NAIST–IS–DT9961017

博士論文

ソフトウェア技術者の教育支援体制と プログラミング学習プロセスに関する研究

武村 泰宏

2002 年 2 月 12 日

奈良先端科学技術大学院大学
情報科学研究科 情報システム学専攻

本論文は奈良先端科学技術大学院大学情報科学研究科に
博士(工学)授与の要件として提出した博士論文である。

武村 泰宏

審査委員： 松本 健一 教授
渡邉 勝正 教授
井上 克郎 教授（大阪大学）
関 浩之 教授

ソフトウェア技術者の教育支援体制と プログラミング学習プロセスに関する研究*

武村 泰宏

内容梗概

本論文は，ソフトウェア技術者の派遣労働における組織的な管理手法の整備と，技術的な管理手法を確立するためのプログラミング教育の体系化を目的としている．組織的な管理手法はソフトウェア技術者の継続的に安定した確保を実現し，プログラミング教育の体系化はソフトウェア技術者が開発環境に関する未修得の知識・技術を系統的に把握して効率的に習得することを可能にする．近年の情報通信技術の進歩にともない，多くの企業では情報システムの開発・運用・保守のためのソフトウェア技術者の確保が急務である．一方，企業は社員数の増加を避け経営の効率化を目指しているが，ソフトウェア技術者を確保し雇用するには定常的なコストが必要になる．このような相反する課題を解消する一つの手段として派遣労働の需要が高まっている．

本論文では，ソフトウェア技術者を確保するための組織的な管理手法の一つとして，ソフトウェア技術者の労働者派遣事業における教育支援体制を提案する．そして，開発環境の一つとして Java プログラミング言語に注目し，学習過程の進行による理解状態の解析と，Java 言語における理解順序を知識間の順序関係（ある知識の理解に他の知識の理解を必要とする関係）によって解明する．

ソフトウェア技術者の教育支援体制では，情報システムにおける労働力需給の経年的変化と派遣労働の形態を分析し，ソフトウェア技術者の派遣に適応した労働者派遣事業を取り上げる．そして派遣元企業が，派遣先企業からの人的数量の需要に対応するための組織的な管理手法として，従来の労働者派遣事業の形態に長期，短期的な需要予測，関連業務の兼業，教育機関との連携が統合された教育支援体制をソフトウェア工学教育の観点から提案する．

Java 言語の学習過程の進行による理解状態の解析では, 学習者の理解状態に応じた教育

*奈良先端科学技術大学院大学 情報科学研究科 情報システム学専攻 博士論文, NAIST-IS-DT9961017,2002 年 2 月 12 日.

形態を形成するため，学習過程における因子の遷移を因子負荷量の変化から解析する手法を提案する．Java 言語のそれぞれの学習過程（学習段階①：学習の前，学習段階②：基礎的な知識の学習後，学習段階③：プログラミング演習の後）において実験を行い，テスト結果の因子分析で抽出した因子の遷移から学習者の理解状態を解明する．その結果，プログラミング言語固有の概念がプログラミング演習によって理解されることを解明し，プログラミング演習によってプログラムの理解状態に変化が生じることを確認した．

Java 言語における理解順序の解析では，Java 言語の体系的な教育を実現させるための一方法として，知識間の順序関係を導出する統計的分析手法を提案する．統計的分析手法は，P.W.Airasian らの Ordering theory（順序理論）のモデルを適用し，テストで生じるいわゆる「まぐれ当たり」と「うっかりミス」による回答を，二項分布の分布関数を用いて統計的に補正して知識間の順序関係を導出する．また，順序関係の全体構造を表した関連構造から理解順序を解析するため，知識項目だけの複雑な関連構造を因子分析の適用によって簡単化する手法も提案する．

次に Java 言語の効率的な学習プロセスを導出するため，講義・演習とテストによる実験を行い，実験結果に統計的分析手法を適用して Java 言語の順序関係を導出する．Java 言語の理解順序は，順序関係の全体構造を簡単化して表した関連構造から解析する．実験結果の分析では，25 の知識項目から 104 の順序関係を導出し，その知識項目を 9 つのクラスタに集約した関連構造から Java 言語の理解順序を解明した. 関連構造から Java 言語の効率的な学習プロセスを導出し，学習プロセスと Java 言語が解説された著書の目次との違いを考察した．

以上，ソフトウェア技術者の安定供給を労働者派遣事業において可能にする教育支援体制を提案すると共に, Java 言語における理解状態の変化と理解順序を実験と提案手法によって解明した．本研究の成果は，情報通信技術の進歩に対応したソフトウェア技術者の派遣労働と，ソフトウェア工学教育の体系化に貢献すると考えられる．

キーワード

ソフトウェア技術者，派遣労働，技術教育，理解状態，順序関係，関連構造，理解順序，Java 言語

A Framework for Supporting Software Engineer Education and an Effective Process for Learning Programming*

Yasuhiro Takemura

Abstract

The purposes of this paper are to establish a systematic management method for securing enough software engineers stably and continuously under the employment system of dispatched workers, and to systematize the programming education process so that software engineers can achieve efficient learning, which enable them to systematically acquire knowledge and techniques concerning development environments. In today's situation where the technologies for information communication are in progress, it is an urgent task for a company to secure enough software engineers for the development, operation, and maintenance of a computer system. On the other hand, the company has to avoid increase in numbers of employees to aim the rationalization of management. However, employment of a full-time software engineer requires the steady cost. The demand of dispatched workers is increasing as one solution for such an inconsistent problem.

This paper proposes a framework for supporting software engineer education as one of the systematic management methods to secure software engineers who may satisfy the required technical level from a company in the system of dispatched workers. This paper also analyzes the change of the comprehension levels in the process of learning phases and the ordering relation (relation which needs to understand other knowledge for understanding one knowledge), in recent trend of Java programming language as a development environment.

In order to establish a framework for supporting software engineer education, the author firstly analyzes the tendency of supply and demand of dispatched workers and the forms of dispatched workers. Then proposes a supporting framework for software engineers from the viewpoint of software engineering education. The framework incorporates practical solutions in the conventional form of the system of dispatched workers: forecasting the needs of

* Doctor's Thesis, Department of Information Systems, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-DT-9961017, February 12, 2002

dispatched workers in the near and far future, offering the additional job to support excess personnel, and engineer education by cooperation with educational organizations. This framework is based on the characteristics of a software engineer's learning, which requires a lot of time and cost to adjust to the change of development environment.

In the analysis of evolving engineer's comprehension levels followed by the change of a learning phase, the author proposes a method for observing the changes of factors between learning phases based on factor loadings. In the experiment, subjects took a test in each learning phases ((1) before learning Java language, (2) after learning Java language using a textbook, (3) after a programming exercise using a computer). The subject's comprehension levels were clarified by illustrating the changes of factors extracted from the result of the test. Consequently, the author found that the concept peculiar to a programming language was understood after the programming exercise, which was not understood during the learning phase using a textbook.

In the analysis of the ordering relation, the author proposes a statistical analysis method employing Ordering Theory proposed by P. W. Airasian et al., then extracts the ordering relation between knowledge items of Java language based on the method. The method statistically rectifies the ill-suited relations produced by "a freak of chance" and "careless mistake" of subjects. Moreover, the author also proposes a method for simplifying the ordering relations to represent a sophisticated structure of whole ordering relations. In order to evaluate the proposed method, the author conducted an experiment in which subjects took a lecture, an exercise, and a test. The understanding order of Java language and an efficient learning process was derived by the propose method. In the analysis of experiment result, 104 ordering relations was extracted from 25 knowledge items, and the relational structure was constructed, where the knowledge items are integrated into 9 clusters. The efficient learning process of Java language was extracted from the relational structure, and the difference between the learning process and the table of contents in the books of Java language was compared.

The result of this research will contribute to the establishment of an efficient employment system of dispatched workers and to the systematization of software engineering education, which are essential parts in the progress of the information technologies.

Keywords：

software engineers, dispatched work, technical education, comprehension levels, ordering relation, relational structure, understanding order, Java programming language

関連発表論文

1. 武村泰宏，下左近多喜男: コンピュータ技術者の技術教育と派遣労働について，日本経営システム学会論文誌，Vol.12, No.1, pp.1-7, 1995.6.
2. Yasuhiro Takemura and Haruhisa Yamagichi: An evaluation of learning effect for education form by means of computer communication with foreign country, *Proceedings of the International Conference on Technology Education in School around Asian Countries*, pp.165-168, Sep. 1995.
3. Yasuhiro Takemura and Haruhisa Yamagichi: An evaluation of learning effect for education form by means of applying both international personal computer communication and translation system, *Proceedings of the International Conference on Technology Education in School around Asian Countries*, pp.169-172, Sep. 1995.
4. Haruhisa Yamagichi and Yasuhiro Takemura: Learning result evaluation by fuzzy learning model, *Proceedings of the International Conference on Technology Education in School around Asian Countries*, pp.181-184, Sep. 1995.
5. Yasuhiro Takemura and Takio Shimosakon: A study on home use information system by multimedia, *Proceedings of the 14th International Conference on Production Research*, Vol.2, pp.1344-1347, Aug. 1997.
6. 武村泰宏，新井基祐，工藤英男: 報処理教育における学習者の意識調査，平成 11 年度情報処理教育研究集会講演論文集，pp.213-216, 1999.11.
7. 武村泰宏，新井基祐: マルチメディアを活用した新語学教育システムの学習効果，全国大学・短期大学実務教育協会，平成 11 年 3 月実務教育研究年報，第 5 号, pp.91-97, 1999.11.
8. Yasuhiro Takemura, Kazuyuki Shima, Ken-ichi Matsumoto, Katsuro Inoue and Koji Torii: Factor analysis of comprehension states in the learning phases of a programming language, *Proceedings of the 6th Asia-Pacific Software Engineering Conference (APSEC'99)*, pp.136-143, Dec. 1999.
9. 寺井淳裕，内田眞司，島 和之，武村泰宏，松本健一，井上克郎，鳥居宏次: ソースコードの並び替えによるソフトウェアの問題発見手法，電子情報通信学会技術研究報告，ソフトウェアサイエンス研究会，SS2000-52, Vol.100, No.570, pp.81-88, 2001.1.
10. 武村泰宏，島 和之，松本健一，井上克郎: Java 言語の理解順序に関する関連構造の統計的分析，電子情報通信学会技術研究報告，ソフトウェアサイエンス研究会，SS2000-53, Vol.100, No.677, pp.1-9, 2001.3.

11. 武村泰宏，内田眞司，工藤英男: インターネットにおける知識と倫理意識の関連，教育システム情報学会第 26 回全国大会論文集，pp.155-156, 2001.8.
12. 内田眞司，寺井淳裕，武村泰宏，阪井 誠，島 和之，松本健一: ソフトウェアの理解性向上によるデバッグ時間の短縮，情報処理学会第 63 回全国大会講演論文集，6H-05,pp.1-101-102, 2001.9.
13. Yasuhiro Takemura, Kazuyuki Shima, Ken-ichi Matsumoto, Katsuro Inoue, and Koji Torii: Factor analysis of concepts in learning a programming language, Submitted to *International Journal of Computers & Education*.

目 次

図 目 次

表 目 次

1. はじめに

1970 年代初頭から始まった情報通信技術の導入による Office Automation や Factory Automation は[11],[31], 労働者に求められる知識・技能の質を変化させ, 企業における雇用の質的な変化をもたらしながら就業形態に大きなインパクトを与えてきた[12]. 情報通信技術の産業分野における活用は, 生産, 販売, 流通等の企業活動を効率化し, 単純な作業を情報システムに置き換え, 情報通信技術に関する職種を拡張させている. また情報通信技術の活用は, 専門的・技術的な処理の標準化を可能にして業務委託や派遣労働を容易にしている. 企業が産業構造の変化に対応しながら先進的な情報通信技術の導入やそのための人材育成を実現し, その導入効果を十分に発揮するためには, 情報システムの開発・運用・保守を担当するソフトウェア技術者の確保が急務である. 一方, 多くの企業では社員数の増加を避け経営の合理化を目指しているが, ソフトウェア技術者を確保し雇用するには給与, 福利厚生費, 教育費などの定常的なコストが必要になる. このような相反する課題を解消する一つの手段として派遣労働の需要が高まっている.

本論文では, ソフトウェア技術者を労働者派遣事業において安定して継続的に確保するための組織的な管理手法の整備と, ソフトウェア技術者が開発環境に関する未修得の知識・技術を系統的に把握して効率的に習得するためのプログラミング教育の体系化を目的としている.

情報システムの導入にともない企業ではより高度な知識・技術を有するソフトウェア技術者が必要になる一方, 雇用コストの削減にも配慮しなければならない. このような労働環境や経営環境の変化の結果として, 終身雇用制, 年功序列の崩壊を招きつつ派遣労働が注目されている. 労働者派遣法は昭和 60 年法律第 88 号として制定されて改定を重ねながら現在に至り, その規模を拡大させている. 例えば平成 8 年度から 11 年度における派遣労働の規模は, 年間売上高 1 兆 1,827 億円が 1 兆 4,605 億円, 派遣先件数 221,885 件が 264,439 件, 派遣された派遣労働者数 724,248 人が 1,067,949 人へと拡大している[14]. 米国においても業務委託が進んだ結果, 業務の受け皿となる企業向けサービス業の雇用者数が 1980 年の 2,564 万人から 2000 年の 9,570 万人へと 3.7 倍に拡大している[12].

このように柔軟性を増した労働環境の下では，ソフトウェア技術者は多様な就業形態の選択と就業機会の拡大を図れるようになるが，そのための知識・技能の向上と習得といった課題に直面する．派遣労働という制度の下において，ソフトウェア技術者が未習得の開発環境を割り当てられれば，ソフトウェア技術者は短期間に派遣先企業の開発環境に関する知識・技術の学習に迫られる． 開発環境の変更にともなう知識・技術の学習において，ソフトウェア技術者は初心者としての特徴と熟練者の特徴を併せ持っている．つまり，ソフトウェア技術者は，開発環境固有の知識については初心者と同様に基礎的な項目すら習得していない一方，開発一般に関しては熟練者として基礎から応用まで幅広い知識を習得している．よって，ソフトウェア技術者が効率的に学習を進めるためには，既に習得済みの知識に応じて系統的に未修得の知識を学習する必要がある．

本論文では，ソフトウェア技術者が派遣労働において効率的な学習を実現できる教育支援体制を提案する．そして開発環境の一つとして Java プログラミング言語に注目し，学習過程の進行による理解状態の変化と，Java 言語における理解順序を知識間の順序関係（ある知識の理解に他の知識の理解を必要とする関係[1]）によって解明する．

算数や理科などの教科では，知識間に順序関係が存在する場合が多く，このような教科では知識間の順序関係に基づく教育手順の体系化がなされている．プログラミング言語においても，学習過程の進行にともなう理解状態の変化や，知識間の順序関係が解明されれば，プログラミング言語の体系的な教育手順が形成され，ソフトウェア技術者への効率的な教育が期待できる．このような背景の下，学習者の理解や教授の方法などをモデル化した知的教育システムの開発が進められている[38].[39],[40],[41]．知的教育システムは，教材モジュール／学習者モデルモジュール／教授戦略モジュール／インタフェースで構成される．学習者モデルは学習者の理解状態を柔軟に把握し、その状態に適応した適切な教授行動をとるための不可欠な情報を提供している[20]．学習者モデルの構築においては，学習者の理解状態や思考過程などの把握が重要になる．学習者の思考，学習，知識獲得といった過程のモデル化では，(1) 知識表現（プロダクション・ルール／フレーム）[47]，(2) 算数の文書題における因果タイプ（単体変化問題／相対変化問

題／比較問題／部分全体問題／時間問題／空間問題）[10], (3) 振舞いに基づくプログラミングモデル[19]などが提案されている．また，プログラミング言語の学習に関する研究や[3],[15],[18],[29],[35],[42],[46],[54],[55], 学習者の理解度評価の解析もさまざまなアプローチで進められている[44],[45].

本論文では，まず第 2 章において，情報システムの労働力需給の経年的変化と派遣労働の形態を分析し，ソフトウェア技術者の派遣に適した労働者派遣事業を取り上げる．そして派遣元が，派遣先からの人的数量の需要に対応するための組織的な管理手法として，長期・短期的な需要予測，関連業務の兼業，教育機関との連携が統合された教育支援体制をソフトウェア工学教育の観点から提案する．

第 3 章では，学習者の理解状態の変化に適応した教育形態を構築するため，学習過程の進行にともなう理解状態の変化を解析する[42]．プログラミング言語の関連知識をもとにして実際にプログラミング演習を行った後では，理解状態に変化が生じることは経験的に知られている．このような学習過程の進行にともなう理解状態の変化に注目し， Java 言語の 3 つの学習過程（①学習の前，②基礎的な知識の学習後，③プログラミング演習の後）においてテストを行い，学習者の理解状態の変化をテスト結果の因子分析で抽出した因子の変化によって解明する．

第 4 章では，プログラミング言語の効率的な学習と体系的な教育を実現させるための一方法として, Java 言語の知識間の順序関係を導出するための統計的分析手法を提案する[43]．統計的分析手法は，P.W.Airasian らの順序理論（Ordering theory）のモデルを適用して知識間の順序関係を導出する．順序理論は知識間の順序関係を導出する方法の一つで，多数の学習者に対してテストを行い，任意の 2 つのテスト項目における回答結果を分析して順序関係を導出する[1]．学習者がテストに回答する際，いわゆる「まぐれ当たり」や「うっかりミス」は止むを得ない事象であり，テスト結果の分析においてこれらの事象を考慮しなければ正しい順序関係が得られない．本研究では，テストで生じるいわゆる「まぐれ当たり」と「うっかりミス」の回答を二項分布の分布関数によって統計的に補正し，順序関係が導出できる統計的分析手法を提案する．また，順序関係の全体構造が表された関連構造から理解順序を把握するため，因子分析の適用で関連構造を簡単化する手法も提案する．

第 5 章では，Java 言語の効率的な学習プロセスを導出するため，講義・演習とテストによる実験を行い，前章の提案手法を用いて Java 言語の理解順序を導出する． Java 言語の知識間の順序関係を統計的分析手法によって導出し，その全体構造が簡単化された関連構造から理解順序を解析する[43]．実験結果の解析では 25 の知識項目から 104 の順序関係を導出し，その知識項目が 9 つのクラスタに集約された関連構造を構築して Java 言語の理解順序を解明する．関連構造から 3 つの学習プロセスを設定し, 効率的な Java 言語の学習プロセスを導出する．また，導出した学習プロセスと Java 言語が解説された著書[24],[27],[28],[36]の目次との違いを考察する．

最後に第 6 章で，本論文の全体のまとめと考察を行う．

２．ソフトウェア技術者の教育支援体制

2.1　あらまし

企業が産業構造の変化に対応しながら、先進的な情報通信技術の導入やそのための人材育成を実現し，その導入効果を十分に発揮するためには，情報システムの開発・運用・保守のためのソフトウェア技術者の確保が急務となっており，人材確保の手段として派遣労働の需要が高まっている[37].

ソフトウェア技術者の派遣労働では，その技術力の影響が大きいことから，ソフトウェアに関する技術教育を高度にかつ広範囲な分野について行える方法を確立させることが重要である．よって情報システムの開発・運用・保守において，ソフトウェア技術者の継続的で安定した派遣体制を確立するには，ソフトウェア技術者の派遣労働における教育支援体制の整備が必要である．既に 1992 年の産業構造審議会情報産業部会情報化人材対策小委員会中間報告の「情報化人材育成策の基本的方向」でも，企業における人材育成の指針となる標準カリキュラムの策定・公表，普及を必要とした，ソフトウェア技術者への教育の重要性が報告されている[33].

また，情報通信技術の進歩と労働環境の変化により従来の雇用方式では対処できない状況が生じ，派遣労働の制度が注目されている．労働環境に変化をもたらす要因として次のような背景が挙げられる.① 高齢化の進行による退職者の再雇用が行われている.② 出生率の低下による若年労働力の減少傾向が進行している.合計特殊出生率は 1989 年から 1990 年にかけて 1.6 を割り, 1992 年の 18 才人口の 205 万人をピークにその後は激減している[48]. 生産力人口は 1995 年の 8,717 万人をピークに，その後激減し，日本経済はこれまでに経験したことのない慢性的かつ構造的な労働力不足へと突入し，なかでも看護婦，ソフトウェア技術者の不足が予想されている[23]. ③ 海外から流入する労働力の急増により，外国人労働者問題と労働市場の国際化が進行している[22]. ④ コンピュータ技術を中心とする新たな技術革新が広がり，専門的な技術を必要とする分野が急増している．とりわけ情報システムの開発・運用・保守を担当するソフトウェア技術者の不足

が問題となっており，派遣労働という形で労働形態に大きな変化をもたらしている．⑤ 競合他社との技術競争による専門技術の向上にともない，社員の専門技術教育を充実させるか，または専門技術者の派遣を求めることが必要になる．⑥ Office Automation の進展にともなう業務の二極分化によって，業務内容が高度化，専門化または単純化が進行している．⑦ 自分の希望する日時にあわせて，専門的知識・技術・経験を活して，職場のわずらわしい人間関係にとらわれずに就業することを希望する労働者が増加している [8]. ⑧ 多くの企業が社員数の増加を避けて経営を合理化するため，ソフトウェア技術者の派遣を継続して受け入れている．

本章では労働力需給の変化と派遣労働の形態を分析し，ソフトウェア技術者の派遣に適した労働者派遣事業を取り上げる．派遣元が，派遣先からの人的数量の需要に対応するための組織的な管理手法として，長期・短期的な需要予測，関連業務の兼業，教育機関との連携を統合した教育支援体制をソフトウェア工学教育の観点から提案する．

2.2 情報サービス産業の雇用状況

コンピュータの納入状況は，情報化白書 1990 による汎用コンピュータの納入推移[26]によると，1980 年代後半より伸び率が増大し，1987 年 9 月の実働状況は 34 万台を超えて全額で 9 兆 4679 億円となっている．これはコンピュータがあらゆる産業の基幹をなしていることを意味しており，それによるソフトウェア技術者の必要性が増幅されることを示している．情報通信技術の革新によるコンピュータの利用分野の拡大により情報サービス産業が発展し，ソフトウェア技術者を中心とした労働力の需給ギャップが予測されている．1992 年の産業構造審議会情報産業部会情報化人材対策小委員会中間報告の「情報革命の担い手として求められる情報化人材」では，2000 年でのソフトウェア技術者の需要数がシステムアナリスト 32 万人，プロジェクトマネージャ 19 万人，アプリケーションエンジニア 62 万人，プロダクションエンジニア 46 万人，テクニカルスペシャリスト 9 万人，システム運用管理エンジニア 7 万人，教育エンジニア 5 万人，ディベロップ

表1　情報サービス産業売上高推移　　（単位：億円）

	1988年	1989年	1990年	1991年	1992年
売上高	32,973	43,514	58,726	70,396	71,276
前年比	143.4%	132.0%	135.0%	119.9%	101.3%

（出所：特定サービス産業実態調査）

表2　新卒採用計画および計画達成率　　（単位：人）

	1989年	1990年	1991年	1992年	1993年
採用計画数	7,774	9,285	10,215	11,205	8,840
採用実績数	6,272	7,551	8,716	11,881	8,154
計画達成率	80.7%	81.3%	85.3%	106.0%	92.2%

（情報サービス産業協会「情報サービス産業白書」より作成）

表3　中途採用計画および計画達成率　　（単位：人）

	1989年	1990年	1991年	1992年	1993年
採用計画数	2,317	2,686	2,948	1,242	944
採用実績数	1,908	2,365	2,174	1,412	664
計画達成率	82.3%	88.0%	73.7%	113.7%	70.3%

（情報サービス産業協会「情報サービス産業白書」より作成）

メントエンジニア 13 万人で，合計 194 万人程度の需要に対し，供給者数が全体で 140 万人程度となり 54 万人のソフトウェア技術者が不足すると予測されている[33]．また，1987 年の同報告によれば 97 万人のソフトウェア技術者が不足すると予測されていたが[32],[49]，現状は需給ギャップが解消されてきている．このような状況に至った要因は，CASE などの開発環境の充実と，ソフトウェア工学の各種手法を適用したソフトウェア開発技術の進展によるものと考えられる．

特定サービス産業実態調査の「情報サービス産業売上高推移」[50]によると，88 年からの伸び率は景気後退の影響により減少傾向にあるが，売上高は 92 年が 7 兆 1276 億円で 88 年売上高の 2.16 倍と増加している．情報サービス産業売上高推移を表 1 に示す．

情報サービス産業協会（JISA）による「情報サービス産業白書」の 1992 年版[51]と 1993 年版[52]より作成した表では，加盟会社 243 社に対する新卒の採用計画達成率は 85.3%(91 年)，106.0%(92 年)と推移しており，89 年まで低下傾向にあった計画達成率が上昇している．また，中途採用の計画達成率においても 88.0%(90 年)，113.7%(92 年)と推移し，新卒採用計画と同様に 89 年まで低下傾向であった計画達成率が上昇している．新卒と中途採用の採用計画達成率を表 2，表 3 に示す．採用計画の達成率は，1992 年以外は 100%未満であり採用に関して困難な状況が伺える．将来的にはこの状況が継続し，情報サービス産業におけるソフトウェア技術者の採用が厳しい状況になると考えられる．

2.3 派遣労働力の需給状況

2.3.1 派遣労働の形態

派遣労働は，労働力の需要に対する労働者の供給として行われているがその形態は一様でなく，図 1 に示されるような 5 つの形態に分類できる[7],[37].

