずに各自で課題に取り組んでもらった．使用言語は Java で，開発環境は eclipse である．

4.2　各課題の評価

この節では，各課題の評価をした結果を説明する．

4.2.1 プログラムソースに関する回答の評価

表 4 では，各課題で出題した問題のうち，プログラムソースに関する問題を抜粋している．学習者 5 人の回答に正誤評価をつけ，正解した人数を表示している．

表 2 に沿って評価の結果を説明する．プログラムソースに関する問は，課題 1，2-1，2-2 すべてにおいて共通の項目を出題した．課題 1，2-1 では，特に目立つ誤りは見られなかった．しかし課題 2-2 の「ソースの差異」の項目の正答数が比較的少なかった．ソースの差異とは，学習者が拡張した OOP，手続き型のプログラムでデータ構造にズレが生じていないかどうかを確認するための項目である．この項目を誤答した学習者のプログラムソースの特徴は，継承の概念を適用させる箇所に誤りがある（スーパークラスとするクラスを間違える），というものであった．具体的には，課題 2-2 で行う，お菓子・野菜のクラスを食品クラスに継承させる箇所である．図 2 に，学習者 A のプログラムソースの例を示しておく．学習者 A の回答では，親クラスを食品クラス food とするところを Item クラスとしており，図 1 で示したようなデータ構造をとっていない．また，図 3 は学習者 A の手続き型のプログラムソースであるが，OOP と同様の箇所に対応するところで誤りがあった．図 4 には模範解答を示しているが，お菓子・野菜の処理は食品の小計計算のメソッドの内容として呼び出すことが理想である．しかし，学習者 A の手続き型の回答では，お菓子・野菜に関する処理を，元々定義していた shoukei_food メソッドを呼び出さず，新規にメソッドを作成していた．そのため，OOP のプログラムと同様にデータ構造の誤りを起こしてしまったと考えられる．

表 4 プログラムソースに関する課題の正答者数
（学習者 5 名）

課題 / 設問	1	2-1	2-2
1.プログラムソースの実行結果	5	5	5
2.ソースの差異(OOP)	5	5	2
3.ソースの差異(手続き型)	5	5	3
4.拡張したクラス	5	4	4
5.拡張していないクラス	5	5	4