(1) 労働者派遣事業は派遣元，労働者，派遣先で構成される．労働者派遣法は大旨, 現在の技術革新下に適合した労働力需給策を規定していると考えられる．労働者派遣法は，昭和 60 年法律第 88 号として制定されて改定を重ねながら現在に至っている．第 1 条では労働者需給の適正な調整と派遣労働者の雇用の安定，その他福祉の増進を目的としている．第 2 条において労働者派遣法の対象となる労働者派遣とは，派遣元（人材派遣会社）が自ら雇用する労働者（派遣社員）を，派遣先（派遣社員使用会社）の指揮命令を受けて労働させる，となっている．

(2) 労働者供給事業は，供給元，労働者，供給先で構成され，この事業は労働組合以外に禁止されている．（職業安定法第 44 条）

(3) 請負は，請負業者，労働者，注文主で構成され，作業に従事する労働者を指揮監督するのは派遣元のみで（職業安定法第 4 条 2 号），注文主が労働者に詳細な指示が出せない．

(4) 職業紹介は，求人者，職業紹介機関，求職者で構成され，特別の技術（美術，音楽，演芸その他）を必要とする職業に限り労働大臣の許可により許されている（職業安定法第 32 条）．

(5) 出向型は，出向元，労働者，出向先で構成され，出向先と出向元の関係が深い場合に用いられて出向元の経費負担の割合が大きい．

情報システムの開発・運用・保守では，派遣先企業の担当社員との詳細な指示やコミュニケーションが重要になり，派遣されたソフトウェア技術者に専門的な知識・技術が要求される．このような場合，(3)請負では注文主の企業がソフトウェア技術者に詳細な指示が出せないため業務への支障が考えられる．また(2)労働者供給事業と(4)職業紹介の適用範囲が特定の組織や技術に限られるため情報システムの開発・運用・保守には適用できない．(5)出向型は，出向先と出向元の関

係，出向元の経費負担の増大などにより一般的な活用ができない．(1)労働者派遣事業は，企業内の社員だけでは処理が困難な専門的知識・技術を必要とする業務に対応し，派遣先企業が派遣されたソフトウェア技術者の指揮監督が容易であるため情報システムの開発・運用・保守に適していると考えられる．

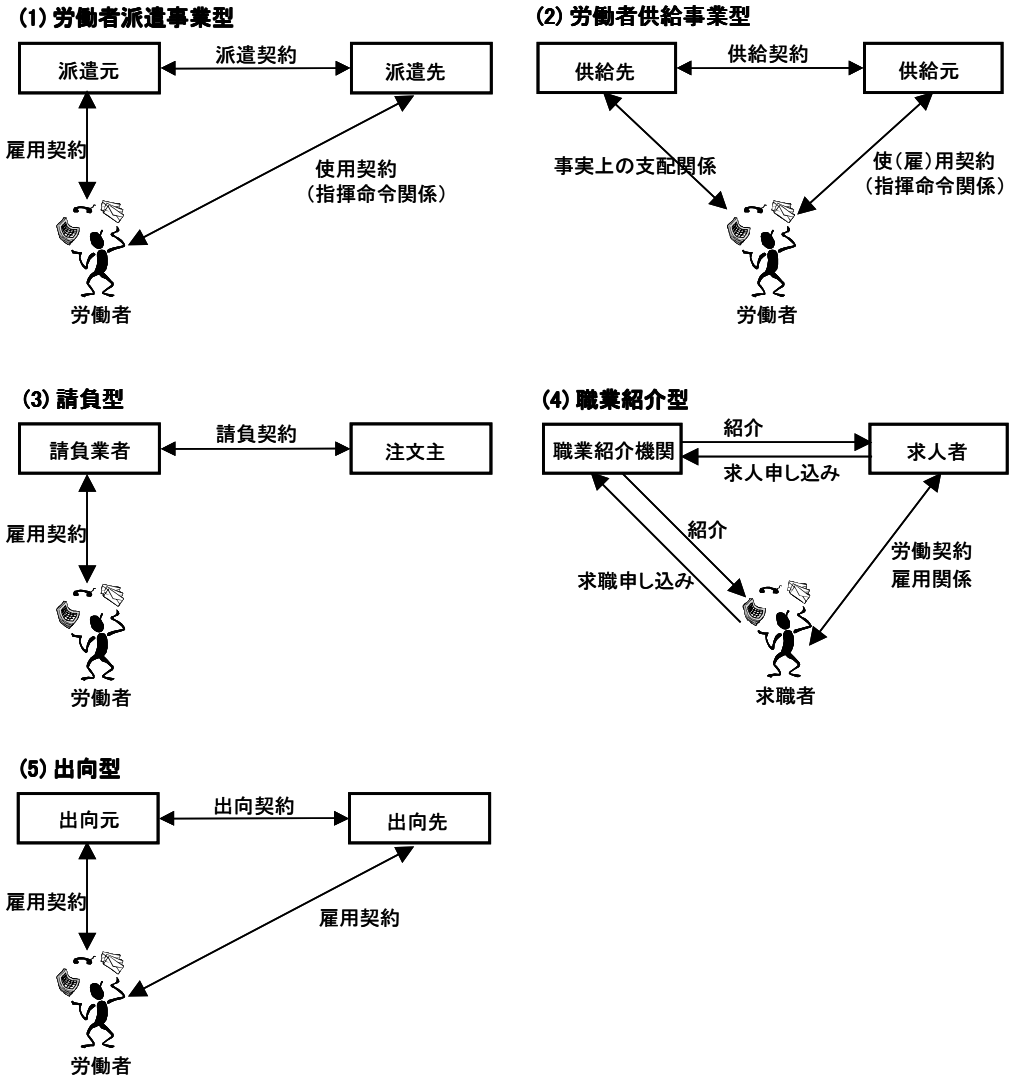


図1　労働力の派遣形態の分類

2.3.2 ソフトウェア技術者の労働力需給状況

JISA が国内の 1･2 部上場企業および金融機関 658 社を対象に 1992 年 7 月に実施した「情報システム化の現状と将来動向の調査」（ユーザ・アンケート調査）による産業界における情報システムの構築状況と潜在的な情報化需要の動向の分析によると，企業での情報システム部門の要員は派遣要員も含め，1 社あたり平均 63.4 人となっている．これを内部要員と派遣要員に分けると，内部要員が全体の 61.0%，派遣要員が 39.0%になり，派遣要員の内訳はシステムエンジニアが 29.1%でプログラマが 44.4%になる[53] ． 企業での情報システム部門の派遣要員の比率が約 40%の高い割合で占められており，その技術力は情報システム部門に大きな影響を及ぼすと考えられる．

国内における派遣労働者を対象とした教育訓練は，2000 年でその種類またはコースが延べ 30,948 コース，対象者数は延べ 732,903 人である[13]．アメリカにおいては，調査対象となった労働者の約 84%が訓練施設や教育機関等を活用した公式訓練を受け，約 96%が OJT（On the Job Training）等の非公式訓練を受けていると報告されている[12]．これら教育訓練の多くは，当面予想される業務に必要な実務能力をできるだけ早く修得させることにあり，今後必要となる専門分野に特化した高度なソフトウェア技術者の教育を行う必要がある．

情報通信技術の導入による業務の標準化が可能になったことで，外部の人材活用が容易になり非正規雇用の活用を促進している．平成 13 年版労働経済白書の分析によれば，情報化が進んでいる企業ほど情報化を非正規雇用の活用理由にあげる割合が高く，情報通信技術の導入によってアルバイトなどの比率が高まるとする企業が 35%，派遣労働者の比率が高まるとする企業が 35%に及んでいる[14]．1998 年の情報通信技術に関連（ソフトウェア開発，事務用機器操作，OA インストラクション）する派遣労働の比率は，派遣労働者の 45.7%（41 万人）を占めている．この分野の派遣労働者は，1993 年から 1998 年の 5 年間に 88.5%（19 万人）増加している．今後も情報システムの開発・運用・保守の専門的分野や，一時的なプロジェクトの分野においても派遣労働が進むと考えられる[14]．

2.4　ソフトウェア技術者の派遣労働者における教育支援体制

2.4.1　情報システム部門における労働力需給の課題

情報システムの開発・運用・保守を迫られている企業にとっては，ソフトウェア技術者の確保は重要である．企業では情報システム部門の技術を担当するソフトウェア技術者が必要になるが，適切な人材がない場合はソフトウェアの技術を持った人材の育成か，新規採用という選択を迫られることになる．このような場合，次のような問題が考えられる．① ソフトウェアの技術を持った人材を育成するには，日常業務の時間を割いて教育に充当している場合が多い．このような場合，本来の業務に支障が生じるか，ソフトウェアに関する教育が業務に阻まれて計画どおり進行しない．② 新規採用の場合，その人材が企業の業務や開発環境固有の知識が無いため，採用された企業でソフトウェア技術者として就業できるようになるには一定期間の教育が必要になる．

このような問題への対処の一つに適切な技術・知識を有するソフトウェア技術者の派遣が活用される．情報システムへの人材派遣においては技術・知識レベルが重要であることから，ソフトウェア技術者の労働力需給において次のような課題が考えられる．

① 経済情勢の変動などが，ソフトウェア技術者の長期的な需要に影響を及ぼす．
② 派遣先の企業業績の変化が，ソフトウェア技術者の短期的な需要に影響を及ぼす．
③ 突然の派遣依頼が発生し，ソフトウェア技術者の需給ギャップが生じる．
④ ソフトウェア技術者に派遣先の開発環境固有の知識が無いと，業務の遂行に支障が生じる．
⑤ ソフトウェア技術者は，派遣先の開発環境に関する知識・技術を短期間に学習する必要に迫られる．
⑥ 情報システム部門の労働は知的作業であるため，ソフトウェア技術者の知識・技術レベルによって処理時間に差異が生じる．
⑦ ソフトウェア技術者ごとの処理時間の差異により，業務に対する適正な労働時間の確保が難しく，労働者派遣法による監督行政の困難さが存在する．

⑧ ソフトウェア技術者の労働過程は有形化しづらい側面があるため，派遣元と派遣先の労働評価に差異が生じることがある．

2.4.2 ソフトウェア技術者の教育支援体制の提案

ソフトウェア技術者の労働力需給の課題への対応を次に上げる．

① 課題 ①（経済情勢などの変動によるソフトウェア技術者の需要ギャップ）に対処するため，長期的な需要予測を行う．需要予測は，1986 年からの景気拡大期，その後の景気後退期における経験やソフトウェア技術者の需要実績に加え，現在の派遣対象である産業分野の情勢，地域特性，経済と技術革新の動向を分析して行う．

② 課題 ②（派遣先の企業業績の変化）に対応するため，個々の派遣先企業において要求される派遣人数と技術レベルの短期的な需要予測を行う．需要予測は，派遣されたソフトウェア技術者からの報告や，派遣先企業における情報システムの稼動年数などに基づいて行う．

③ 課題③（突然の派遣依頼）に加え課題 ①，課題 ②の外的要因への対応は，余裕のあるソフトウェア技術者の確保が必要である．そのためには，受託計算業務，ソフトウェア開発，キャリアアップスクールといった関連業務を兼業しながら労働者派遣事業を行うことが望ましい．

④ 課題 ④（知識の有無による業務への支障），課題 ⑥（処理時間に差異），課題 ⑦（適正な労働時間の確保）への対応には，需要予測に基づいた計画的なソフトウェア技術者の技術教育が必要である．効率的な技術教育を実現するため，派遣元企業が教育機関と短期間または一定期間の連携を行う．教育機関との連携の例を次に示す．

(a) 派遣労働者や社会人の技術教育は，アメリカの MIT Sloan School of Management における社会人を対象とした社会人大学院のような教育機関がある．

(b) 国内においても本学 奈良先端科学技術大学院大学のように，入学希望者の専攻分野にとらわれず，企業で活躍中の技術者なども受け入れる体制が確

立されてきており，社会人に対して高度な技術教育が行える．

(c) 18 歳人口の減少にともない, 社会人を対象とした公開講座が大学を中心に開講されており，このような講座を活用した技術教育が可能である．

⑤ 課題 ⑤（知識・技術の短期間の学習）の対応では効率的な教育が重要になり，派遣元の教育担当者とソフトウェア技術者自身が，その知識・技術を系統的に把握する必要がある．したがって課題 ⑤は，プログラミング教育の体系化によって実現できると考えられる．

⑥ 課題 ⑧（労働過程の有形化）は，技術教育の資料などを基に評価基準を作成し，派遣先の労働評価に対処できるようにする．

派遣元が，派遣先からの人的数量の需要に対応するための組織的な管理手法として，(a) 長期的需要予測，(b) 短期的需要予測，(c) 関連業務の兼業，(d) 教育機関との連携, を統合した教育支援体制を図 2 に提案する．派遣元は, 経済情勢, 派遣先の企業業績，情報システムの稼動年数などを基に需要予測を行い，ソフトウェア技術者の確保のために関連業務を兼業し，必要になるソフトウェア技術者の知識・技術の教育を教育機関との連携によって行う．

2.4.3 教育支援体制の考察

ソフトウェア技術者の需給における課題に対応した教育支援体制の考察を行う．

(a)長期的需要予測は，将来の雇用計画の立案を可能にし，安定したソフトウェア技術者の雇用と経営を実現できると考えられる．派遣者派遣法の制定以来，情報システム部門への派遣労働は国内の高度成長に支えられながら拡大してきたが, 近年の景気後退による派遣労働の需要減少への対処が困難であった．派遣労働は企業活動の期間が短く，派遣元企業にとって景気後退が初めての経験であったため，長期的な需要予測が行えなかったことが要因と考えられる．したがて，教育支援体制において長期的な需要予測は重要であると考える．

(b)短期的需要予測は，個々の派遣先に対し短期的な需要予測を行うことで，需要予測の精度が向上すると考えられる．例えば，現在は情報システムの導入が進み, 1980 年代後半における汎用コンピュータの新規導入が増加した時期とは異な

ったソフトウェア技術者の需要傾向にある．よって，派遣先における情報システムの開発・運用・保守の状況から新しいシステム開発の時期が予測できるようになり，ソフトウェア技術者の詳細な需要予測が可能になると考えられる．

(c)関連業務の兼業により，突然の派遣依頼や外的要因による需給ギャップが吸収でき，安定した雇用が促進されると考えられる．

(d)教育機関との連携は，先端的な技術の習得、教育コストの削減が可能になり、高度な技術を必要とする派遣に対応できると考えられる．一定期間の技術教育は，ソフトウェア技術者の知識・技術の陳腐化を防ぐことにも繋がる．また技術教育の充実は，ソフトウェア技術者の知識・技術レベルを平準化して処理時間の差異を縮小できる．その結果，計画的な労働が実現され，適正な労働時間の確保にも繋がると考えられる．

このような景気変動を踏まえた需要予測と技術教育を行い，ソフトウェア技術を有する派遣元の要員を派遣先企業へ派遣することで，安定した雇用の促進とより高度なソフトウェア技術者の派遣を実現できる．さらに，労働力需給の課題に対応し，ソフトウェア技術者が高度な技術力を養うことで，派遣労働の価値が高まると考えられる．

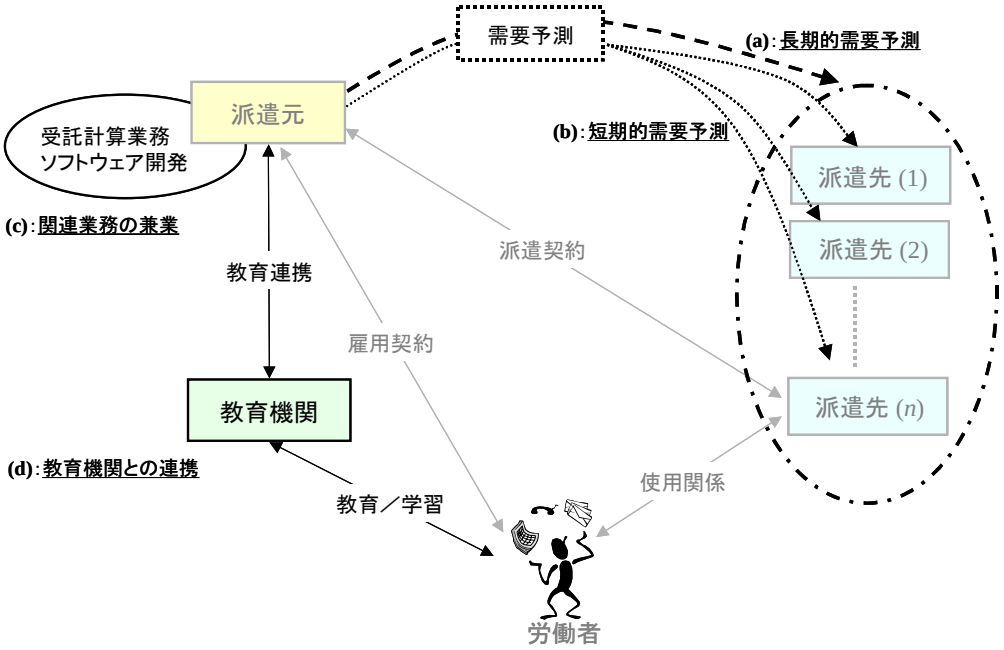


図2　労働者派遣業におけるソフトウェア技術者の教育支援体制

2.5　派遣労働における技術教育の考察

派遣労働は，企業にとっては経営の合理化という効果があるが，それにともなう弊害も見逃せない．その弊害の一つに，正社員では企業経営の日本的特殊性（終身雇用慣行，年功序列制度，企業内労働組合，包括的労働契約，企業内教育訓練制度）による企業への愛社心，忠誠心や帰属意識等が存在するが，派遣労働者ではこれらが減退し業務に対する積極性を欠くことになり，派遣労働者の比率が高くなると，企業経営に悪い影響を及ぼすと考えられる．

派遣労働者は，労働力の需要に応え，派遣先企業で歓迎され華やかな一面があるが，労働力の需要がなくなると派遣対象業務から排除されることから業務に対する愛着感が薄れ，労働者としての意識に欠けることにもつながる．派遣労働は多くの企業で採用され，外部からの観察では正社員と見分けづらく，派遣先企業の一員として就業して活躍している．しかし，内面的には企業への帰属意識や責任感などに相違があり，正社員と同一の意識で就業していない場合は，その意識改革を行う必要が生じる．

派遣先企業が派遣元へ支払う金額と派遣元から派遣労働者に支払われる金額とに差があり，派遣労働者の賃金に対する誤認が，派遣労働者と派遣元の関係に悪い影響を與えている．この差額は派遣先企業での募集費，福利厚生費，事務費，企業利益などが計上された結果として生じているが，派遣労働者へは知らされていない場合が多く，支給金額への認識不足から勤労意欲を減退させる可能性がある．このような状況を避けるため，労働力の派遣形態を明確に整備・分類し，派遣労働者に理解を促すことが重要である．

労働者派遣業においては，派遣元企業が自ら雇用する労働者を派遣先企業へ派遣する．労働者は，派遣が終了すれば派遣元の社員であり指揮・命令は派遣元にある．よって，一般企業と同様に企業経営の日本的特殊性の一つである企業内教育訓練制度を適用することができ，派遣元が需要予測に基づいた技術教育の計画を立案し，技術教育を社員の業務として行わせることができる．このような技術教育の観点から，ソフトウェア技術者のように専門的な知識・技術を必要とする労働力の派遣形態には，労働者派遣業が適していると考えられる．

2.6　まとめと今後の課題

本章では，情報サービス産業における雇用状況とソフトウェア技術者の労働力需給状況を分析し，労働力の派遣形態を分類して労働者派遣事業がソフトウェア技術者の派遣労働に適していることを示した．またソフトウェア技術者が不足するとされていた需給予測を，CASE などの開発環境の充実とソフトウェア工学の各種手法による開発技術の進歩により，ソフトウェア技術者の需給ギャップが解消されていることを考察した．

ソフトウェア技術者の労働者派遣事業における労働力需給の課題を整理した．そして課題への対応として，過去の景気変動やソフトウェア技術者の派遣実績を生かした長期的需要予測，ソフトウェア技術者の報告による短期的需要予測，需給ギャップを吸収するための関連業務の兼業，派遣元企業と教育機関との連携を上げた．派遣元が派遣先からの人的数量の需要に対応するための組織的な管理手法として，長期的需要予測，短期的需要予測，関連業務の兼業，教育連携が統合された教育支援体制を提案した．教育支援体制により，情報システムの開発・運用・保守におけるソフトウェア技術者の継続的で安定した派遣労働を実現できると考えられる．

また労働者派遣法によって派遣労働の形態が明瞭になり，ソフトウェア技術者の安定した労働環境が形成されてきているが，ソフトウェア技術者の技術評価基準を明確にし，技術教育の成果も技術力の評価に加えることも重要であると考えられる．今後は，教育支援体制における教育連携の具体的な仕組みを提案し，課題⑤に対応するためのプログラミング教育の体系化に関する研究に展開したいと考えている．

３．プログラミング学習における理解状態の解析

3.1　あらまし

学習者の思考や理解状態に関する研究はさまざまな分野で行われ[2],[4],[21],[25],[30]，プログラミング学習の解析にも適用されている．プログラミング言語のような複合的な理解を必要とする学習において学習者の理解状態の把握は重要である．プログラミング言語の関連知識をもとにして実際にプログラミング演習を行った後では，理解状態に変化が生じることは経験的に知られている．このような学習者の理解状態の変化は，プログラミング言語の学習に関する因子の遷移とプログラミング言語固有の概念に関連があると考えられる[42]．プログラミング言語を効率的に学習するには，学習者の理解状態の変化を解析して学習の進行にともなう理解状態に適応した学習形態の形成が重要であると考えられる．

本章では，学習者の理解状態の変化に適応した学習形態を構築するため，プログラミング学習の進行にともなう学習者の理解状態の変化に着目した．プログラミング言語の学習状態を,① プログラミング言語の学習以前でその知識が無い段階，② プログラミング言語の基礎的な知識を学習した段階,③ その知識を基にしたプログラミング演習によってプログラミングを経験した段階，に分類する．それぞれの学習過程においてテストを行い，テスト結果の因子分析から抽出した因子の遷移によって学習者の理解状態を解析する．

3.２　プログラミング学習における理解状態の測定

3.2.1　理解状態の測定方法

本章ではプログラミング言語における学習段階を，①プログラミング言語学習の前，②教材や資料による学習の後，③プログラミング演習による学習の後（以下，学習段階①，学習段階②，学習段階③と呼ぶ）に分類し，プログラミング学習の対象を Java 言語にした．

被験者は, Java 言語に関する知識が無い学習者ほど理解状態の変化が明確であ

ると考え, 本学修士前期課程 1 学年で Java 言語に関する知識が無い 5 名である. 因子分析の結果を詳細に分析するため，被験者のプログラミング経験や現在のプログラミング言語の学習状況についてアンケート調査も行った．被験者は，それぞれの学習段階の終了時に Java 言語に関する 20 問のテストを受ける．Java 言語の理解状態の測定実験を以下の手順で行った.