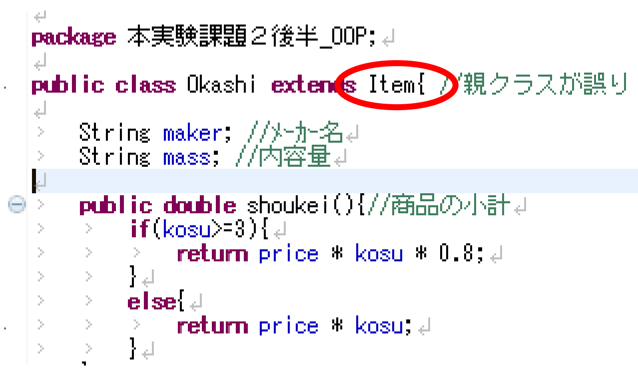

図 2 学習者 A のプログラムソース（課題 2-2OOP）

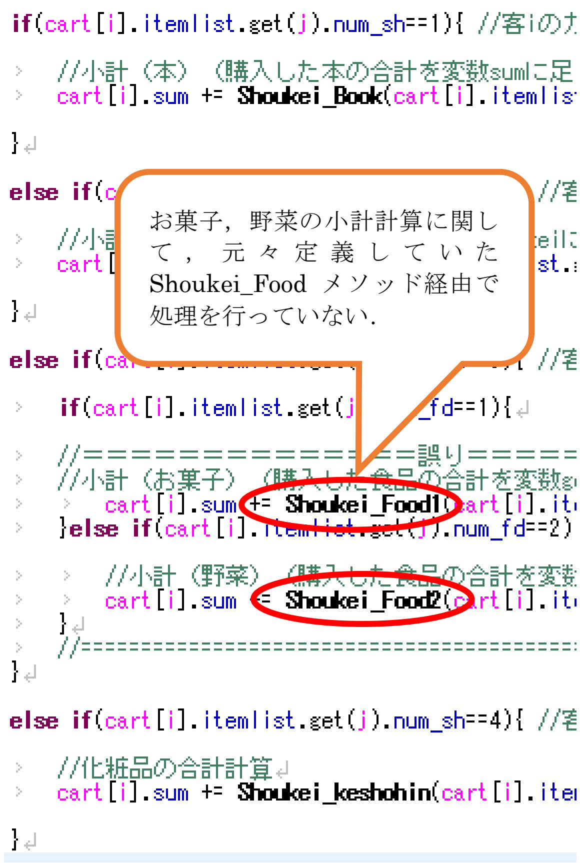

図 3 学習者 A のプログラムソース(課題 2-2 手続き型)

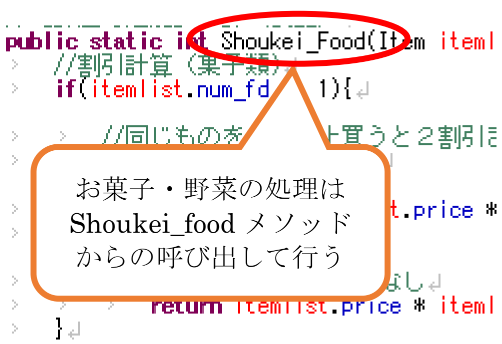

図 4 模範解答(課題 2-2 手続き型)

4.2.1 記述課題の評価

実験では各課題において，プログラミング課題で体感させた OOP の基礎概念と利便性について，学習者の理解度を確認するための記述問題を用意した．記述問題はすべて，各プログラミング課題を回答したあとに回答するように学習者に指示した．表 5 は，各項目に関する学習者の記述内容が表 1 の学習目標を満たしているかどうか，各課題で出題した項目につき 5 点満点で評価し学習者全員の平均を取ったものである．表 5 から言えることは，OOP の 2 つの利便性である拡張性・保守性は比較的高評価であるのに対し，OOP の基礎概念であるクラス・継承・多態性の評価が低いということである．

表 5 学習目標に基づいた記述問題の評価

項目＼学習者	平均（5 点満点）
クラスの利点（課題 1）	1.8
継承の利点（課題 2-2）	3
多態性の利点（課題 2-1）	3
OOP の拡張性（まとめ）	3.4
OOP の保守性（まとめ）	3.8
合計（25 点満点）	14.6

表 6 学習者の記述例

課題１：問３）クラスと構造体の違い，クラスが独自に持つ機能は何でしょう？
（学習者 A の回答）クラスは構造体と比べてクラスをたくさん追加しないといけないが、行が少ないのでプログラムの流れの把握がしやすく、機能拡張もやりやすい。クラスが独自に持つ機能はクラスの継承ができることである。
（学習者 B の回答）構造体を拡張してクラスを作成する。独自に持つ機能はオーバーライドである．
（学習者 C の回答）構造体はデータのみを持っていて、クラスはデータとメソッドを持っている。クラスが独自に持つ機能はインスタンス化である。
（学習者 D の回答）構造体はそれぞれを定義しないといけないが、クラスは一度定義するとほかのクラスでも使うことができる。
（学習者 E の回答）クラスと構造体の違いは、クラスで判別できる点と構造体の中身を見て判別しなくてはならない点である。クラスが独自に持つ機能はオーバーライドである。

4.2.2 記述課題の評価

記述課題の評価の例を説明する．表 1 で示した学習目標を達成するため，クラスに関する記述問題は，構造体との違いを聞く問と，クラスの独自の機能を聞く問を用意した．表 6 に，学習者全員のこの記述問題の回答例を示す．クラスと構造体の違いについて正しく回答していたのは学習者 C のみである．そのほかの学