① 学習段階① において，学習者に Java 言語に関するテストを行う.
② テスト終了後に Java 言語に関連する学習資料を配布し，その内容について解説する．(120 分間)
③ 学習段階② が終了し，再度，同内容のテストを行う.
④ Java 言語のプログラム 2 例を用いてパーソナルコンピュータによるプログラミング演習を行う．プログラム例はソースコードの穴埋めと，計算処理の出力である．演習では，2 例のソースコードの加筆修正とコンパイルを行ってプログラムを実行する．(120 分間)
⑤ 学習段階③ が終了し，再度，同内容のテストを行う.

3.2.2 テスト内容と回答手順

学習段階①，学習段階②，学習段階③における理解状態を，20 問のテスト結果によって測定する．学習段階① で用いた知識項目とテスト問題を表 4 に示す. 知識項目は, テストの問題を正解するために必要な知識を示している. テストは, 問題①～⑩がオブジェクト指向の関連知識，問題⑪～⑳がプロラム言語の関連知識で，正解までの回答回数，回答時間をテスト結果として記録した.

テストは，同じ知識項目と難易度のテスト問題を 3 種類作成し，それぞれの学習段階の終了時にテストを行う．実験のためのテストシステムを Java 言語で開発した．テストシステムは，5 つの解答例の中から一つの解答を選択する多岐選択方式で機能は以下のようになる．テストシステムの画面構成を図 3，回答過程を図 4 に示す.

表4　知識項目とテスト問題

(a) 知識項目

1	2	3	4	5
アイデンティティ	分類	ポリモーフィズム	インヘリタンス	属性
6	7	8	9	10
機能	カプセル化	属性	クラス	スーパークラス
11	12	13	14	15
クラス	インスタンス	メソッド	クラスライブラリ	インスタンス変数
16	17	18	19	20
インスタンス化	スーパークラス	クラスメソッド	インタフェース	パッケージ

(b) テスト問題

問1	オブジェクトの特徴の一つで、データが個々に区別されるオブジェクトという実態で表現できる。
問2	オブジェクトの特徴の一つで、同じデータ構造（属性）と、同じ振る舞い（操作）を持つオブジェクトを、１つのクラスにグループ化できる。
問3	オブジェクトの特徴の一つで、同じ操作であっても、異なるクラスに対しては異なる振る舞いをする。
問4	オブジェクトの特徴の一つで、クラスの階層関係に基づいて、クラス間で属性と操作を共有できる。
問5	あるオブジェクトと別のオブジェクトの違いを表現する。
問6	データに対する手続き的な操作のことをいう。
問7	ほかのオブジェクトから直接、オブジェクトの中の変数（インスタンス変数）をアクセスできないようにする。
問8	多角形クラスにおいて、辺、境界色、内部色は、何を意味しているか。
問9	Personal Powered Vehicle とEngine Powered Vehicleのクラスはどれになるか。
問10	サブクラスにあたる野球ボール、バレーボールのスーパークラスはどれになるか。
問11	オブジェクトの属性をあらわす「変数」と、動作を表す「メソッド」が定義されている。
問12	個々のオブジェクト（もの）と同じ意味のことで、クラスが「ものの抽象的な表現」であるに対し、これは具体的なものを表している。
問13	クラス内で定義される関数で、そのクラスのインスタンスに対して適用される。
問14	プログラムの部品として使えるようなクラスである。
問15	個々のインスタンスが所有している変数で、その値はインスタンスごとに異なっている。
問16	クラスからインスタンスを生成することをいう。
問17	さまざまなクラスの基になるクラスのことをいう。
問18	そのクラスに作用するか、またはどのオブジェクトと特定せずに使えるようなメソッドのことをいう。
問19	個々のクラスが実現することができる、抽象的な動作の使用の集まりをいう。
問20	クラスとインタフェースの集まりのことをいう。

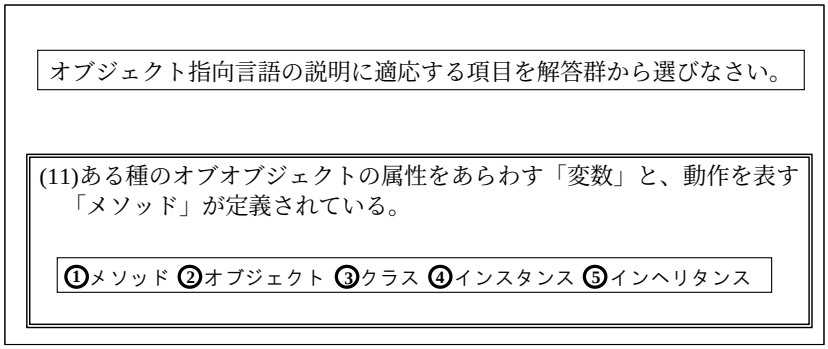


図3　テストシステムの画面構成

① 回答は，ウィンドウに表示された 5 つの回答群から一つをクリックして選択する．

② 回答番号をマウスでクリックした時点で，学習者の回答番号，回答時間，回答回数が記録される．

③ 学習者がクリックした回答を自動判定し，その結果を”〇”，”×”で表示する．

④ 回答が間違いであれば，再度，学習者が回答を入力する．

⑤ 正解であれば，同じウィンドウに次の問題と回答群が表示される．

⑥ 最後の問題が正解であればテストを終了する．

3.2.3 テスト結果の処理

因子分析は，一般的に①量の変化を測ることによる分類，②多くの変量をいくつかの少ない変量にまとめる，③因子負荷量から情報の集約ができるので，本章においてもテスト結果の解析に因子分析を用いた[34]．本章では因子負荷量から学習段階①，学習段階②，学習段階③ における因子を抽出し，学習段階の進行に

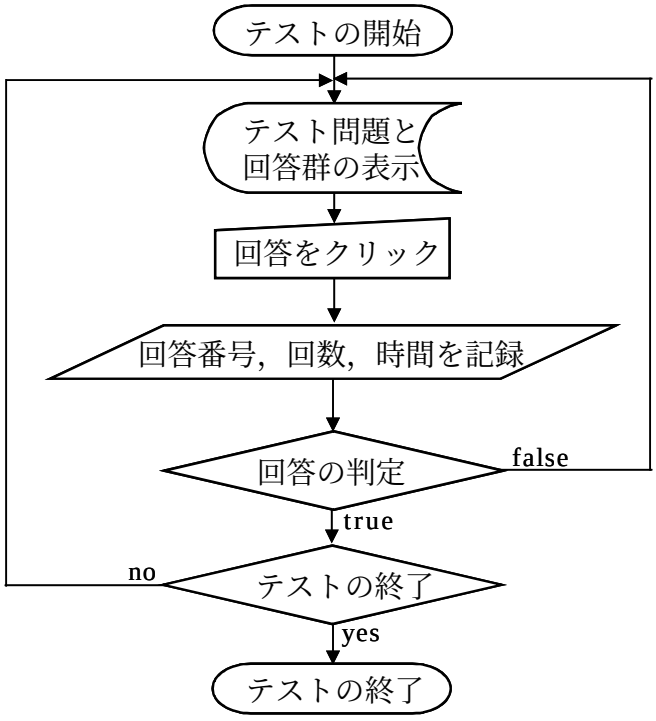


図4　テストシステムにおける回答過程

よる理解状態の変化を解析する.

実験においてテストシステムで記録された回答時間と回答回数から，それぞれの知識項目における学習者の回答を 1 位から 5 位の順位に設定する．順位は回答回数が少なければ上位になり，同じ回答回数であれば回答時間を優先する．それぞれの知識項目における学習者の順位を点数（1 位：50 点，2 位：40 点，3 位：30 点，4 位：20 点，5 位：10 点）に変換して因子分析を行う.

3.3 学習段階における理解状態の解析

3.3.1 テスト結果の因子分析

学習段階①，学習段階②，学習段階③ におけるテスト結果を因子分析し，算出された知識項目の因子負荷量を表 5，6，7 に示す.

因子分析において，固体i の変量 j の観測値 x_{ij} を，変量ごとに標準化したものを z_{ij} で表すと，因子分析モデルの基本式は式(1)のようになる.

$$z_{ij} = a_{j1} f_{i1} + \cdots\cdots + a_{jk} f_{ik} + \cdots\cdots + a_{jm} f_{im} + e_{ij} \qquad (1)$$

z_{ij}： 固体i の変量 j における標準得点（Z-score）

$a_{j1}, \cdots a_{jk}, \cdots a_{jm}$： 因子負荷量（factor loading）

$f_{i1} \cdots f_{ik} \cdots f_{im}$： 因子スコア（factor score）

e_{ij}： 誤差（error）

ここで a_{jk} は 変量 j への k 因子（$k = 1,2,\cdots\cdots,m$ ）との結びつきの度合いを示し因子負荷量と呼ばれる．f_{ik} はすべての変量に共通に関与する変数(共通因子)で，その値が因子スコアと呼ばれる．e_{ij} は共通因子で説明しきれない誤差分を表している．知識項目 j における k 番目の因子負荷量は，知識項目 j と第 k 因子との結びつきの度合いを表し，因子負荷量が大きければ知識項目 j と第 k 因子との結びつきが強いことを意味する．因子分析の解を求めることは因子負荷量を求めることを意味することが多く，因子負荷量によって因子の特徴を集約的に表すことができる[34]．因子負荷量の絶対値がある定数以上の値を示す項目を抜き出せば，これらに共通するものを因子として抽出できる．本章では，k 番目の因子における因子負荷量の絶対値が 0.9 以上の知識項目から第 k 因子を抽出した.

表5 学習段階① の Factor Loading

知識項目	第一因子	第二因子	第三因子
1	-0.3064	0.7816	0.5047
2	0.2008	0.7284	0.1246
3	-0.8912	0.1253	0.4329
4	0.8101	-0.5247	-0.2502
5	0.1331	-0.0765	0.9881
6	-0.9983	-0.0361	0.0301
7	0.1924	-0.3226	0.3618
8	-0.8101	0.5247	0.2502
9	0.0516	0.9475	-0.1334
10	-0.3022	-0.1017	0.6464
11	0.0516	0.9475	-0.1334
12	-0.0555	0.7861	-0.5361
13	-0.7898	-0.5014	-0.3382
14	-0.9983	-0.0361	0.0301
15	-0.9983	-0.0361	0.0301
16	-0.6108	-0.4184	-0.6572
17	0.0516	0.9475	-0.1334
18	-0.1530	-0.3209	0.9044
19	-0.1824	0.2273	0.9537
20	-0.9003	-0.3525	-0.1063

表6 学習段階② の Factor Loading

知識項目	第一因子	第二因子	第三因子
1	-0.8968	0.3766	-0.2302
2	0.9841	-0.1129	-0.0959
3	0.9614	0.2199	0.0693
4	0.4066	0.8685	-0.1913
5	0.1525	-0.9176	-0.3383
6	0.8741	-0.0438	0.3954
7	0.0180	0.8453	0.4766
8	0.4066	0.8685	-0.1913
9	0.2128	-0.9208	0.2912
10	0.3150	0.0053	0.9251
11	-0.8544	-0.4719	-0.0772
12	0.3150	0.0053	0.9251
13	0.6828	0.1275	0.2460
14	-0.6601	-0.4603	-0.4112
15	-0.1974	-0.1945	0.8173
16	-0.1340	-0.0147	0.9635
17	0.7491	-0.6371	-0.0733
18	0.2080	0.2574	0.9330
19	0.9841	-0.1129	-0.0959
20	0.7910	0.2922	-0.0689

表7 学習段階③ の Factor Loading

知識項目	第一因子	第二因子	第三因子
1	0.6856	-0.0334	0.5258
2	-0.1985	-0.0476	0.9376
3	0.1094	0.1474	0.9827
4	0.9245	0.2521	0.1707
5	-0.1729	0.6223	0.6490
6	-0.8980	0.0850	-0.4238
7	-0.0806	-0.6785	0.7239
8	0.7346	-0.5738	-0.0881
9	0.1210	0.9164	-0.1415
10	0.6856	-0.0334	0.5258
11	0.8291	0.0056	0.0236
12	0.8841	0.0142	-0.4117
13	0.9520	0.2564	-0.0470
14	-0.4573	-0.7829	0.2786
15	0.2273	-0.4835	0.7690
16	0.6581	-0.0377	0.7435
17	0.8575	-0.3229	-0.1586
18	0.8291	0.0056	0.0236
19	0.1189	-0.9637	-0.1782
20	0.0914	-0.9680	0.0394

3.3.2 学習段階の因子の抽出

学習段階① の第一因子は, 知識項目⑥ (機能), ⑭ (クラスライブラリ), ⑮ (インスタンス変数), ⑳ (パッケージ) が絶対値 0.9 以上の高い因子負荷量を示している. これら知識項目は学習者が知らない用語であることに加え, 学習者がテストシステムにおいて回答した回答番号の記録から, 回答群の回答番号の ”①” から順番にクリックしたと考えられる. これら知識項目の正解番号が回答群の後半にあることから, この因子を「回答番号の順番押しの傾向 (正解番号が後半)」とした. 第二因子は, 知識項目⑨ (クラス), ⑪ (クラス), ⑰ (スーパークラス) の共通な因子として「オブジェクト指向の基礎理解 (1)」とした. 第三因子は, 知識項目⑤ (属性), ⑱ (クラスメソッド), ⑲ (インタフェース) が絶対値 0.9 以上の高い因子負荷量を示し, 第一因子と同様に学習者の知らない用語であった. 第三因子に関連する知識項目の正解番号が第一因子とは逆に回答群の前半に置かれていることに加え, テストシステムに記録された回答番号の分析から, この因子を「回答番号の順番押しの傾向 (正解番号が前半)」とした.

学習段階② の第一因子において絶対値 0.9 以上の高い因子負荷量を示す知識項目は, ② (分類), ③ (ポリモーフィズム), ⑲ (インタフェース) である. これら知識項目の共通な因子として「オブジェクト指向プログラミングの高度な概念」とした. 第二因子は, 学習段階① の第二因子と共通する知識項目⑨ (クラス) に加え, 知識項目⑤ (属性) が含まれていることから「オブジェクト指向の基礎理解 (2)」とした. 第三因子は, 知識項目⑩ (スーパークラス), ⑫ (インスタンス), ⑯ (インスタンス化), ⑱ (クラスメソッド) に共通する因子として「基礎的なプログラミングに必要な知識 (1)」とした.

学習段階③ では, 第一因子において絶対値 0.9 以上の高い因子負荷量を示す知識項目が④ (インヘリタンス), ⑬ (メソッド) である. また, これら知識項目の直近に学習段階② で第三因子を抽出した知識項目⑫ (インスタンス), ⑱ (クラスメソッド) を含んでいることから, この因子を「基礎的なプログラミングに必要な知識 (2)」とした. 第二因子は, 学習段階② と同様に知識項目⑨ (クラス) に加え, 知識項目⑲ (インタフェース), ⑳ (パッケージ) が含まれる. これよ

り第二因子を「オブジェクト指向の基礎と高度なプログラミングに必要な Java 言語固有の概念」とした．第三因子は，知識項目②（分類），③（ポリモーフィズム）でプログラミングに直接影響しない知識であることから「プログラミング言語に依存しない抽象的な概念」とした．

3.3.3　因子負荷量による因子の遷移の観察

学習段階①，学習段階②，学習段階③ における因子の遷移を，因子負荷量の変化をもとに解析する．k 番目の因子において因子負荷量が変化すれば，異なる知識項目が抜き出されて抽出される第 k 因子も変化するので，因子負荷量の変化から因子の遷移を観察できると考えられる．

学習段階② における第一因子の遷移を，因子負荷量の変化によって観察した一例を図 5 に示す．学習段階② の第一因子を抽出した”分類”（知識項目②）と”ポリモーフィズム”（知識項目③）は，学習段階③ の第三因子を抽出する知識項目

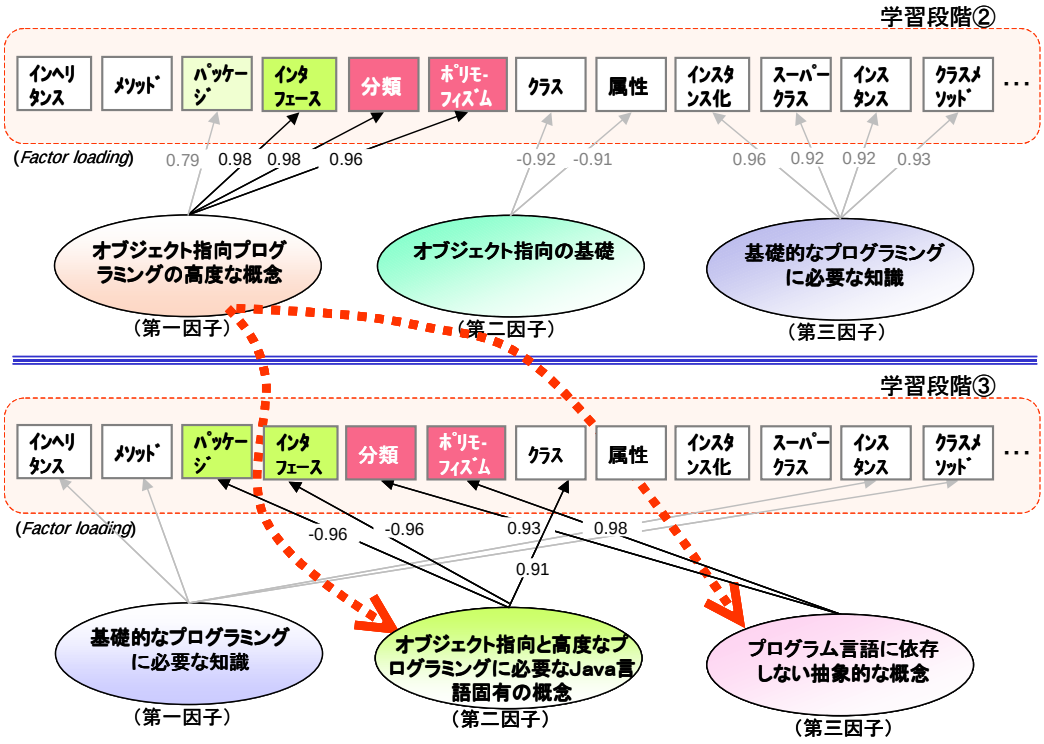


図 5　因子負荷量による因子の遷移の観察

として表れている．また，学習段階② の第一因子を抽出した”インタフェース”（知識項目⑲）と因子負荷量 0.79 の”パッケージ”（知識項目⑳）が，学習段階③ の第二因子を抽出する知識項目に表れている．このような場合，学習段階② の第一因子が，学習段階③ の第二因子と第三因子に分かれて遷移したと観察できる．本章では，学習段階の進行による因子の遷移を図 5 に示したような因子負荷量の変化によって観察する．

3.3.4　学習段階の進行による因子の遷移

本章では被験者数と知識項目数を考慮し，因子負荷量の絶対値が 0.9 以上の値を示す知識項目を抜き出して因子を抽出した．図 6，7，8 は，それぞれの学習段階における因子負荷量の絶対値が 0.9 以上で，同じ知識項目が学習段階の進行によって異なる学習段階の因子に遷移した状態を表している．学習段階 s の第 k 因子において因子負荷量の絶対値が 0.9 に近い値を示す知識項目を “ 直近の知識項目 j ” と呼ぶ．

学習段階②の第一因負荷量

知識項目	第一因子
2 分類	0.9841
19 インタフェース	0.9841
3 ポリモーフィズム	0.9614
1 アイデンティティ	-0.8968
6 機能	0.8741
11 クラス	-0.8544
20 パッケージ	0.7910
17 スーパークラス	0.7491
13 メソッド	0.6828
14 クラスライブラリ	-0.6601
4 インヘリタンス	0.4066
8 属性	0.4066
10 スーパークラス	0.3150
12 インスタンス	0.3150
9 クラス	0.2128
18 クラスメソッド	0.2080
15 インスタンス変数	-0.1974
5 属性	0.1525
16 インスタンス化	-0.1340
7 カプセル化	0.0180

学習段階③の第二因負荷量

知識項目	第二因子
20 パッケージ	-0.9680
19 インタフェース	-0.9637
9 クラス	0.9164
14 クラスライブラリ	-0.7829
7 カプセル化	-0.6785
5 属性	0.6223
8 属性	-0.5738

学習段階③の第三因負荷量

知識項目	第三因子
3 ポリモーフィズム	0.9827
2 分類	0.9376
15 インスタンス変数	0.7690
16 インスタンス化	0.7435
7 カプセル化	0.7239
5 属性	0.6490
1 アイデンティティ	0.5258

図 6 学習段階② から学習段階③ への因子負荷量の変化(1)

図 6 では，学習段階② の第一因子を抽出した知識項目②，③，⑲ が，学習段階③ の第三因子へ遷移している．また，学習段階② の第一因子を抽出した知識項目⑲ と直近の知識項目⑳（因子負荷量 0.791）が，学習段階③ の第二因子に遷移している．このような学習段階② における知識項目②，③，⑲，⑳ の学習段階③ への遷移は，プログラミング演習による理解状態の変化で，学習段階② の第一因子が学習段階③ の第二因子と第三因子に分かれて遷移したことを意味していると考えられる．

図 7 では，学習段階② の第三因子を抽出した知識項目⑩，⑫，⑯，⑱ の中で，知識項目⑫，⑱ が学習段階③ の第一因子を抽出した知識項目④，⑬ の直近の知識項目として遷移している．知識項目⑫，⑱ の遷移から，学習段階② の第三因子の一部が学習段階③ の第一因子へ遷移したことを意味していると考えられる．

図 8 では学習段階① の第二因子を抽出した知識項目⑨，⑪，⑰ の中で，知識項目⑨ が学習段階② の第二因子へ遷移し，その後，学習段階③ の第二因子へ遷移している．学習段階①，学習段階②，学習段階③ の第二因子における知識項目⑨ の遷移は，それぞれの学習段階において第二因子の中心的な部分が変わらない

学習段階②の第三因負荷量

知識項目	第三因子
16 インスタンス化	0.9635
18 クラスメソッド	0.9330
10 スーパークラス	0.9251
12 インスタンス	0.9251
15 インスタンス変数	0.8173
7 カプセル化	0.4766
14 クラスライブラリ	-0.4112
6 機能	0.3954
5 属性	-0.3383
9 クラス	0.2912
13 メソッド	0.2460
1 アイデンティティ	-0.2302
4 インヘリタンス	-0.1913
8 属性	-0.1913
2 分類	-0.0959
19 インタフェース	-0.0959
11 クラス	-0.0772
17 スーパークラス	-0.0733
3 ポリモーフィズム	0.0693
20 パッケージ	-0.0689

学習段階③の第一因負荷量

知識項目	第一因子
13 メソッド	0.9520
4 インヘリタンス	0.9245
6 機能	-0.8980
12 インスタンス	0.8841
17 スーパークラス	0.8575
11 クラス	0.8291
18 クラスメソッド	0.8291
8 属性	0.7346
1 アイデンティティ	0.6856
10 スーパークラス	0.6856
16 インスタンス化	0.6581
14 クラスライブラリ	-0.4573
15 インスタンス変数	0.2273
2 分類	-0.1985
5 属性	-0.1729
9 クラス	0.1210
19 インタフェース	0.1189
3 ポリモーフィズム	0.1094
20 パッケージ	0.0914
7 カプセル化	-0.0806

図 7 学習段階② から学習段階③ への因子負荷量の変化(2)

ことを意味していると考えられる. また学習段階② の第一因子を抽出した知識項目⑲と直近の知識項目⑳（因子負荷量 0.791）に加え，第二因子を抽出した知識項目⑨は, 学習段階③ の第二因子を抽出した知識項目として現れている. よって学習段階③ の第二因子は，学習段階② の 2 つの因子の中心的な部分が収束して形成されたと考えられる.

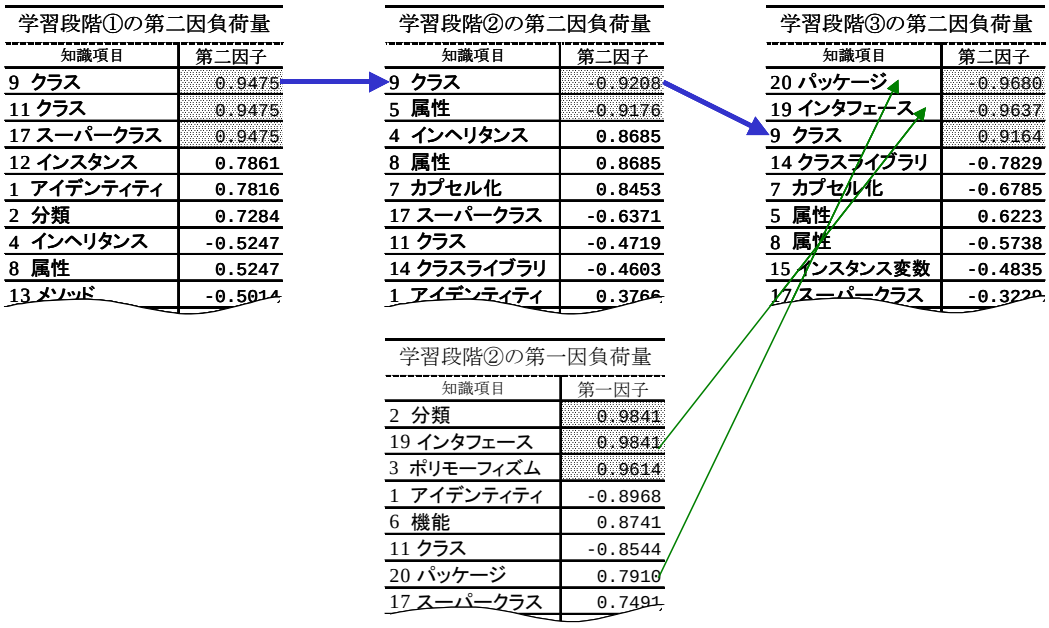


図 8 第二因子における因子負荷量の変化

3.4　因子の遷移による理解状態の考察

学習段階①，学習段階②，学習段階③ において抽出された因子と，因子負荷量から観察した因子の遷移(a), (b), (c), (d), (e)を矢印 a, b, c, d, e にして図 9 に示す．

遷移(a) では，学習段階① の第二因子「オブジェクト指向の基礎理解 (1)」が学習段階② の第二因子「オブジェクト指向の基礎理解 (2)」として遷移し，因子の中心的な部分が変わっていない．一方，学習段階① の第一因子，第三因子は，学習段階の進行による因子の遷移が観察できない．学習段階① における第一因子と第三因子は，プログラミング学習によって新しい知識が理解できたことで，異なる因子に変化して消滅したと考えられる．また，学習段階① の第一因子「回答番号の順番押しの傾向（正解番号が後半)」と，第三因子「回答番号の順番押しの傾向（回答番号が前半)」は，理解できない項目つまり難しい問題への対応であると考えられる．

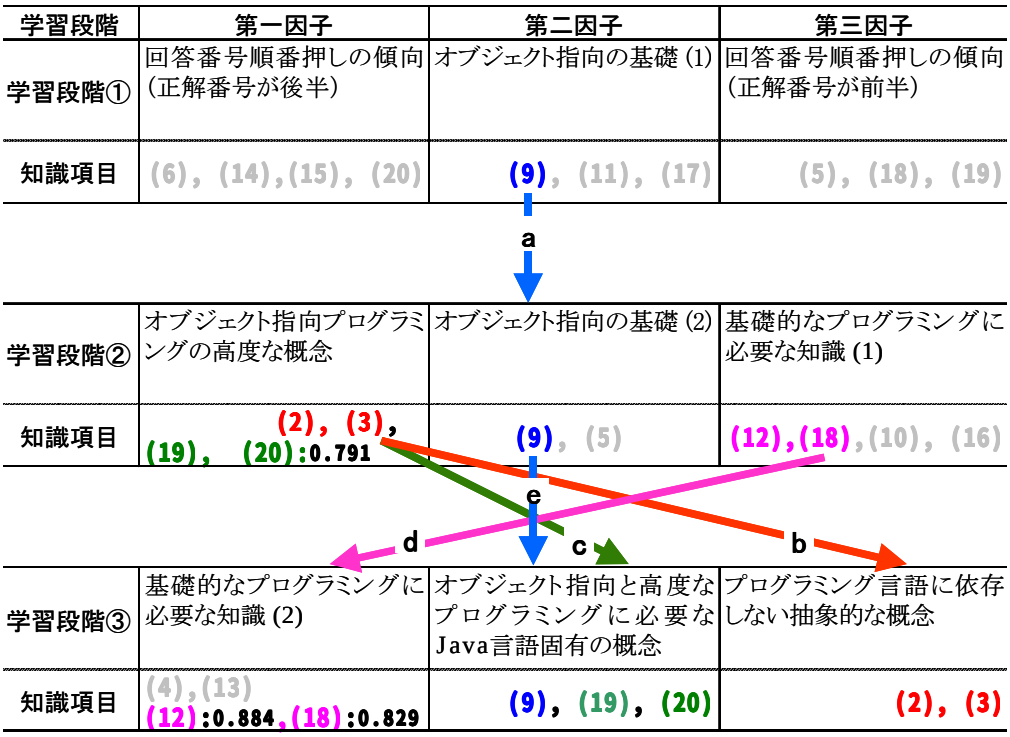


図 9　学習段階①~③における因子の遷移

遷移(b), (c) では, 学習段階② の第一因子「オブジェクト指向プログラミングの高度な知識」が, 学習段階③ の第二因子「オブジェクト指向と高度なプログラミングに必要な Java 言語固有の概念」と第三因子「プログラミング言語に依存しない抽象的な概念」に分かれて遷移している. 学習段階③ における Java 言語のプログラミング演習によって学習者が ②分類や ③ポリモーフィズムといった新しい知識を理解できたと考えられる. よってプログラミングに必要な Java 言語固有の概念と, プログラミングに直接影響しない概念に分けて理解できたことで因子の遷移が生じたと考えられる.

表 4 において Java 言語固有の概念は, 知識項目⑲（インタフェース）と⑳（パッケージ）と考えられる. 遷移(c) では, これら知識項目は学習段階② において第一因子を抽出した知識項目とその直近の知識項目として表れ, 学習段階③では第二因子に表れている. これはプログラミング演習によって Java 言語の新しい概念が理解できた結果, 学習段階② の「オブジェクト指向プログラミングの高度な概念」と「オブジェクト指向の基礎理解」といった因子に「Java 言語固有の概念」が加えられ, 学習段階③ の第二因子「オブジェクト指向と高度なプログラミングに必要な Java 言語固有の概念」に遷移したと考えられる.

遷移(d) では, 学習段階② の第三因子「基礎的なプログラミングに必要な知識 (1)」が, 学習段階③ の第一因子「基礎的なプログラミングに必要な知識 (2)」に遷移している. 学習段階③ のプログラミング演習によって java 言語のプログラミングを経験した結果, オブジェクト指向に関する理解が向上したと考えられる. よってオブジェクト指向に関する因子の影響が小さくなり, 学習段階② の第三因子が同じ因子のまま影響力の大きな学習段階③ の第一因子へ遷移したと考えられる.

3.5 まとめと今後の課題

プログラミング言語の理解状態は, 配布資料と説明による学習によって向上したが, プログラミング言語固有の概念に関する理解状態の向上は見られなかった. 学習段階③ のプログラミング演習を行った後では, それぞれの因子が遷移してプ

ログラムに関する理解状態に向上が見られた．これら知見から新たにプログラミング言語を学ぶ学習段階② においては, その学習に関する新しい因子が形成される．そして，その因子はプログラミング演習によって「基礎的なプログラミングに必要な知識」,「その言語固有の概念」，プログラミングに必要のない「プログラミング言語に依存しない抽象的な概念」へと遷移することが解明できた.

本実験の被験者にとって Java 言語は初めて学習するプログラミング言語であるが，他のプログラミング言語におけるプログラミング経験を有している．よって学習段階①，学習段階② の第二因子として抽出された,「オブジェクト指向の基礎理解 (1)」に関する知識項目は，他のプログラミング言語の学習で習得していたと考えられる．一方，他の因子に関する知識項目は初めて学習したため，学習の進行による知識の増加や理解の深まりによって因子の遷移が生じたと考えられる.

プログラミング言語固有の概念がプログラミング演習によって理解されることを解明したことで，プログラミング演習によってプログラムの理解状態に変化が生じることが確認できた．よってプログラミング言語の学習においては，プログラミング言語の関連知識とプログラミング言語固有の概念に分類し，前者はテキストや解説で学習させ，後者はプログラミング演習によって理解させることで効率的なプログラミング教育を実現できると考えられる.

本章で提案した解析手法は，効率的な教育形態の形成を目的とした学習者の理解状態の把握に有効であると考えられる．また本知見はプログラミング言語の教育システム開発における学習者モデル構築に有用と考えられる．今後は専攻分野が異なる被験者を対象にした実験と解析を行い，理解状態の詳細な知見を適用したプログラミング教育を実現したいと考えている.

４．Java 言語における順序関係の統計的分析手法

4.1　あらまし

算数や理科などの教科では，知識間に順序関係[1]（ある知識の理解に他の知識の理解を必要とする関係）が存在する場合が多く，知識間の順序関係に基づく教育手順の体系化がなされている．このような体系化はプログラミング言語教育においても重要である．プログラミング言語に関する知識間の順序関係が解明されれば，体系的な教育が実現されて効果的な教育が期待できる[42]．例えば，プログラミング言語の教育用著書の場合，学習者が効率的に習得できる順序で学習項目を配置できる．しかし，現状では Java 言語に関する著書[24],[27],[28],[36]の目次編成はそれぞれに異なっており，専門家の間でコンセンサスが得られた体系が存在しない．長谷川らの，初心者のプログラミング学習を対象にしたプログラム制御構造のイメージと理解度に関する研究[6]や，吉川の視線追跡装置や脳波測定装置を使ったプログラムの理解過程の研究[56]など，プログラミング言語における知識間の順序関係に関する研究が行われているが Java 言語の理解に関する研究は見あたらない．

知識間の順序関係を導出する方法の一つとして，テストを行い任意の 2 つのテスト結果を比較すれば，その項目に含まれる知識の順序関係が導出できる．例えば，加算と乗算のテストを行ってその結果を比較すれば，加算のテスト項目を間違い，乗算のテスト項目を正解する学習者は稀である．このような結果から，加算から乗算への順序関係が導出できる．知識間の順序関係を解明する方法の一つに順序理論[1]が提案されている．順序理論は，多数の学習者に対してテストを行い，テスト結果から順序関係を導出する．

P.W.Airasian らは，順序理論により 6 つのテスト項目の順序関係を解析している．伊藤らは，ファジィ理論を用いて順序理論を拡張したファジィ項目関連構造分析法（Fuzzy item relational structure analysis: FIRS 分析）を提案し[9]，8 つのテスト項目の順序関係を解析している．学習者がテストに回答する際,「まぐれ当たり」や「うっかりミス」はやむをえない事象であり，テスト結果の分析に

おいていわゆる「まぐれ当たり」や「うっかりミス」を考慮しなければ，正しい順序関係が得られない．先行研究[1],[9]では，回答で生じる「まぐれ当たり」と「うっかりミス」が考慮されずに知識間の順序関係が導出されるため，順序関係から正確な理解順序を得られない．また，先行研究[1],[9]の方法を適用してテスト項目を増やした場合，順序関係の全体構造を表した関連構造が複雑になり学習者の理解順序の把握が困難になる．

本章では，順序理論の仮説に対し矛盾とみなされる回答が生じる要因として，学習者の回答におけるいわゆる「まぐれ当たり」と「うっかりミス」に注目した．Java 言語の知識間における順序関係を導出するための一方法として，テストで生じるいわゆる「まぐれ当たり」と「うっかりミス」の回答を考慮した統計的分析手法を提案する．統計的分析手法は，P.W.Airasian らの Ordering Theory（順序理論）のモデルを適用し，いわゆる「まぐれ当たり」と「うっかりミス」の回答を二項分布の分布関数を用い統計的に補正して順序関係を導出する．また関連構造は，それぞれの知識項目に因子分析を適用することによってその構造を簡単化して構築する．

4.2 順序理論の基本概念

順序理論は，学習者が修得すべき 2 種類の知識が存在する場合，多数の学習者にテストを行い，その知識間に順序関係が存在するか否かをテスト結果から解明する方法の一つである．

順序理論では，あるテスト項目 Y が，他の項目 X の理解を必要条件として達成されると仮定する．テスト項目 X とテスト項目 Y の達成（正解）を ”correct”，非達成（間違い）を ”error” とすると，n 人の学習者は以下の集合 M_{00}, M_{01}, M_{10}, M_{11} のいずれかに割り当てられる．これら集合に含まれる学習者数を，$N_{00}, N_{01}, N_{10}, N_{11}$ として 2×2 分割表に示せば表 8 のようになる．ある知識の理解に他の知識の理解を必要とする関係を，順序関係と呼ぶ．

① 集合 M_{00} は，テスト項目 X が ”error”，テスト項目 Y も ”error” になる学習者の集合で，その学習者数は N_{00} になる．

表 8 テスト項目 X / Y におけるの達成／非達成 の 2×2 分割表

		test item X		total
		error	correct	
test item Y	error	N_{00}	N_{10}	$N_{00}+N_{10}$
	correct	N_{01}	N_{11}	$N_{01}+N_{11}$
total		$N_{00}+N_{01}$	$N_{10}+N_{11}$	n

② 集合 M_{01} は，テスト項目 X が ”error”，テスト項目 Y が ”correct” になる学習者の集合で，その学習者数は N_{01} になる．

③ 集合 M_{10} は，テスト項目 X が ”correct”，テスト項目 Y が ”error” になる学習者の集合で，その学習者数は N_{10} になる．

④ 集合 M_{11} は，テスト項目 X が ”correct”，テスト項目 Y も ”correct” になる学習者の集合で，その学習者数は N_{11} になる．

テスト項目 X を間違い，テスト項目 Y を正解する集合 M_{01} は，順序理論の仮説に矛盾する．順序理論においては，集合 M_{01} に含まれる学習者数 N_{01} の全体に対する割合が小さければ順序関係が成立するとされている．集合 N_{01} の全体に対する割合を示す順序係数 ε は式(1)で表され，順序係数 ε が小さければ順序関係の存在を意味する．

$$\varepsilon = \frac{N_{01}}{n} \qquad (1)$$

順序理論のモデルでは，全てのテスト項目から任意の 2 つのテスト項目を取り出し，それぞれのテスト項目の ”correct” と ”error” の数値を 2×2 分割表に割り当て，集合 M_{01} の割合から順序関係を導出する．例えば，任意の 2 つのテスト項目，「テスト項目 i」，「テスト項目 j」を取り出す．「テスト項目 i」が「あるテスト項目 Y」，「テスト項目 j」を「他のテスト項目 X」として 2×2 分割表に割り当て，集合 M_{01} の割合から「テスト項目 i」から「テスト項目 j」への順序関係を導出する．さらに，「テスト項目 i」が「他のテスト項目 X」，「テスト項目 j」を「あるテスト項目 Y」とした 2×2 分割表の集合 M_{01} の割合から，「テスト項目 j」か

ら「テスト項目i」への順序関係を導出する．2 つのテスト項目間には，「テスト項目i」から「テスト項目 j」，「テスト項目 j」から「テスト項目i」の 2 つの順序関係が導出できる．順序理論における順序関係の導出手順を図 10 に示し，そのモデルの特長を以下に示す．

① 順序関係は，任意に取り出した 2 つのテスト項目の双方向において導出されるので，テスト問題の難易度や出題順序に影響されない．

② テスト項目 X, Y の正解率に有意な差が無い事象においても，任意の 2 つのテスト項目 X，Y の ”correct” と ”error” の数値を 2×2 分割表に割り当てるため，テスト項目間の順序関係の導出が可能である．

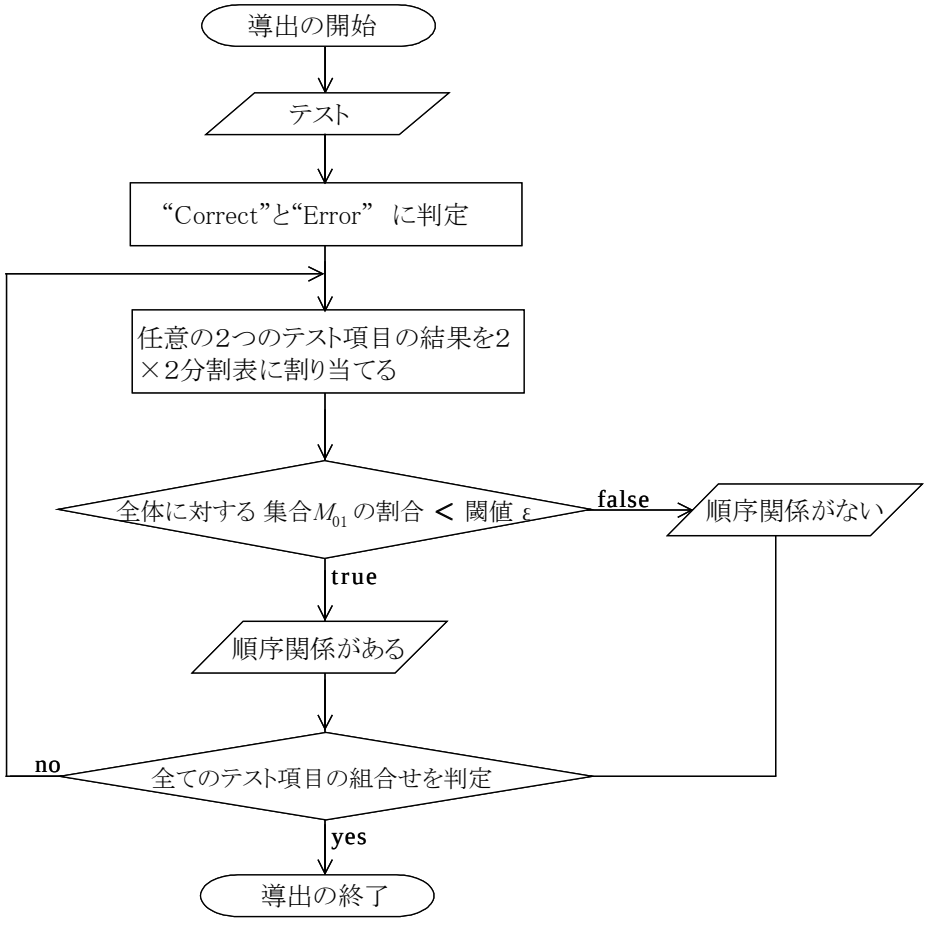


図 10　順序理論における順序関係の導出手順

4.3 Java 言語の順序関係の解析手法

Java 言語の知識やソースコードには, 算数や理科の知識のように, 基礎的で独立した知識と，他の知識に関係した知識が存在し[43]，順序理論のモデルが適用できると考えられる．本章では，P.W.Airasian らの順序理論のデルを適用した．順序理論においては, 集合 M_{01} の学習者数 N_{01} の全体に対する割合が小さければ, テスト項目 X とテスト項目 Y に順序関係が存在するとされているが, 多岐選択法（Multiple-choice Method）による回答では，いわゆる「まぐれ当たり」による正解が生じる．また，回答時の「うっかりミス」により回答を間違うことがあるが，順序理論ではこれらの要素事象（elementary event）が考慮されずに順序関係が導出される．本章では，いわゆる「まぐれ当たり」と「うっかりミス」によって生じる回答を統計的に補正し，順序関係を導出する統計的分析手法を提案する．また，それぞれの知識項目に因子分析を適用することにより，その関連構造を簡単化する．

本章では，あるテスト項目の正解に含まれる知識を「知識項目」，いくつかの知識項目を集約した一つのグループを「クラスタ」，順序理論に矛盾が生じる集合 M_{01} に遷移した学習者を「矛盾学習者」, 集合 M_{01} に矛盾学習者が出現する確率を「矛盾率」, 知識項目やクラスタにおける順序関係の全体構造を「関連構造」と呼ぶ.

4.3.1 順序関係による理解順序の解析過程

矛盾率によって導出された順序関係とクラスタに集約した関連構造による, Java 言語の理解順序の解析過程を以下に示す.

① 矛盾関数 P に,「まぐれ当たり」の確率 f と,「うっかりミス」の確率 c を適用して矛盾率を算出する.

② 知識項目の関連構造を，ある定数より小さい矛盾率から導出された順序関係によって構築する.

③ テスト結果の集計データを因子分析して知識項目を因子負荷量からいくつかのクラスタに集約し，その因子も抽出する.