習者の回答では，「データとメソッドの両方が持てる」という趣旨の記述がみられないと判断し，不正解とした．クラス独自の機能に関して正しく指摘できていたのは，学習者 A である．一方で，学習者 B，E は独自の機能をオーバーライドと回答していた．これは厳密には誤りではないが，オーバーライドの概念単体から利便性が生まれるものではないと判断し，不正解とした．

4.3 実験前後の学習者の理解度の変化

表 7 は，実験の事前事後でアンケートを実施し，学習者の理解度の平均をまとめたものである．抜粋した項目は，主に本研究で重視しているクラス・継承・多態性の 3 つの基礎概念とそれに関与する基礎概念である．表 7 からは，継承以外の項目に関して，事前よりも事後の方が学習者全体の理解度が高い．また，表 8 では，事後アンケート内で，手続き型との比較が有意義であったかどうかを学習者に 5 段階評価させた結果を示している．この調査ではほとんどの学習者が 4 点以上の高評価をしている．

表 7 実験前後の学習者の理解度の変化（５件法）

学習者平均 / アンケート項目	事前平均	事後平均
クラス	3.40	3.60
インスタンス	3.40	3.80
メソッド	3.40	3.80
継承	3.00	3.00
スーパークラス･サブクラス	2.80	3.20
オーバーライド	2.00	2.80
多態性	1.60	2.40

表 8 比較に関するアンケート

手続き型との比較	平均
比較することで理解は深まったか	4.2
手続き型との違いは理解できたか	4.4

5. 考察

5.1 OOP の利便性と基礎概念の結びつきの理解度

本研究では，OOP と手続き型でプログラムを比較させる学習手法を提案し，OOP に対する学習者の理解度の改善を試みた．表 8 が示すように，この手法は，OOP について理解を深めるために有効な手法であることがわかった．また，表 5 が示す記述の評価により，今回実施した実験では，OOP の利便性に関しては学習者の理解度を改善することができたと考えられる．しかし，基礎概念の項目に関しては，理解度の改善が大きく見られなかった．今回提示した課題では，OOP の利便性と基礎概念の関係性を学習者に意識させることが困難であったことがわかった．

次節からは，OOP の利便性と各基礎概念の学習者の理解度に関する個別の考察を行う．

5.2 「拡張性」「保守性」の理解度について

OOP の「拡張性」,「保守性」という利便性の理解度に関して考察する．

表 5 より，実験後の学習者の理解度の平均が比較的高かったのは，保守性の項目である．保守性に関しては，学習者の中で，表 1 で示した保守性についての指摘を含んだ記述が十分見られた．

拡張性に関しては，学習者 C 以外の回答に部分点をつけ，学習が不十分であるとみなした．これは，表 1 で示した,「書き換えが新しく定義した部分のみになり作業量が減る」という評価基準は学習者のほぼ全員が満たしていたが,「OOP の概念で書き換えの量が減る」ということに関しての指摘をした学習者が少なかったためである．この原因として，OOP の基礎概念と利便性を，課題の中で関連付けさせることが十分にできなかったという点と，基礎概念の利点を十分に理解させることができなかったという点が挙げられる．

5.3 「クラス」の理解度について

表 1 より，クラスに関する学習目標は「構造体に対してメソッドを持つことができる」,「継承ができる，多態性を実現するのに必要な概念である」の 2 点である．この 2 点に関してはどちらも正答率が半分以下となっている．前者の学習目標が満たされていた学習者は 5 人中 2 名であった．この 2 名が正解した理由については，事前に配布した課題用資料(課題のテキスト)内で「クラスの概念は構造体がデータのみを持つのに対して、データに加えて処理（メソッド）を持つ程度

の認識で構わない」と説明した箇所を見てから，課題に取り組んだことで理解につながったのではないかと考えている．不正解であった残り 3 名については，課題用資料の説明に目を通さず課題を行ったか，あるいは目を通していたがクラスと構造体のイメージが漠然としたままであったということが考えられる．