因子分析は，一組の多変量データを比較的少数の共通な潜在変数の線型結合とし，集約的に表すことを目的としたモデルである．因子負荷量は新しい概念の抽出や変数の集約に適用できるので，本章においても知識項目の集約に因子負荷量を用いる．

④ 解析過程②で構築した知識項目だけの関連構造では，順序関係を示す矢線の数が多すぎて関連構造が複雑になり，関連構造から理解順序を把握できない．そこで，知識項目を集約したクラスタ間の順序関係に変換し，その関連構造を簡単化して Java 言語の理解順序を解析する．

4.3.2 順序理論のモデルを適用した矛盾関数の提案

本章における順序関係は，二項分布の分布関数に「まぐれ当たり」や「うっかりミス」の出現率を適用した矛盾率を提案して解析する．2 つの性質や出現割合などは 2 項分布に従っており[17],「まぐれ当たり」と「うっかりミス」もこれに相当するので，順序関係の導出に 2 項分布を用いた．矛盾率は，偶然の成功と失敗が生じている条件の下で, 順序理論に矛盾する集合 M_{01} に矛盾学習者が出現する確率である．順序理論では，表 8 の集合 M_{01} における学習者数 N_{01} の全体に対する割合がある定数よりも小さいと順序関係が存在するとしている．矛盾率が小さければ，順序理論に矛盾する集合 M_{01} に出現する学習者数 N_{01} が小さくなり，順序理論の仮説に適合する．したがって，ある定数より小さい矛盾率を示す知識項目の組み合せを取り出せば順序関係が導出できる．

表 9 の 2×2 分割表においてテスト項目 X からテスト項目 Y に順序関係が存在するとすれば，集合 M_{01} に出現する N_{01} 人の学習者の大部分は,「まぐれ当たり」か「うっかりミス」といった偶然の成功か失敗により，集合M_{00} と M_{11} から集合 M_{01} に遷移したと考えられる．ただし集合M_{10}から集合 M_{01} への遷移は,「まぐれ当たり」と「うっかりミス」が連続して生じる事象で，出現確立が小さいため本提案手法では考慮しない．本提案手法では，表 9 の 2×2 分割表の集合 M_{01} に出現する N_{01} 人の学習者を，グループ S_0 とグループ S_1 の 2 つのグループに分類する．グループ S_0 はテスト項目 Y を「まぐれ当たり」して集合 M_{00} から M_{01}

表9 「まぐれ当たりと」と「うっかりミス」により回答が遷移した2×2 matrix

		test item X	
		error	correct
test item Y	error	N_{00} S_0	N_{10}
	correct	k_0+e_0 + k_1+e_1	N_{11} S_1


「まぐれ当たり」が含まれるグループS_0
「うっかりミス」が含まれるグループS_1


：テスト項目Yの「まぐれ当たり」による回答の遷移
：テスト項目Xの「うっかりミス」による回答の遷移

S_0 ：テスト項目Yを「まぐれ当たり」して集合M_{00}からM_{01}への遷移を含む
k 人のグループ

S_1 ：テスト項目Xを「うっかりミス」して集合M_{11}からM_{01}への遷移を含む
$(N_{01}-k)$ 人のグループ

e ：集合M_{01} において「まぐれ当たり」や「うっかりミス」がない学習者数,
$e = e_0 + e_1$, $0 \leqq e \leqq N_{01}$

e_0 ： e の中でグループS_0に含まれる学習者数, $0 \leqq e_0 \leqq k$

e_1 ： e の中でグループS_1に含まれる学習者数, $0 \leqq e_1 \leqq N_{01}-k$

k_0 ：「まぐれ当たり」によってテスト項目Yを正解する学習者数

k_1 ：「うっかりミス」によってテスト項目Xを間違う学習者数

への遷移を含むk人のグループである．グループS_1はテスト項目Xを「うっかりミス」して集合M_{11}からM_{01}への遷移を含む$(N_{01}-k)$人のグループである．集合M_{01}において「まぐれ当たり」や「うっかりミス」がない状態の学習者数を"e"で$e=e_0+e_1$とし，e_0はグループS_0に含まれ$0\le e_0\le k$，e_1がグループS_1に含まれ$0\le e_1\le N_{01}-k$とする．グループS_0において，テスト項目Yを「まぐれ当たり」して集合M_{00}からM_{01}へ遷移する学習者数をk_0人とすれば$k_0=k-e_0$となる．e_0の存在より「まぐれ当たり」で遷移する学習者数は，$0,1,2,\cdots\cdots,k-1,k$となり定数でなく最大出現人数がk人になる．グループS_1においても，テスト項目Xの「うっかりミス」により集合M_{11}からM_{01}に遷移する学習者数をk_1人とすれば，$k_1=(N_{01}-k)-e_1$となる．e_1の存在により「うっかりミス」で遷移する学習者数は$0,1,2,\cdots\cdots,(N_{01}-k)-1,N_{01}-k$となり，定数でなく最大出現人数が$(N_{01}-k)$人になる．

2項分布における一般的な標本の抽出数は測定者が決定しているが，「まぐれ当たり」や「うっかりミス」の出現人数は，学習者の状態によって出現状況が変わるため測定者が出現人数を決定できない．よって「まぐれ当たり」と「うっかりミス」による出現人数は，順序理論における学習者数N_{01}の全体に対する割合が小さいと順序関係が存在するとした仮定において，定数とせずに最大出現人数として扱い，前者による最大出現人数はk人，後者による最大出現人数を$(N_{01}-k)$人とした．

「まぐれ当たり」により集合M_{00}からM_{01}に遷移した矛盾学習者の最大人数はk人になり，kの範囲は$0\le k\le N_{01}$になる．また，「うっかりミス」で集合M_{11}からM_{01}に遷移した矛盾学習者の最大人数は$(N_{01}-k)$人になり，その範囲は$0\le N_{01}-k\le N_{01}$になる．矛盾学習者において，「まぐれ当たり」が出現率f，「うっかりミス」が出現率cで，いずれかが生じていると仮定すると，提案手法のアルゴリズムは以下①～④のようになる．提案手法による矛盾率の計算を式(2)に示す．

式(2)の$P(N_{00},N_{01},N_{11},f,c)$は，矛盾学習者が出現する平均確率で，$P$を「矛盾関数」，算出された値を「矛盾率」と呼ぶ．矛盾関数Pに含まれる$B(k;N_{00}+k,f)B(N_{01}-k;N_{11}+N_{01}-k,c)$は，「まぐれ当たり」と「うっかり

ミス」による矛盾学習者が 0 人からN_{01}人までの，それぞれの人数において集合M_{01}に出現する確率である． $B(k;N_{00}+k,f)B(N_{01}-k;N_{11}+N_{01}-k,c)$ は$N_{01}+1$個算出され，「単独矛盾率」と呼ぶ.

① $N_{00}+k$ 人の事象において「まぐれ当たり」による矛盾学習者の最大人数をk人 とすれば，分布関数 Bから，テスト項目 Y に「まぐれ当たり」が出現率 f でk 人出現する累積確率は $B(k;N_{00}+k,f)$ になる.

② まぐれ当たり」による矛盾学習者の最大人数をk 人 とすると，「うっかりミス」による矛盾学習者の最大人数は $N_{01}-k$ 人になる． $N_{11}+N_{01}-k$ 人の事象において分布関数 Bから，テスト項目 X に「うっかりミス」が出現率 f で$N_{01}-k$ 人出現する累積確率は $B(N_{01}-k;N_{11}+N_{01}-k,c)$ になる.

③ 矛盾学習者は，「まぐれ当たり」か「うっかりミス」のいずれかにより集合M_{01}に遷移したと仮定すると，集合M_{00}からM_{01}に遷移した矛盾学習者の最大人数が k の時，単独矛盾率は①，②の積 $B(k;N_{00}+k,f)B(N_{01}-k;N_{11}+N_{01}-k,c)$ になる.

④ k の範囲が $0\le k\le N_{01}$ であるので，k が $0,1,2,\cdots\cdots N_{01}-1,N_{01}$ における単独矛盾率を算出する.

⑤ 「まぐれ当たり」と「うっかりミス」による出現人数を最大出現人数として扱うので，$N_{01}+1$ 個の単独矛盾率の累積平均 $\dfrac{1}{N_{01}+1}\sum_{k=0}^{N_{01}}B(k;N_{00}+k,f)B(N_{01}-k;N_{11}+N_{01}-k,c)$ が矛盾率になる.

$$P(N_{00},N_{01},N_{11},f,c)=\frac{1}{N_{01}+1}\sum_{k=0}^{N_{01}}B(k;N_{00}+k,f)B(N_{01}-k;N_{11}+N_{01}-k,c) \quad \textbf{(2)}$$

ただし，$B(k;n,q)=\sum_{x=0}^{k}\binom{n}{x}q^x(1-q)^{n-x}$

f ：テスト項目 Y における「まぐれ当たり」の出現率

c ：テスト項目 X における「うっかりミス」の出現率

$B(k;n,q)$ ：二項分布の分布関数（確率qで出現する事象がn人のうち最大 k 人出現する確率）

4.4　矛盾関数の適用例

図 11 から図 14 は，集合 M_{01} で $N_{01}=5$ 時の $k=0,1,2,5$ における，矛盾学習者の出現率を表している．図の目盛りは，縦軸がテスト項目 Y における「まぐれ当たり」による矛盾学習者数，横軸がテスト項目 X における「うっかりミス」による矛盾学習者数を表している．縦軸の目盛り y_i の長さは「まぐれ当たり」が確率 f で，$N_{00}+k$ 人のうち i 人の矛盾学習者が集合 M_{01} に出現する確率で，原点から目盛り k までの距離は矛盾学習者が最大 k 人出現する確率になる．同様に，横軸の目盛り x_j の長さは「うっかりミス」が確率 c で，$N_{11}+N_{01}-k$ 人のうち j 人の矛盾学習者が集合 M_{01} に出現する確率で，原点から目盛り $N_{01}-k$ までの距離は矛盾学習者が最大 $N_{01}-k$ 人出現する確率になる．，

縦軸の ”0” から ” k ” までの線分と，横軸の ”0” から”$N_{01}-k$ ”の線分で囲まれた領域は，「まぐれ当たり」による矛盾学習者数が k 人，「うっかりミス」による矛盾学習者が $N_{01}-k$ 人である時の単独矛盾率を表している．

図 11 は $k=0$ で，横軸の目盛り ”0” から”5”までの距離は，

$$B(N_{01}-k;N_{11}+N_{01}-k,c)=\sum_{x=0}^{N_{01}}\binom{N_{11}+N_{01}}{x}c^x(1-c)^{(N_{11}+N_{01})-x}$$

縦軸の”0”の距離が

$$B(k;N_{00}+k,f)=\sum_{x=0}^{0}\binom{N_{00}}{x}f^x(1-f)^{N_{00}-x}$$

になる．前者と後者で囲まれた面積が，この事象における単独矛盾率を表し $B(0;N_{00},f)B(N_{01};N_{11}+N_{01},c)$ になる．

図 12 は $k=1$ の事象で，横軸の目盛り ”0” から”4”の距離が

$$B(N_{01}-k;N_{11}+N_{01}-k,c)=\sum_{x=0}^{N_{01}-1}\binom{N_{11}+N_{01}-1}{x}c^x(1-c)^{(N_{11}+N_{01}-1)-x}$$

になり，縦軸の ”0” から”1”までの距離は

$$B(k;N_{00}+k,f)=\sum_{x=0}^{1}\binom{N_{00}+1}{x}f^x(1-f)^{(N_{00}+1)-x}$$ になる．

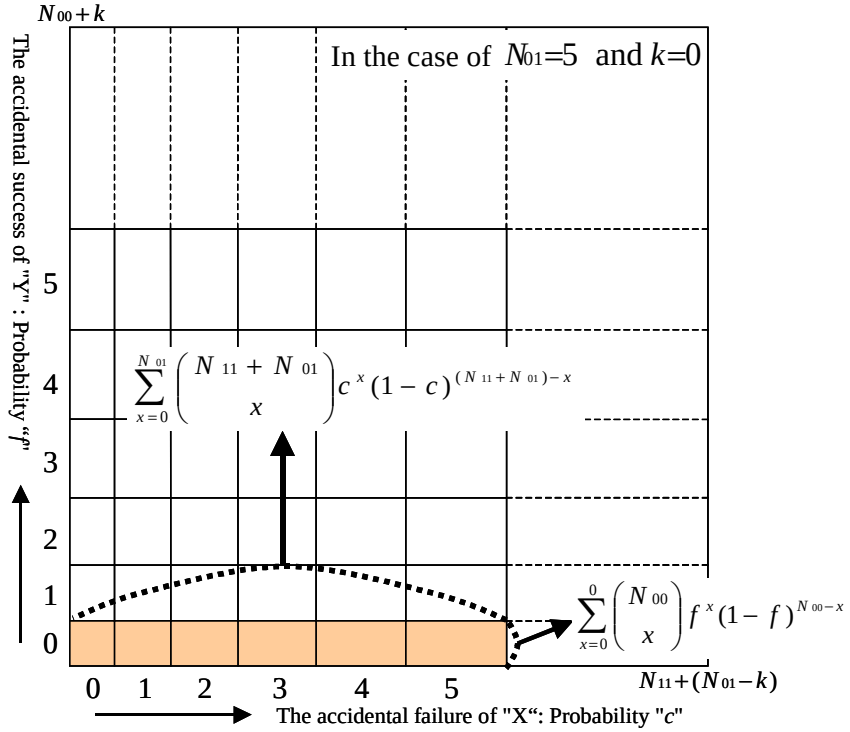
図 13 は $k=2$ の事象で，「まぐれ当たり」による矛盾学習者が 2 人，「うっか

りミス」による矛盾学習者が$N_{01}-k$人で，▪▪▪▪で囲まれた面積が$B(2;N_{00}+2,f)B(N_{01}-2;N_{11}+N_{01}-2,c)$になり，単独矛盾率を表す.

図14の$k=5$の事象では$N_{01}=k$になり,「うっかりミス」により矛盾学習者数が出現する事象はN_{11}人になる.

したがって，$k=0,1,2,3,4,5$における単独矛盾率の累積平均が，$N_{01}=5$の時の矛盾率になる.

例えば，100人の学習者においてテスト項目Xの正解が100人，テスト項目Yの間違いが100人になるような事象では，テスト項目Xの知識によってテスト項目Yが一つとして正解していない．このような事象では，順序関係が存在しないと考えられるが，順序理論では集合M_{01}の学習者数N_{01}が“0”になり順序係数εとして“0”が算出され，間違った順序関係が導出される．このような回答が偏った事象についても，矛盾関数Pによれば$B(k;N_{00}+k,f)=1$，$B(N_{01}-k;N_{11}+N_{01}-k,c)=1$になり，矛盾率に”1”が算出されて間違った順序関係が導出されない.



k: The number of times of "the accidental success" of test item "Y"

図 11　$k=0$ の単独矛盾率

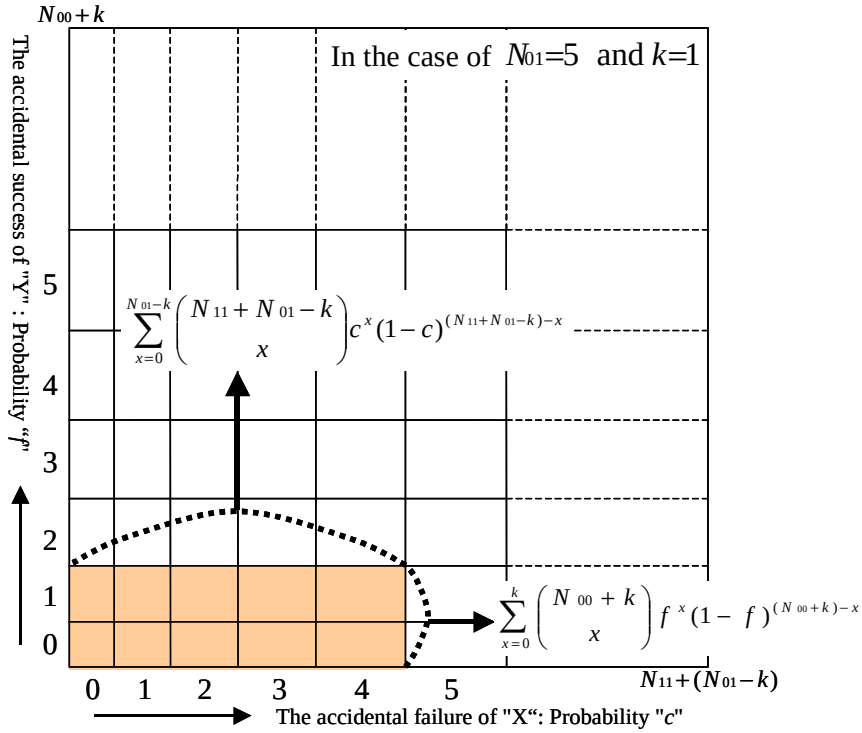


図 12　$k=1$ の単独矛盾率

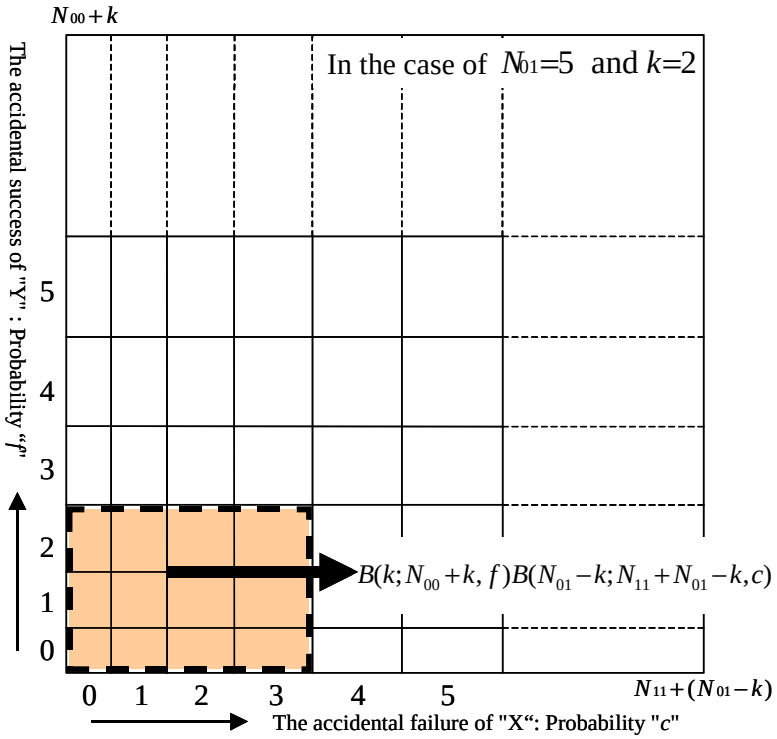


図 13　k = 2 の単独矛盾率

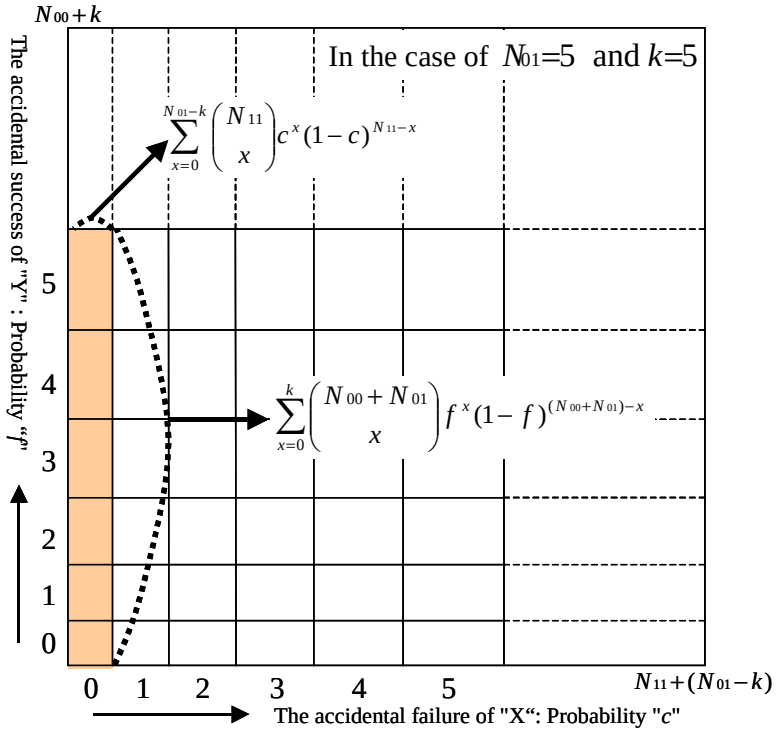


図 14　k = 5 の単独矛盾率

4.5　クラスタへの集約と関連構造の簡単化

3.1 節の解析過程④における，①知識項目を集約したクラスタ間の順序関係への変換と，②その関連構造を簡単化する手法を以下に示す.

① 25 のテスト項目のデータを因子分析し, その因子負荷量から以下の方法でいくつかのクラスタに集約する.

- 知識項目 j の第一因子から第 n 因子において, 因子負荷量 a_{jx} の絶対値が最大になる第 x 因子を選択する.
- 知識項目 j を，第 x 因子のグループとしてクラスタ f(x) に集約する.

⑥ 順序関係を表す矢線が多くなると理解順序が把握しづらいので，以下の方法に従って関連構造を簡単化する.

- 複数・同方向の順序関係の簡単化

  クラスタ f(x) の任意の知識項目からクラスタ f(y) の任意の知識項目への順序関係が複数存在する場合，それらをクラスタ f(x) からクラスタ f(y) への順序関係とし，図 15 のように１つの矢線で示す.

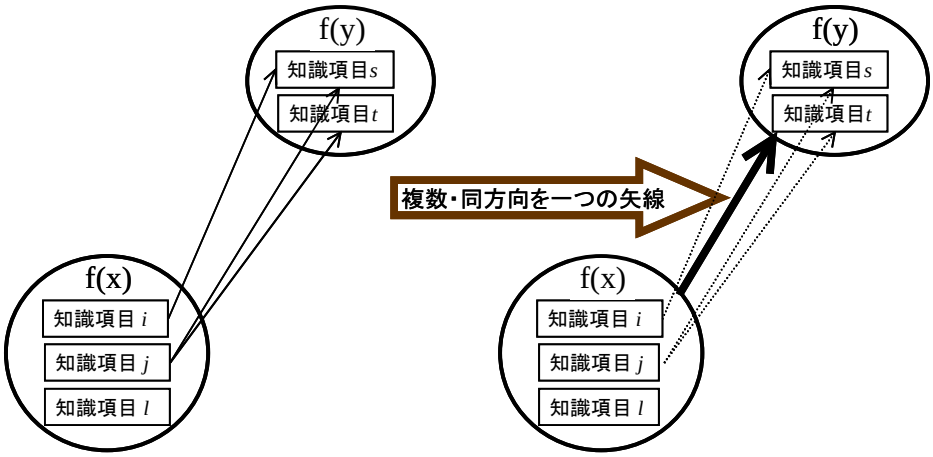


図 15　複数・同方向の順序関係の簡単化

・ 双方向の順序関係の簡単化

クラスタ f(x) の知識項目からクラスタ f(y) の知識項目への順序関係, および, クラスタ f(y) の知識項目からクラスタ f(x) の知識項目への順序関係が両方向に存在する場合, 矛盾率が小さい方向は矛盾学習者が少なく, より多くの学習者の理解順序に従っていると考えられるので, それぞれの矛盾率の平均値が小さい方向の順序関係のみを図 16 のように一方向の矢線で示す.

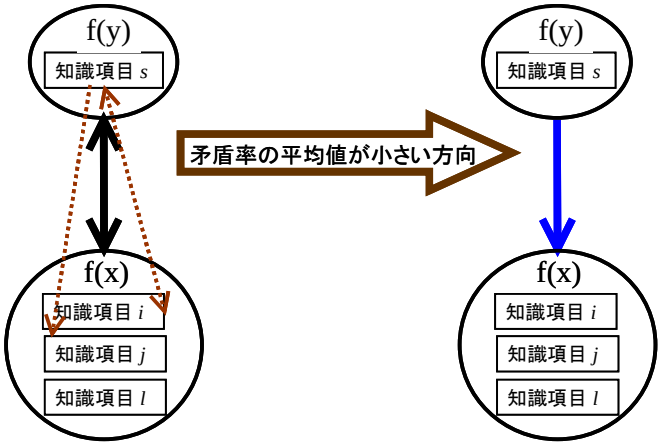


図 16　双方向の順序関係の簡単化

・ 並行する順序関係の簡単化

クラスタ f(x) からクラスタ f(y) への順序関係, クラスタ f(y) からクラスタ f(z) への順序関係, および, クラスタ f(x) からクラスタ f(z) への順序関係が存在する場合, クラスタ f(x) からクラスタ f(z) への順序関係は, クラスタ f(y) を経由して間接的に示されていると考えられるので, クラスタ f(x) からクラスタ f(z) への順序関係を示す矢線を図 17 のように消去する.

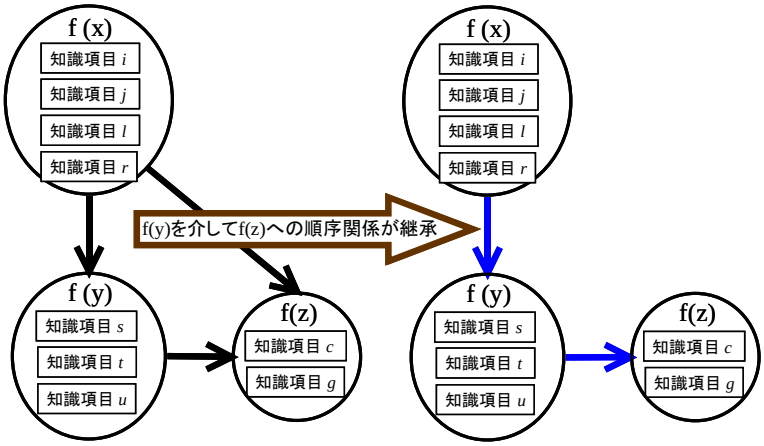


図 17　並行する順序関係の簡単化

4.6　まとめと今後の課題

本章では，順序理論に矛盾した回答がいわゆる「まぐれ当たり」と「うっかりミス」によって生じると仮定し，これら要素事象を統計的に補正して順序関係を導出する統計的分析手法を提案した.「まぐれ当たり」や「うっかりミス」の回答の遷移による間違った順序関係を補正するため，二項分布の分布関数に「まぐれ当たり」と「うっかりミス」の出現率が適用された矛盾率を提案した．また関連構造の解析では，知識項目だけの複雑な関連構造を，因子負荷量によって知識項目がクラスタに集約された関連構造に簡単化する手法も提案した.

本提案手法は，結果がテスト問題に依存し因子分析において因子の意味を抽出する必要もあるため，本稿の成果は Java 言語特定の領域に基づいている．しかし，テスト問題の作成と因子分析における因子の意味付けさえできれば，適用領域を広げることは可能と考えられる．また，クラスタに含まれる知識項目が同じ特性の因子であることから，一つのクラスタに含まれる知識項目を一つの単元とした教育や学習の体系化が考えられる．知識項目を集約したクラスタの関連構造は，教育システムの開発において必要となる学習者モデルの構築に有用であると考えられる.

５．Java 言語の理解順序の解析

5.1　あらまし

本章では,「まぐれ当たり」や「うっかりミス」を考慮した提案手法によって Java 言語の知識間の順序関係を導出し, 順序関係の全体の構造を表した関連構造から理解順序を解析する．Java 言語の順序関係を導出するため 2 グループ（4 クラス）の 66 名に，クラス単位で講義・演習を行いテストを実施した．テストは，Java 言語の関連知識（method，instance，instance variable，package，super class，sub class，interface など）が多岐選択式で 15 問，プログラムの穴埋めと出力結果の回答が短答式でそれぞれ 5 問である，学習者のテスト結果から Java 言語の知識間の順序関係を提案手法によって導出し，知識項目だけの複雑な関連構造を因子分析の適用によって簡単化して構築した．実験結果の解析では，25 の知識項目から 104 の順序関係を導出し，その知識項目が 9 つのクラスタに集約された関連構造を構築して Java 言語の理解順序を解明した．関連構造から 3 つの学習プロセスを設定し，効率的な Java 言語の学習プロセスを導出した．また導出した学習プロセスと，Java 言語を解説した著書[24],[27],[28],[36]における知識項目の出展順序との比較も行った.

本章の特徴は，今までの経験的知見によって推測されていたであろう Java 言語の順序関係を実験によって解析した点，解析において統計的分析方法を用いた点，簡単化された関連構造からソフトウェア工学教育の一つであるプログラミング教育の体系化の促進が期待できる点である.

5.2　理解順序の測定実験

5.2.1　実験の環境

プログラミング言語に対する習熟の差異，テスト問題の出題順序などが及ぼす影響を考慮し，Java 言語の初学者を対象にし，実験環境とテストの内容的妥当性を以下のように設定した．

① 理解状態の測定は，学習環境の影響を考慮して実験環境の異なる 2 グループで行った．これらグループを 2 つのクラスに分け，クラス単位で講義・演習とテストを実施した．2 グループ（4 クラス）における学習者の総数は 66 名である．実験結果から学習者の能力の分布を，正解率で表したヒストグラムを図 18 に示す．正解率は最低 20%，最高 100%で平均 71%である．本提案手法においては，学習者の能力のばらつきが大きければ多くの順序関係を得られるので望ましが，均等に能力が分布している必要はない．

② それぞれのクラスでの講義・演習は，教授者の教育方法の違いによる学習効果の差異をなくすため１人の教授者が行った．

③ テスト問題の出題順序は，正解の記号の並びなどを考慮してクラスごとに無作為に変更した．ただしテスト問題の作成時には，4.2 節における順序理論の“特長 ①”，“特長 ②”からテスト項目の難易度や正解率による出題順序は考慮していない．

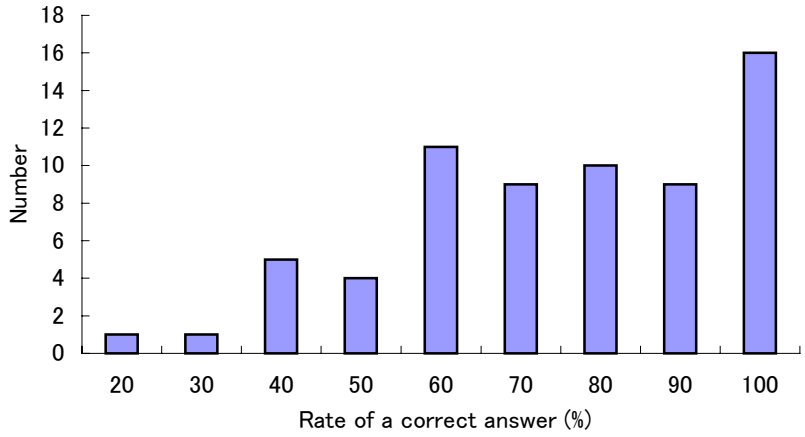


図 18　実験結果による学習者の能力分布

④ テスト問題は，実験と解析を一般的なプロセスにするため，その知識に関する基礎的な内容で単純な設問にした．この条件の下では，問題作成者の思考過程，表現方法などの違いによる影響が小さくなり，問題に関連しない知識の混在，問題に対する読解力の差異なども排除できると考えられる．

⑤ テスト問題の構成は，前半は Java 言語の関連知識，後半はソースコードに関する設問で，前者と後者の理解を測定する．前者は，オブジェクト指向の特徴であるカプセル化, 継承, 多相性などを考慮し, attribute, method, instance, instance variable， package，super class，sub class，class variable， class method，interface に関する設問にした．Java 言語のプログラムは，スタンドアロンアプリケーションとアプレットの 2 つの動作形態がある．本テストでは Java 言語の一般的なプログラミングにあたる前者を取り上げた. テスト問題は，これら事項を考慮して Java 言語に関する著書[27]，[36]の演習問題とプログラム例をもとに作成した.

⑥ 学習者の理解状態の測定において考えられる影響として，①学習者自身によるもの, ②教授者によるもの, ③学習環境によるもの, ④テストによるもの, が上げられる．①学習者自身によるものとしては，学習者のプログラミング経験，年齢，性別，職業，数学や国語に関連した基礎的能力，集中力，記憶力などが上げられる．②教授者によるものは，教授者の教育経験，被験者との年齢差，性別，指導する知識項目の順序，などがある．③学習環境によるものは，学習期間（定期的学習／集中的学習），教室の環境，プレゼンテーション技法，補助者の有無，学習者数，学習者数の性別の割合，などが上げられる．④テストによるものとしては，問題読解の容易性，出題順序，出題形式，テスト時間，テスト形態，テストメディア，テスト結果の影響力，などが考えられる．定量的なデータを得るためには，理解状態の測定に影響を与えていると考えられる，このような事項を考慮することは望ましいことであるが，教育あるいは学習という人間を扱った測定においては，これら事象の組み合わせが膨大になることから，実際の実験においては詳細な設定が困難であると考えられる．よって本実験においては，学習者の理解状態の測定において考えられるこれらの影響を考慮していない.

5.2.2 実験の手順

理解状態を測定する実験を，以下のような手順で行った.

① 教授者は，Java 言語の関連の資料から基礎的な知識を学習者に講義する.

② 学習者は例題のプログラムに対し，修正，コンパイル，実行といったコンピュータを用いた演習を行う.

③ 演習の終了後，学習者は Java 言語に関するペーパーテストを受ける.

テスト内容は，Java 言語の関連知識が 15 問（多岐選択式；A1～A15），プログラムの穴埋めが 5 問（短答式；B1.1～B1.5），出力結果の回答が 5 問(短答式；B2.1～B2.5)である，テストの内容を付録 A，付録 B にテスト項目の正解を表 10 に示す．テストの回答は，テスト項目ごとに「正解」か「間違い」のいずれかに判定される．出題形式は，定量的な回答の判定，多くの被験者に対応できる効率的な回答の判定処理を配慮し，多岐選択法，穴埋め記述，出力結果の記述とした.

プログラミングでは，アルゴリズム，コーディング，コンパイル，出力結果の確認といった，それぞれの過程における知識と理解が必要になる．本テストにおいても，付録 A の設問によりアルゴリズムとコーディングに関連した知識レベルが測定され，付録 B のプログラムの穴埋め記述，出力結果の記述によって，プログラミングに関するアルゴリズム，コーディング，処理過程の確認といった理解レベルが測定できると考えられる.

表10　テストの正解と知識項目

	テスト項目の正解 correct answer of the test items	問題に必要な知識 knowledge required for the answers	正解率 rate of the correct answers
A1	属性	Instance variable	83.3%
A2	哺乳類クラス	Super class	97.0%
A3	クラス変数	Class variable	83.3%
A4	継承	Inheritance	84.8%
A5	サブクラス	Sub class	93.9%
A6	属性	Instance variable	81.8%
A7	ボールクラス	Super class	80.3%
A8	クラス	Class	50.0%
A9	インスタンス	Instance	77.3%
A10	メソッド	Method	90.9%
A11	インスタンス変数	Instance variable	74.2%
A12	スーパークラス	Super class	95.5%
A13	クラスメソッド	Class method	93.9%
A14	インタフェース	Interface	74.2%
A15	パッケージ	Package	84.8%
B1.1	Class	Class	81.8%
B1.2	String	Method variable	66.7%
B1.3	new	Instance	65.2%
B1.4	set	Instance method	60.6%
B1.5	obj	Instance method	34.8%
B2.1	obj=(10,20)	String	42.4%
B2.2	x=20	Local variable	45.5%
B2.3	y=30	Local variable	45.5%
B2.4	getx: 10	Instance variable	47.0%
B2.5	gety: 20	Instance variable	47.0%

表 11　3 つの学習フェーズのテスト項目における回答回数

		The test items																												
Subjects		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
A	A-1	3	3	1	1	3	1	1	1	1	1	1	2	1	1	1	1	1	3	1	1	1	1	1	3	1	1	1	1	1	2
	A-2	3	1	2	1	3	1	1	1	1	1	1	1	1	1	1	1	1	1	2	1	1	1	1	1	2	1	1	2	1	1
	A-3	1	2	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	3	2	2
B	B-1	5	3	1	1	5	1	1	2	2	3	1	2	1	1	1	1	1	5	5	1	4	2	2	3	1	4	1	1	2	4
	B-2	1	1	5	1	2	1	1	1	1	1	1	3	1	1	2	1	1	3	4	2	1	1	5	4	1	4	1	1	1	1
	B-3	1	1	1	1	2	1	1	1	1	1	2	4	1	1	1	1	1	1	1	4	1	1	1	2	1	1	1	2	1	2
C	C-1	5	3	3	1	4	5	1	3	1	3	1	1	4	4	3	1	1	5	4	3	1	1	3	2	2	1	1	1	1	1
	C-2	1	2	3	1	4	1	1	1	4	1	1	2	2	1	1	1	2	2	5	1	1	1	1	1	1	1	1	1	1	1
	C-3	1	1	1	3	2	1	1	1	1	1	4	5	2	1	1	1	1	2	1	1	2	1	1	5	1	1	1	1	1	1
D	D-1	1	2	1	1	1	1	1	1	1	2	1	2	1	3	1	1	1	1	1	1	1	1	3	3	1	1	1	1	1	1
	D-2	1	1	2	1	3	1	1	1	1	1	2	1	1	1	1	1	1	1	2	1	2	1	3	1	2	1	1	1	1	1
	D-3	1	1	1	1	2	1	1	1	1	1	1	1	1	4	1	1	1	1	1	1	1	1	1	1	1	1	1	2	1	1
E	E-1	5	5	1	1	3	3	1	3	4	1	3	3	1	3	2	1	2	2	2	1	3	1	2	1	1	1	1	1	1	1
	E-2	3	1	1	1	3	1	1	1	3	1	3	3	1	1	2	1	1	2	1	1	5	1	3	1	4	1	1	3	1	1
	E-3	1	1	1	1	1	1	1	1	1	1	1	3	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

「うっかりミス」がないパターン : a 65
The pattern without ″an accidental failure″ : a 65
「うっかりミス」が生じたパターン : b 5
The pattern of ″an accidental failure″ : b 5

5.3　提案手法による順序関係の導出

「まぐれ当たり」の確率" f "は，多岐選択法の問題において，選択肢が 5 で偶然に成功する確率$\frac{1}{5}$から $f = 0.2$ とした．短答式の問題（B1.1～B2.5）については，回答として記述される文字の組み合わせが無限に存在するため $f = 0$ とした．

「うっかりミス」については，表 11 の 3 つの学習フェーズ（①学習前，②資料だけによる学習後，③演習による学習後）における Java 言語の理解状態の測定データ[42]から，偶然の失敗が生じる確率を求めた．第 1 学習フェーズから第 3 学習フェーズへと学習が進むにつれ，学習者の知識や理解が高まり，同じ内容の問題項目は間違わないと仮定すると，一つの学習フェーズを間違ったパターンが「うっかりミス」と考えられる．第 1，2 学習フェーズだけが間違ったパターンや，第 3 学習フェーズで 2 回以上間違ったパターンはこの仮説に矛盾するので除外し，第 1，2 学習フェーズを 1 回で正解して第 3 学習フェーズを 2 回目に正解するパターンを「うっかりミス」とした．　「うっかりミス」の確率 c は，3 つの学習フェーズで全て正解するパターン数と「うっかりミス」のパターン数の総和に対する後者の割合とし $c = 0.071$ にした．

表 12　矛盾関数 P により算出した矛盾率

The knowledge item which gives an ordering relation		The knowledge item which receives an ordering relation																								
		f1			f2			f3		f4			f5	f6	f7				f8			f9				
		B2.1	B2.4	B2.5	A1	A6	A7	B2.2	B2.3	A10	A12	A5	A3	A2	A14	A15	A8	B1.5	A9	B1.1	B1.2	A11	A13	A4	B1.3	B1.4
f1	B2.1	0.05	0.60	0.60	0.81	0.83	0.81	0.79	0.79	0.88	0.90	0.89	0.81	0.91	0.79	0.82	0.46	0.77	0.78	0.90	0.87	0.77	0.88	0.84	0.83	0.82
	B2.4	0.37	0.03	0.03	0.79	0.77	0.80	0.79	0.79	0.86	0.88	0.87	0.83	0.89	0.80	0.83	0.45	0.81	0.71	0.88	0.84	0.72	0.86	0.79	0.78	0.76
	B2.5	0.37	0.03	0.03	0.79	0.77	0.80	0.79	0.79	0.86	0.88	0.87	0.83	0.89	0.80	0.83	0.45	0.81	0.71	0.88	0.84	0.72	0.86	0.79	0.78	0.76
f2	A1	0.13	0.20	0.20	0.00	0.11	0.02	0.22	0.22	0.39	0.44	0.37	0.44	0.43	0.23	0.34	0.10	0.08	0.06	0.37	0.41	0.58	0.45	0.34	0.34	0.38
	A6	0.37	0.20	0.20	0.16	0.00	0.17	0.45	0.45	0.45	0.56	0.43	0.63	0.49	0.21	0.23	0.17	0.20	0.19	0.37	0.41	0.39	0.43	0.23	0.34	0.29
	A7	0.37	0.43	0.43	0.08	0.23	0.00	0.54	0.54	0.50	0.60	0.48	0.48	0.53	0.28	0.30	0.07	0.34	0.26	0.50	0.49	0.53	0.48	0.39	0.43	0.38
f3	B2.2	0.75	0.81	0.81	0.80	0.83	0.83	0.04	0.04	0.87	0.87	0.88	0.80	0.89	0.81	0.84	0.36	0.84	0.78	0.90	0.84	0.81	0.86	0.82	0.82	0.80
	B2.3	0.75	0.81	0.81	0.80	0.83	0.83	0.04	0.04	0.87	0.87	0.88	0.80	0.89	0.81	0.84	0.36	0.84	0.78	0.90	0.84	0.81	0.86	0.82	0.82	0.80
f4	A10	0.25	0.20	0.20	0.10	0.10	0.11	0.22	0.22	0.00	0.12	0.06	0.10	0.19	0.14	0.17	0.09	0.34	0.06	0.21	0.15	0.24	0.13	0.17	0.08	0.19
	A12	0.13	0.10	0.10	0.02	0.08	0.08	0.04	0.04	0.01	0.00	0.00	0.02	0.04	0.04	0.07	0.15	0.20	0.04	0.08	0.08	0.04	0.01	0.02	0.08	0.10
	A5	0.13	0.10	0.10	0.02	0.02	0.02	0.11	0.11	0.01	0.01	0.00	0.06	0.03	0.01	0.06	0.05	0.20	0.01	0.03	0.08	0.09	0.01	0.02	0.08	0.10
f5	A3	0.13	0.43	0.43	0.44	0.60	0.37	0.22	0.22	0.39	0.44	0.45	0.00	0.43	0.07	0.43	0.10	0.20	0.30	0.55	0.33	0.14	0.45	0.43	0.25	0.10
f6	A2	0.13	0.10	0.10	0.01	0.01	0.01	0.04	0.04	0.02	0.01	0.00	0.01	0.00	0.00	0.03	0.01	0.08	0.01	0.01	0.03	0.05	0.00	0.00	0.03	0.01
f7	A14	0.57	0.66	0.66	0.53	0.47	0.48	0.68	0.68	0.65	0.67	0.59	0.39	0.62	0.00	0.38	0.09	0.34	0.26	0.67	0.72	0.58	0.59	0.52	0.66	0.60
	A15	0.13	0.32	0.32	0.28	0.12	0.13	0.33	0.33	0.41	0.46	0.39	0.38	0.45	0.03	0.00	0.01	0.20	0.23	0.29	0.49	0.25	0.39	0.18	0.43	0.29
	A8	0.78	0.81	0.81	0.79	0.79	0.75	0.76	0.76	0.85	0.88	0.86	0.79	0.86	0.69	0.76	0.00	0.81	0.68	0.88	0.86	0.75	0.84	0.79	0.83	0.82
	B1.5	0.86	0.90	0.90	0.83	0.84	0.84	0.90	0.90	0.91	0.92	0.92	0.85	0.92	0.79	0.86	0.58	0.08	0.81	0.93	0.91	0.80	0.91	0.85	0.90	0.90
f8	A9	0.37	0.20	0.20	0.27	0.36	0.37	0.45	0.45	0.53	0.62	0.51	0.51	0.61	0.15	0.50	0.02	0.34	0.00	0.64	0.65	0.49	0.44	0.34	0.50	0.47
	B1.1	0.37	0.32	0.32	0.33	0.25	0.35	0.54	0.54	0.52	0.56	0.43	0.57	0.49	0.39	0.32	0.27	0.34	0.46	0.00	0.41	0.48	0.50	0.41	0.34	0.47
	B1.2	0.48	0.52	0.52	0.69	0.66	0.66	0.45	0.45	0.73	0.77	0.75	0.65	0.77	0.73	0.75	0.33	0.47	0.72	0.75	0.00	0.49	0.75	0.65	0.62	0.69
f9	A11	0.48	0.43	0.43	0.73	0.60	0.66	0.68	0.68	0.69	0.67	0.68	0.47	0.70	0.58	0.58	0.25	0.47	0.57	0.70	0.49	0.00	0.59	0.46	0.50	0.38
	A13	0.05	0.03	0.03	0.06	0.02	0.02	0.04	0.04	0.04	0.03	0.01	0.06	0.03	0.01	0.06	0.01	0.08	0.00	0.08	0.08	0.01	0.00	0.00	0.01	0.01
	A4	0.25	0.10	0.10	0.28	0.12	0.21	0.22	0.22	0.41	0.38	0.31	0.38	0.37	0.16	0.18	0.05	0.08	0.07	0.37	0.23	0.08	0.22	0.00	0.08	0.04
	B1.3	0.25	0.20	0.20	0.68	0.64	0.65	0.33	0.33	0.72	0.79	0.77	0.64	0.78	0.68	0.74	0.22	0.47	0.62	0.75	0.65	0.53	0.71	0.59	0.01	0.10
	B1.4	0.37	0.32	0.32	0.74	0.68	0.69	0.45	0.45	0.79	0.82	0.81	0.64	0.80	0.67	0.73	0.25	0.58	0.66	0.82	0.77	0.54	0.76	0.63	0.34	0.01

矛盾率は，矛盾関数 P に「まぐれ当たり」の確率 f =0 (短答式)か f =0.2 (多岐選択式)，「うっかりミス」の確率 c =0.071 を適用して算出して表 12 に示す．表 12 は，知識項目を 5.4 節におけるテストデータの因子分析の結果からクラスタごとに並べ替え，”0.1”未満の数値を抽出したものである．表 12 の列方向の項目（A1～B2.5）は「順序関係」を与えている知識項目で，行方向が「順序関係」を受けている知識項目である．表 12 の 600 の矛盾率は，前者の 25 項目の知識項目に，前者と同じ知識項目を除いた後者の 24 項目の知識項目が対応している．

知識項目の関連構造を，”0.1” 未満の矛盾率から導出した順序関係によって構築する．統計資料の整理法の一つである度数分布は，データを要約して対象集団の特性を把握するために用いられる[16]．したがってデータの集まりが示す性質の概要は，度数分布表あるいは度数分布図から読み取れる．表 13 の矛盾率の度数分布表において，“ 0.0~0.10 ”の階級の度数が他の階級と比較して高く，この階級が矛盾率の特性を示していると考えられるので，順序関係を導出するための指標に”0.1” を用いる．

表 13　矛盾率の度数分布

矛盾率の階級境界値 class of inconsistency probability	度数 frequency	相対度数 relative frequency
0.0 ～0.10	104	0.173
0.10～0.20	48	0.080
0.20～0.30	53	0.088
0.30～0.40	62	0.103
0.40～0.50	66	0.110
0.50～0.60	38	0.063
0.60～0.70	48	0.080
0.70～0.80	64	0.107
0.80～0.90	99	0.165
0.90～1.00	18	0.030

5.4　関連構造の構築

5.4.1　クラスタへの集約

知識項目は，テスト項目の記号を用いて知識項目 A1 を”A1”というように記述する.