また，後者の学習目標を満たした学習者はおらず，全員が不正解となっていた．課題であったプログラムソースの拡張では，クラスの利点が「継承ができる」，「多態性が実現できる」ことであるということを体感的に学習させることはできても，説明することができるレベルの理解にはつながらなかった．この原因として，課題で追加する新規クラスを，すでに作成してあるクラスをコピー&ペーストし，それをもとに作成しているという点が挙げられる．課題で拡張したクラスはすべてクラス内の構成が同一のものであり，コピペしたものの中でクラス名やメソッド内の処理，定義するデータ等を書き直せばよいとしていた．このため，作業量がもともと少なかったということもあり，クラスと継承，多態性の関係性について強く印象を付けることができなかったと考えている．

5.4 「継承」の理解度について

表 1 より，継承に関する学習目標は「共通したデータ，メソッドをサブクラスで引き継げる」，「オーバーライドを用いることで多態性を実現できる」の 2 点である．この 2 点のうち，学習者全員が理解していたのは前者の利点であり，後者の利点に関して気づきを得ていた学習者はいなかった．前者の利点を満たしていた理由として考えられることは，すべての課題を通して，継承のクラスの引き継ぎの機能に関して学習する機会が多かったということである．また，課題 2-2 では継承に関する記述問題を出題したため，一通り学習をこなしたあとに復習する形で理解を定着できたのではないかと考えられる．

5.5 「多態性」の理解度について

表 1 より，多態性に関する学習目標は，「メソッドの呼び出し文の書き換えが不要,作業量が減る」,「メインメソッド内の書き換えがなくなる」の 2 点である．前者の利点に関してはほぼ全員が指摘をしていたが，後者の利点に関して気づきを得ていた学習者は 5 人中 2 名であった．前者の利点の指摘が多かった理由は，以下の通りである．学習者には，事前に配布した多態性に関する資料を，課題に取り組んでもらう前に読んでもらった．そのため，資料に記載した多態性の特徴を学習者がうまくイメージできたのではないかと考えている．また，後者の利点の指摘が少なかった理由は以下のとおりである．メインメソッドに触れることなくプログラムを拡張するという点は，全てのプログラミング課題に共通な点である．しかし，今回の場合は，学習者に拡張させる箇所を強く意識させたことで，拡張していない箇所について意識させることが困難になっていた.そのため,「メインメソッドに触れていない」という気づきを得られにくくなっていたと考えている．

6. 今後の課題

考察をふまえ，今後の課題としては，OOP の各基礎概念を利用しているという意識を持たせるため，より各基礎概念にバイアスをかけた課題を考慮することが必要なのではないかと考えている．また，そのほかにも，コピー&ペーストで拡張を済ませないような課題の難易度調整と課題のテーマの再設定，そして，学習者の回答の採点の精度を上げるための，評価基準と課題内容の明確な対応付けなどが挙げられる．

参 考 文 献

(1) 立山秀利，: “Java のオブジェクト指向がゼッタイにわかる本”,秀和システム （2006）

(2) 中鉢直宏，伊藤一成 : “オブジェクト指向プログラミング教育における LEGO を用いた体験型課題の試み，情報処理学会研究報告，Vol.2014-CE-124，No.8，pp.1-6（2014）

(3) 大城正典，永井保夫 : “プログラミング初学者を対象としたオブジェクト指向プログラミング教育システムの提案 ―オブジェクト指向の基本概念の理解に基づいたプログラムの作成・実行支援機能を中心として―”，情報教育シンポジウム 2016 論文集，pp.114-121 (2016)

(4) 三浦元喜，杉原太郎，國藤進 : “オブジェクト指向言語における変数とデータの関係を理解するためのワークベンチ”, 情報処理学会誌, Vol.50, No.10, pp.2396-2408（2009）

(5) 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”，情報教育シンポジウム，pp.43-49(2011)
(6) 石川裕季子，松澤芳昭，酒井三四郎：”オブジェクト指向言語におけるポリモーフィズムの概念を理解するためのワークベンチ”, 教育システム情報学会誌, Vol.31, No.2，pp.208-213（2014）