表 12 における 0.1 未満の矛盾率から導出した順序関係において，A1 から A9 への一方向の関係は，(A1) ⟶ (A9) と表し，A1 から A7，A7 から A1 といった双方向の関係を(A1) ⟷ (A7) として構築した関連構造を図 19 に示す.

因子分析で得られた因子負荷量から, 25 の知識項目をいくつかのクラスタに集約する．10 因子による因子分析では，第 10 因子目の因子負荷量が“0”に近似し，第 10 因子からは絶対値が最大の因子負荷量を抽出できない．本章では 9 因子による因子分析の因子負荷量をもとに, 25 の知識項目を 9 つのクラスタに編成した.

それぞれのクラスタの因子を次のように抽出した．クラスタ f(2)，f(4)，f(7)，f(8)，f(9)においては，一つの概念では説明できないため 2 つの概念で表した．因子分析から抽出したクラスタの因子と，クラスタに含まれる知識項目を表 14 に示す.

クラスタ f(1)を構成する B2.1，B2.4，B2.5 は，ソースコードであることから，これらに共通する因子としてソースコードの Method を抽出し，Method(Code)

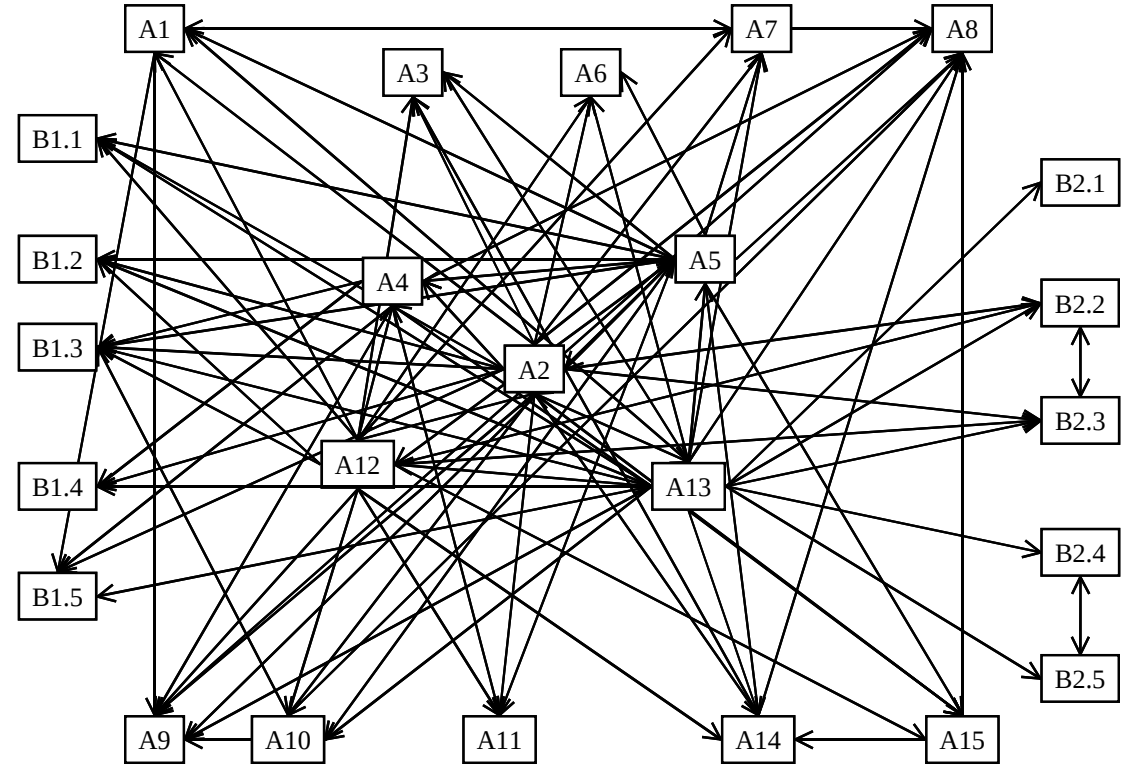


図 19　知識項目の関連構造

表 14　因子負荷量による知識項目の分類

classification of the factors		knowledge item of the test items	factor loadings								
			a(1)	a(2)	a(3)	a(4)	a(5)	a(6)	a(7)	a(8)	a(9)
f(1)	Method (Code)	(B2.4) Instance variable	0.95	0.11	0.14	−0.05	−0.04	−0.01	−0.05	0.05	−0.22
		(B2.5) Instance variable	0.95	0.11	0.14	−0.05	−0.04	−0.01	−0.05	0.05	−0.22
		(B2.1) String	0.68	0.14	0.27	−0.03	0.23	−0.06	−0.20	0.17	−0.10
f(2)	Class,Variable	(A1) Attribute	0.18	0.77	0.15	−0.13	−0.02	0.08	−0.11	0.05	0.14
		(A7) Super class	0.04	0.68	0.00	0.01	0.04	0.07	−0.20	0.07	−0.11
		(A6) Attribute	0.15	0.46	0.01	−0.01	−0.31	0.12	−0.28	0.16	−0.25
f(3)	Variable (Code)	(B2.2) Local variable	0.20	0.06	0.96	−0.10	0.07	0.06	−0.06	0.06	−0.11
		(B2.3) Local variable	0.20	0.06	0.96	−0.10	0.07	0.06	−0.06	0.06	−0.11
f(4)	Super Class, Sub Class	(A5) Sub class	0.03	0.18	−0.02	−0.76	−0.05	0.31	−0.11	0.07	−0.10
		(A12) Super class	0.02	−0.13	0.13	−0.75	0.04	−0.06	0.04	−0.01	0.03
		(A10) Method	0.02	0.27	0.03	−0.37	0.12	−0.11	0.10	−0.05	−0.08
f(5)	Class,Variable	(A3) Class variable	0.08	−0.01	0.14	−0.07	0.80	0.08	−0.10	0.01	−0.12
f(6)	Thought	(A2) Super class	−0.06	0.10	0.10	−0.05	0.07	0.75	−0.02	0.05	−0.07
f(7)	Class,Interface	(A15) Package	0.06	0.17	0.06	0.08	−0.05	−0.10	−0.73	0.09	−0.10
		(A14) Interface	0.02	0.19	−0.05	−0.13	0.30	0.33	−0.58	−0.13	−0.09
		(A8) Class	0.16	0.20	0.28	0.01	0.14	0.10	−0.34	−0.10	−0.21
		(B1.5) Instance method	0.20	0.25	0.12	0.01	0.21	0.08	−0.25	0.24	−0.23
f(8)	Class,Instance	(B1.2) Method variable	0.21	0.09	0.27	−0.07	0.14	−0.05	0.06	0.49	−0.25
		(A9) Instance	0.26	0.40	0.07	−0.24	0.03	0.13	−0.29	−0.44	−0.24
		(B1.1) Class	0.17	0.19	−0.04	0.01	−0.19	0.19	−0.15	0.39	−0.04
f(9)	Variable,Method	(B1.4) Instance method	0.42	0.14	0.33	0.13	0.25	0.17	0.00	0.04	−0.58
		(A11) Instance variable	0.17	−0.21	−0.05	−0.02	0.21	−0.16	−0.15	0.14	−0.58
		(B1.3) Instance	0.41	0.23	0.32	0.03	0.17	0.04	0.15	0.15	−0.58
		(A4) Inheritance	0.14	0.14	0.10	−0.13	−0.11	0.12	−0.26	0.05	−0.57
		(A13) Class method	0.11	0.06	0.10	−0.39	−0.06	0.33	−0.13	−0.19	−0.53

と表記する. クラスタ f(3)を構成する B2.2, B2.3 も同様の解釈で, Variable(Code) とした.

クラスタ f(2)構成する A1，A6，A7 から Class と Variable を因子として抽出した. クラスタ f(4)を構成する A5, A10, A12 からは, Super Class と Sub Class を抽出した. クラスタ f(5)を構成する A3 は，解答が「クラス変数」で，問題中に「クラス」と「変数」の用語が表示されていることから，この因子を Class と Variable とした. クラスタ f(7)を構成している A8，A14，A15，B1.5 には，Java 言語に関する知識項目とソースコードが含まれており，この因子を Class と Interface にした. クラスタ f(8)は A9，B1.1，B1.2 で構成されていることから，この因子を Class と Instance とした. クラスタ f(9)は A4，A11，A13，B1.3，B1.4 で構成され，この因子を Variable と Method とした.

一方，クラスタ f(6)を構成する A2 のテスト回答は，一般常識の知識をもとに問題を熟読すれば，消去法によって回答群から選択できる. A2 に関するテスト項目の回答が思考力に影響を受けることから，この因子を「Thought: 思考」とした.

5.4.2 関連構造の簡単化

知識項目だけの複雑な関連構造を，4.5 節の提案手法によってクラスタに編成して簡単化した関連構造を図 20~23 に示す.

図 20 においてクラスタｆ(6)（以下, それぞれのクラスタをｆ(6)というように記述する.）の A2 からｆ(3) の B2.2 と B2.3 へ，順序関係を表す２本の矢線 ”⟶“ で描かれているが，これらをｆ(6)からｆ(3)への順序関係とし，1 本の矢線 ”➡” で表す．同様に双方向の順序関係は ”⟷” で表し，これらを図 21 に示す．図 21 における f(6)と f(4)，f(6)と f(9)，f(4)と f(9) の双方向の順序関係は，同方向の矛盾率の平均値が小さい方向を順序関係が強いとして一方向の矢線に変換した．これらクラスタの順序関係を f(6)➡f(4)，f(6)➡f(9)，f(4)➡f(9)として表した.

f(9)から f(8)の順序関係は，f(9)➡f(8)と f(9)➡f(2)➡f(8)の順序関係が並行して存在している．後者の順序関係は，f(9)を理解して f(2)が理解され，次に f(8)の理解が成立しており，f(2)の理解を介して f(8)への順序関係が継承されている．このような順序関係では，後者の順序関係が成立すれば前者の順序関係も含まれることになり，f(9)➡f(8)の順序関係は f(9)➡f(2)➡f(8)に包括して表した．他の並行した順序関係についても同様の処理を行い，クラスタの関連構造を簡単化した.

図 22 は，クラスタ間の全ての順序関係を” ┅➡ ”で表し，本提案手法によって簡単化した順序関係を “➡” で表した．さらにクラスタの関連構造を，“➡” だけで構築すれば図 23 のようになる.

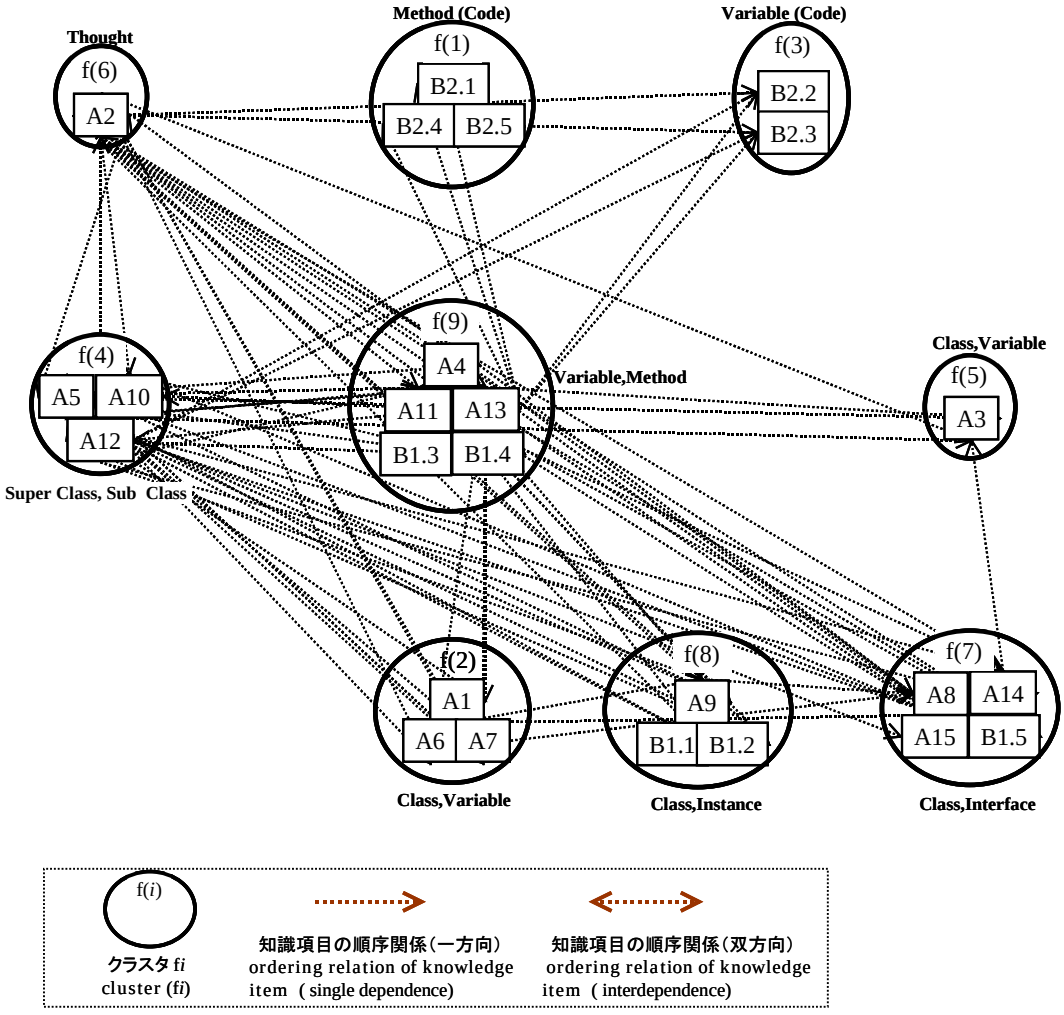


図 20　クラスタに集約された知識項目の関連構造

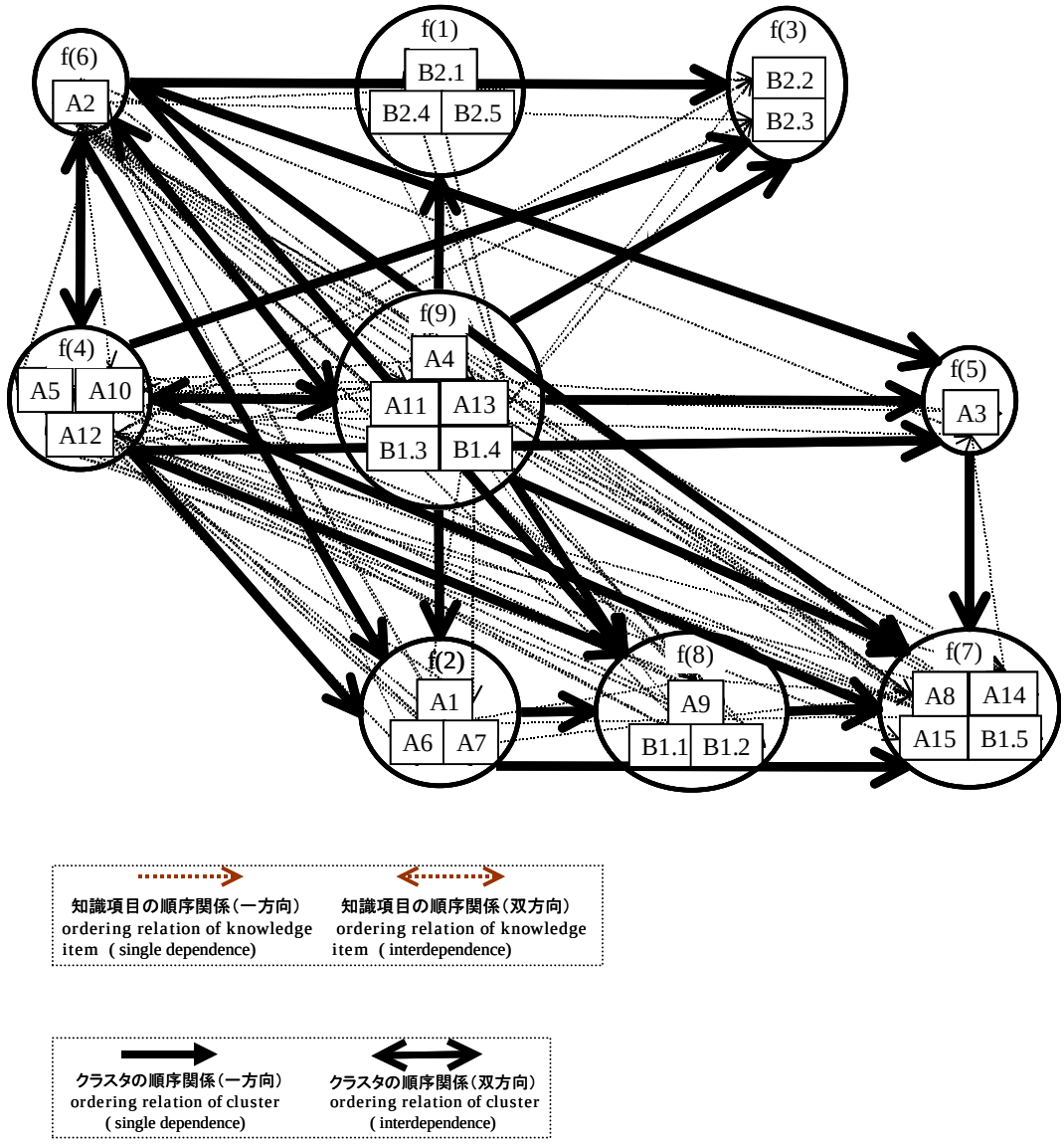


図21　クラスタの関連構造

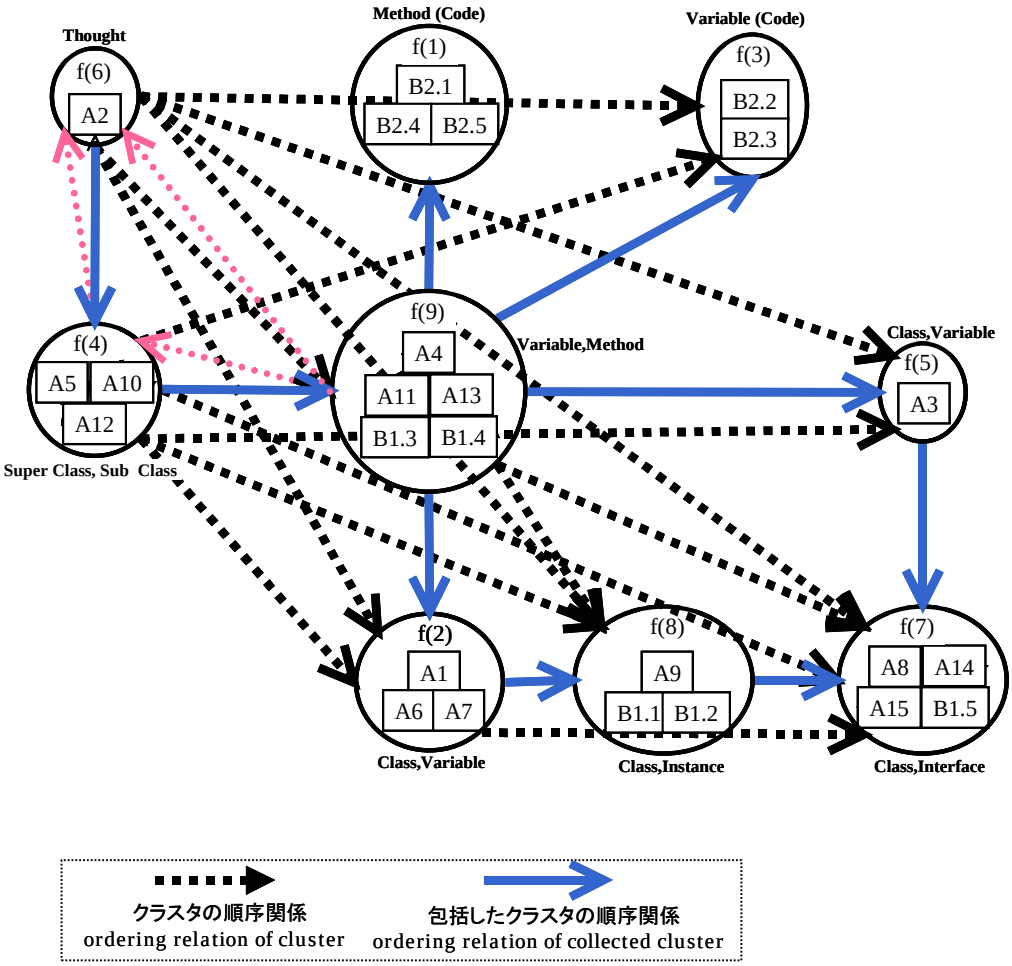


図 22　関連構造にける簡単化の処理

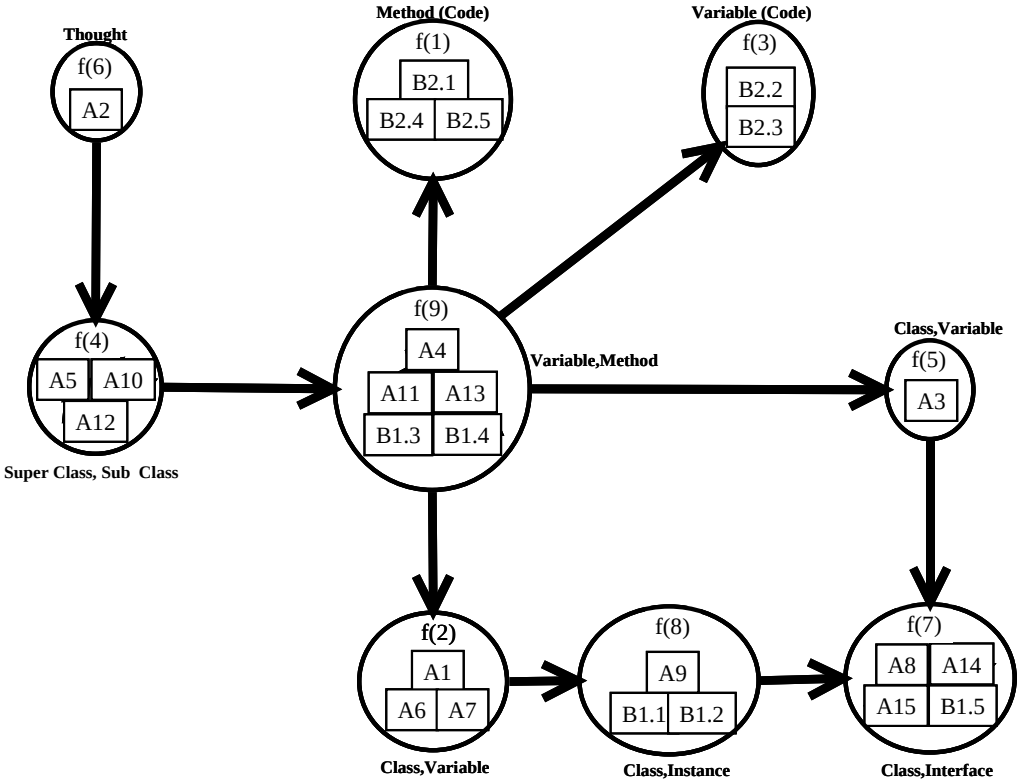


図 23　簡単化されたクラスタの関連構造

5.5 関連構造の考察

5.5.1 関連構造の形状

Java 言語の理解を図 23 の関連構造（以下，関連構造と呼ぶ.）の形状から考察する．伊藤らは，ファジィ項目関連構造分析（Fuzzy item relational structure analysis : FIRS 分析）を提案し，LISP 言語に関するテスト（テスト項目数：8，学習者数：18）から順序関係を解析し，竹谷の項目関連構造分析（Item relational structure analysis : IRS 分析）にテスト結果を適用して順序関係を比較している[9]．FIRS 分析では，テスト項目がノードとなった学習者全体のリダクショングラフを描画している．このグラフでは，全体のノードが直線的にリンクし，途中に 1 つのグループ（2 つのノードが並行して集約されている）が存在している．IRS 分析を適用したグラフでは，1 つの根ノードから 2 つのノードと 1 つのグループ（3 つのノードが並行して集約されている）に分岐している．

FIRS 分析と IRS 分析のグラフは, LISP 言語の順序関係を描画している. LISP 言語は関数型言語で，Java 言語はオブジェクト指向言語である．前者は純粋に数学的な表現だけでプログラムが実行され，後者では変数とメソッドが一つのクラスに統合される．また，インタフェースは一連のメソッドの形態を定義したもので，メソッドの処理形態はインタフェースを使用するクラスに定義され，クラスをもとに生成されたオブジェクトがインスタンスと呼ばれる．このようにオブジェクト指向である Java 言語では，知識の理解の進行に伴い複合的な知識が必要になる．クラスタ f(7)の Class, Interface が，Class, Variable, Method, Instance の知識が複合的に用いられて理解されていることが関連構造から把握できる．これより f(9)⇒(5)⇒f(7)と f(9)⇒(2)⇒f(8)⇒f(7)の関連構造におけるリング状の構造は，Java 言語の特長の一つを示していると考えられる．

藤原らのネットワーク型テスト理論は，各テスト項目の関係を確率ネットワークで表現することによって，その領域の特有の構造を学習者モデルなどに組み込んでいる[5]． 1 次方程式の理解を示した意味ネットワーク構造は，15 のノードで構成され，2 つの根ノード（正負の数の表現，数量の文字表現）と 1 つの終端のノード（応用問題）が存在する．プログラミング言語を扱った関連構造，FIRS

分析や IRS 分析のグラフでは根ノードが 1 つであるが, 数学の領域では 2 つの根ノードが存在している．前者の 1 つの根ノードは，知識の理解がソースコードの理解に繋がっている結果であると考えられる.

5.5.2 学習プロセスと著書の出展順序

図 23 のクラスタの関連構造から導出した学習プロセスと，Java 言語を解説した著書[24],[27],[28],[36]から, Class, Method, Super Class, Sub Class, Variable, Interface に関する出展順序を比較して理解順序を考察する.

学習プロセス① ~ ③を，規則 a，規則 b に従ってクラスタの関連構造から導出する．規則 a は関連構造で連続したクラスタは途中で切断しない．規則 b は連続しないクラスタの学習順序は任意に決定する.

・学習プロセス①

f(4):Super class, Sub class ⇒ f(9):Variable, Method ⇒ f(1):Method(Code) ⇒ f(3): Variable(Code) ⇒ f(5),f(2):Class, Variable ⇒ f(8):Class, Instance ⇒ f(7):Class, Interface

・学習プロセス②

f(4):Super class, Sub class ⇒ f(9):Variable, Method ⇒ f(2),f(5):Class, Variable ⇒ f(1):Method(Code) ⇒ f(3):Variable(Code) ⇒ f(8):Class, Instance ⇒ f(7):Class, Interface

・学習プロセス③

f(4):Super class, Sub class ⇒ f(9):Variable, Method ⇒ f(5),f(2):Class, Variable ⇒ f(8):Class, Instance ⇒ f(7):Class, Interface ⇒ f(1):Method(Code) ⇒ f(3): Variable(Code)

プロセス①では，f(9)から f(1): Method(Code)，f(3): Variable(Code)に関するソースコードを学習した後，Java 言語の関連知識を学習する．このプロセスでは, Class, Interface の概念が形成されない状態でソースコードを扱うため, コードの理解が促進されにくいと考えられる．プロセス②では, f(2), f(5)の Class, Variable に関する学習後，f(1): Method(Code)，f(3): Variable(Code)に関するソースコード

の学習に進み，f(8): Class, Instance，f(7): Class, Interface に関する関連知識を学習する．このプロセスでも，Interface の概念が形成されない状態でソースコードを扱うため，上述と同様の傾向が考えられる．プロセス③は，f(5), f(2), f(8), f(9) の Class, Variable, Interface に関する概念が形成された後に, f(1)：Method(Code), f(3): Variable(Code)に関するコードの学習を行っている．プロセス③では，ソースコードの記述に必要な Java 言語の概念が形成されてからソースコードが理解されるので，Java 言語の応用的な学習が期待できる． Java 言語の教育や学習の体系化は，クラスの関連構造から設定された学習プロセスの検討によって可能であると考えられる．

Java 言語を解説した著書[24],[27],[28],[36]の目次において，本実験で用いた知識項目に関連する出展順序を表 15 に示す．関連構造においてクラスタ f(1)：Method(Code)，f(3)：Variable(Code)はクラスタ f(9)：Variable，Method から順序関係を受けているが，f(1): Method(Code)，f(3): Variable(Code)の間には順序関係が存在しない．そのため著書[24],[27],[28],[36]における Method(Code)，Variable(Code)の出展順序が，著者の知識や経験によって設定されたため異なったと考えられる. また関連構造ではクラスタ f(7): Interface が, クラスタ f(5):Class, Variable，f(8):Class, Instance から順序関係を受けているため複合的な概念の理解を必要としている．よって，それぞれの著書においても複合的な概念の理解の配慮から，Interface の出展順序が後半になっていると考えられる．

関連構造から導出した学習プロセス③と，著書[24],[27],[28],[36]における知識項目の出展順序を図 24 に示す．学習プロセス③では Super Class, Sub Class から始まっているが，それぞれの著書は Method，Variable などの知識項目の後に出展されている．Java 言語の基礎的なプログラミングでは，必要なオブジェクトのクラスを定義し，それらクラスのインスタンスを作り，そのインスタンスを組み合わせるといった手順になっている．主にクラスに定義されることは、作り出されるオブジェクトがどのような定数や変数をもっているのか、どのような動作をするのかの二つである．このような観点からこれらの著書では，プログラミング言語を解説しながら必要な概念を理解させるため，Method，Variable だけで記述できる小さな処理単位を理解できるように出展順序が設定されていると考え

られる．また，それぞれの著書は読者が独習の場合を想定し，学習に時間を要しても簡単なプログラムの理解を優先させた結果，このような出展順序になったと考えられる．一方，導出した学習プロセスから，教授者の指導を伴った学習では Super Class, Sub Class といったクラスの概念や分類が，Class 全体の概念の理解よりも早い時期に学習させれば Java 言語の理解が促進されると考えられる．

表 15　市販著書における知識項目の出展順序

著書①		著書②		著書③		著書④	
知識項目	頁番号	知識項目	頁番号	知識項目	頁番号	知識項目	頁番号
Variable	11,168	Class	47	Variable	50,120	Variable	101
Method	43,124	Method	49	Class	112	Class	105
Class	124	Variable	52	Method	121	Super Class	106
Super Class	165	Super Class	67	Super Class	125	Sub Class	106
Sub Class	165	Sub Class	69	Sub Class	125	Method	107
Interface	197	Interface	80	Interface	166	Interface	112

著書① 独習Java（翔泳社），　著書② JAVAﾌﾟﾛｸﾞﾗﾐﾝｸﾞ講座（アスキー）

著書③ Javaﾌﾟﾛｸﾞﾗﾐﾝｸﾞ徹底演習（日刊工業新聞）　著書④ はじめてのJAVA（技術評論社）

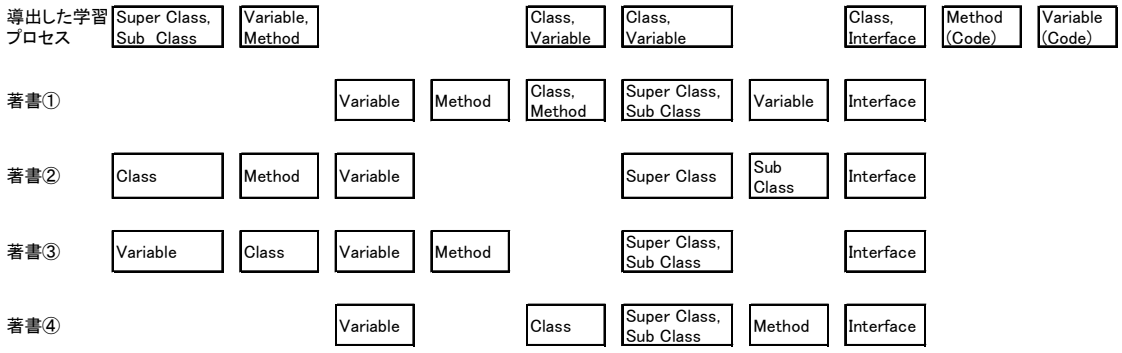


図 24　学習プロセスと著書の出展順序

5.6　まとめと今後の課題

本章では, これまで経験的知見によって推測されていたであろう Java 言語の理解順序を，順序理論のモデルが適用された統計的分析手法による順序関係の導出と関連構造の構築によって解明した．知識項目の理解の有無を判定するため 66 名の被験者に講義・演習とテストによる実験を行い，その結果を 2×2 分割表に割り当てて矛盾率から順序関係を導出した．知識項目だけの複雑な関連構造を，因子負荷量によって 25 の知識項目が 9 つのクラスタに集約された関連構造から理解順序を解析した．Java 言語の学習に関する実験と提案手法による解析から，以下のような知見を得た.

① 順序関係の導出に矛盾率を提案し，600 の矛盾率を算出した.
② 600 の矛盾率から”0.1”未満の数値を抽出し，104 の順序関係を導出した.
③ 25 の知識項目を因子負荷量から 9 つのクラスタに集約し，クラスタの因子を抽出した.
④ 知識項目のクラスタへの集約と関連構造の簡単化により，Java 言語の理解順序を解明した.
⑤ 簡単化されたクラスタの関連構造から，効率的な学習プロセスを導出した.
⑥ Java 言語を解説した著書における知識項目の出展順序と導出された学習プロセスを比較し，Super Class, Sub Class の出展順序の違いから，独習を前提としたトップダウン型の著書の出展順序と，ボトムアップ型の教授者を伴う学習プロセスの違いを考察した.

知識項目をクラスタに集約した簡潔な関連構造は，知的教育システムにおいて学習者モデルを構築する際に有効であると考えられる．関連構造から導出された学習プロセスは今後の実験による評価が必要である．また，本提案手法を他のプログラミング言語にも適用して順序関係の違いを解析し，プログラミング言語の特長を解明したいと考えている.

6．おわりに

本論文では，ソフトウェア技術者を安定して継続的に確保するための組織的な管理手法の整備として，労働者派遣事業におけるソフトウェア技術者の教育支援体制を提案した. そして, ソフトウェア技術者が開発環境に関する未修得の知識・技術を系統的に把握して効率的に習得するためのプログラミング教育の体系化を目的として，開発環境の一つとして Java 言語に注目し，学習過程の進行にともなう理解状態の変化と，理解順序を知識間の順序関係によって解明した.

ソフトウェア技術者の教育支援体制では，労働力需給の変化と派遣労働の形態を分析し，ソフトウェア技術者の派遣に適した労働者派遣事業を取り上げて労働力需給の課題を整理した．課題への対応として，過去の景気変動やソフトウェア技術者の派遣実績を生かした長期的な需要予測，派遣したソフトウェア技術者の報告に基づいた短期的な需要予測，その誤差を吸収するための関連業務の兼業，MIT Sloan School of Management や社会人を対象にした大学院のような教育機関との連携を提案した．これら課題への対応をもとに，派遣元が派遣先からの人的数量の需要に対応するための組織的な管理手法を，長期・短期的な需要予測，関連業務の兼業，教育機関との連携が統合された教育支援体制を提案した．本教育支援体制は，ソフトウェア技術者の労働者派遣事業における安定供給と先端的技術が必要な派遣労働を可能すると考えられる.

Java 言語の理解状態の解析では, 学習段階における因子の遷移を因子負荷量の変化から観察する手法を提案した．Java 言語の学習課程（① 学習の前，② 基礎的な知識の学習後，③ プログラミング演習の後）において実験を行い，テスト結果の因子分析から抽出された因子の遷移によって理解状態を解明した. その結果，新しいプログラミング言語を学ぶ学習段階② においてプログラミング学習に関する新しい因子が形成され，その因子が実際のプログラム演習によって，「基礎的なプログラミングに必要な知識」，「その言語固有の概念」，プログラミングに必要のない「プログラミング言語に依存しない抽象的な概念」へと遷移することを解明できた．またプログラミング言語固有の概念がプログラミング演習によって理解されることを因子の遷移から解明し，実際のプログラミング演習によってプログラムの理解に変化が生じることを実験によって確認した．このような学習段階

の進行にともなう因子の解析は，学習者の理解状態の変化に適応した教育形態の形成に有効であると考えられる．

Java 言語の理解順序の解析では, 知識間の順序関係を導出する統計的分析手法と，その全体構造が表された関連構造の簡単化を提案した．統計的分析手法は，テストで生じるいわゆる「まぐれ当たり」と「うっかりミス」による回答を，二項分布の分布関数によって統計的に補正し順序関係を導出する．関連構造の簡単化では，因子分析を適用して知識項目をクラスタに集約し，知識項目だけの複雑な関連構造を簡単化して構築した．テスト結果の解析では，25 の知識項目から 104 の順序関係を導出し，その知識項目が 9 つのクラスタに集約された関連構造に構築して Java 言語の理解順序を解明した．Java 言語の効率的な学習プロセスを関連構造から導出し，学習プロセスと Java 言語が解説された著書の目次との違いを考察した．

ソフトウェア技術者の教育支援体制においては，派遣元企業の経営経験や派遣実績の蓄積により労働力需要の予測精度が向上し，企業が必要とするソフトウェア技術者の人材育成事業など関連業務の兼業も積極的に行われている．一方，先端的な技術に対応するための教育機関との連携は，具体的な教育連携の仕組み作りがこれからの課題である．例えば，ソフトウェア技術者にとって重要な応用的技能の習得のためのカリキュラムの整備や，イギリスの Brunel 大学のような企業の要請に対応したカリキュラム編成などが上げられる．これら対応においてはプログラミング教育の体系化が必要になり, Java 言語の理解状態や理解順序の解析で得られた教育形態や学習プロセスに関する知見が有用であると考えられる．加えて，労働者の自主的な能力開発を支援する教育訓練給付制度の指定対象に追加された夜間大学院や通信制大学院などの併用なども上げられる．労働者派遣事業における派遣元企業は，情報通信技術の進歩に対応できるソフトウェア技術者の供給元として重要な役割を果たしており，ソフトウェア技術者を安定して供給するための教育支援体制がこれからも重要になると考えられる． また，専門性の高いソフトウェア技術者を確保するため，対象業務の詳細な分析による職務特性の明確化が必要になると考えられる．

Java 言語の理解順序の解析における提案手法は，結果がテスト問題に依存し，因子分析において因子の意味を考察する必要もあるため，成果は Java 言語特定

の領域にもとづいている．しかし，テスト問題の作成と因子分析における因子の意味付けさえできれば，対象となる学習者集団の大多数にとって知識項目間の順序関係が共通している仮定のもとで, 適用領域を広げることは可能と考えられる．例えば, C++, Objective-C などオブジェクト指向のプログラミング言語, UNIX, Linux などのオペレーティングシステム，ネットワーク技術などに加え，工学，理学，農学などにおける新しい知識・技術の領域が考えられる．また，提案手法の適用範囲を広げ，実際の教育システムに組み込んで教育現場で利用すれば，学習者の理解をシステム上で把握することが可能になり，効果的な教育形態が形成できると考えられる．理解順序の解析において関連構造から導出した学習プロセスの実験による評価と，学習プロセスの導出手法の提案が今後の課題として残された．

Java 言語の理解順序の解析における提案手法やクラスタの関連構造などの成果は，CAI (Computer-Aided Instruction), ICAI / ITS (intelligent CAI / Intelligent Tutoring System), ILE (Interactive Learning Environment)などの教育システムにおける学習者モデルの構築において有用であると考えられる．今後は，ソフトウェア工学教育の体系化を促進させるため本提案手法の適用範囲を広げ，教育システムの開発へ展開したいと考えている．

謝辞

本研究を進めるにあたり，常日頃より懇切なご指導，ご助言と共に，適切なご指摘を賜りました，奈良先端科学技術大学院大学 情報科学研究科　松本 健一 教授　に，心から深く感謝申し上げます.

本研究を進めるにあたり，貴重なご指導，ご助言と共に，適切なご示唆を賜りました，奈良先端科学技術大学院大学 情報科学研究科　渡邉 勝正 教授　に，心から深く感謝申し上げます.

本研究を進めるにあたり，貴重なご指導，ご助言と共に，温かい激励を賜りました，大阪大学大学院 基礎工学研究科　井上 克郎 教授　に，心から深く感謝申し上げます.

本研究を進めるにあたり，貴重なご指導，ご助言と共に，詳細なご指摘を賜りました，奈良先端科学技術大学院大学 情報科学研究科　関　浩之 教授　に，心から深く感謝申し上げます.

本研究を進めるにあたり，貴重なご指導，ご助言を賜りました，奈良先端科学技術大学院大学 情報科学研究科　飯田　元 助教授　に，心から深く感謝申し上げます.

本研究を進めるにあたり，認知科学の観点から貴重なご助言を賜りました，奈良先端科学技術大学院大学 情報科学研究科　中小路 久美代 助教授　に，心から深く感謝申し上げます.

本研究を進めるにあたり，実験準備，論文作成，学会発表まで常に適切で熱心なご指導を頂き，細部にわたる有益なご指摘を賜りました，奈良先端科学技術大学院大学 情報科学研究科　島　和之 助手　に，心から深く感謝申し上げます.

本研究を進めるにあたり，終始熱心に相談に応じて頂き，有益なご指導を賜りました，奈良先端科学技術大学院大学 情報科学研究科　門田 暁人 助手　に，心から深く感謝申し上げます.

本研究を進めるにあたり，貴重なご助言，研究者としての姿勢をご教示賜ると共に，本研究の開始から本稿に至るまでの道のりにおいて，懇切なご指導と暖かい激励を賜りました，奈良先端科学技術大学院大学　鳥居 宏次 学長　に，心から深く感謝申し上げます.

本研究を進めるにあたり，貴重なご助言，ご協力を賜りました，奈良先端科学技術大学院大学 情報科学研究科 ソフトウェア計画構成学講座　松本健一 研究室の皆様　に，心から深く感謝申し上げます.

本研究を進めるにあたり，貴重なご助言，ご協力を賜りました，奈良先端科学技術大学院大学 情報科学研究科　内田 眞司 氏(現 近畿大学工業高等専門学校)，古賀 健太郎 氏 (現 株式会社 日立製作所)，白石 裕美 氏 (現 シャープ株式会社)，寺井 淳裕 氏 (現 株式会社 管理工学研究所)，松矢 明宏 氏 (現 日立公共システムエンジニアリング株式会社) に，心から深く感謝申し上げます.

本研究を進めるにあたり，貴重なご助言，ご指摘を賜りました，株式会社 SRA 先端技術研究所　阪井　誠 氏　に，心から深く感謝申し上げます.

本研究を進めるにあたり，貴重なご助言，ご指摘を賜りました，科学技術振興事業団さきがけ研究 21「機能と構成」領域　神谷 年洋 氏　に，心から深く感謝申し上げます.

本稿に至るまでの道のりにおいて，暖かなご支援を賜りました，学校法人塚本学院　塚本　邦彦 学院長　に，心から深く感謝申し上げます.

本研究を進めるにあたり，心理学の観点から分析に関する貴重なご助言を賜りました，大阪芸術大学　深田 尚彦 学長　に，心から深く感謝申し上げます.

本研究を始めるにあたり，貴重なご助言，暖かい激励を賜りました，大阪芸術大学大学院 芸術文化研究科　新井 基祐 教授　に, 心から深く感謝申し上げます.

本研究を進めるにあたり，ここに書ききれなかった多くの方々から有用なご指導，ご支援を賜りましたことに，心から深く感謝申し上げます.

最後に, 精神的にも時間的にも苦しかった本稿に至る道のりにおいて, 3 年間, 影ながら支えてくれた　妻 順子　と, 父親としての責務を果たせなかった私に明るく接してくれた　2 人の娘　と，暖かく黙って見守って下さった　両親　に心から感謝します.

参考文献

[1] Peter W.Airasian and William M.Bart: Ordering Theory: A new and useful measurement model, *Education Technology*, pp.56-60, May 1973.

[2] Ronald J. Brachman, Hector J. Levesque: Reading in Knowledge Representation, San Mateo, CA., *Morgan Kaufmann Publishers*, 1980.

[3] 朴 炳昊, 木村成判, 利 殷碩, 白鳥則郎：LOTOS 言語教育支援環境のためのプロセスの誤り検出法, 電子情報通信学会論文誌, Vol.J80-D-Ⅱ, No.4, pp 961-971, 1997.

[4] Ernest Davis: Representation of Commonsense Knowledge, San Mateo, CA., *Morgan Kaufmann Publishers*, 1990.

[5] 藤原康宏, 植野真臣：OPEN TESTING SYSTEM の開発, 電子情報通信学会技術研究報告, Vol.96, No.500, ET96-103～113, pp.31-36, 1997.

[6] 長谷川聡, 山住富 他：プログラミング教育における制御構造のイメージと理解度について, 情報処理学会論文誌, Vol.39,No.4, pp1180-1183, 1998.

[7] 羽田野 宣彦：ビジネスガイド No.301, 日本法令, pp.23-28, 1986.

[8] 萩原 勝:：派遣社員活用の手引き, 日経連広報部, pp.55-64, 1991.

[9] 伊藤公紀, 大内東：ファジィ項目関連構造分析による学習者の特性解析, 情報処理学会誌, コンピュータと教育 24-4, pp.27-34, 1992.

[10] 岩根典之, 竹内章, 大槻説乎：算数の文章題を対象としたネットワーク型知的教育支援環境, 電子情報通信学会論文誌, Vol.J80-D-Ⅱ, No.4, pp 915-924, 1997.

[11] 加藤鐐吉：経営情報化とオフィス・システム, 晃洋書房, pp.47-52, 1989.

[12] 経済企画庁 home page,
＜http://www5.cao.go.jp/j-j/wp-we/wp-we00/sekaihakusho-00-21.html＞
“平成 12 年度世界経済白書－IT 時代の労働市場と世界経済－（企業による職業訓練の対象と内容）

[13] 厚生労働省 home page,
＜ http://www2.mhlw.go.jp/kisya/syokuan/20001222_01_sy/20001222_01_sy.html ＞

“派遣労働者数107万人と大幅に増加ー労働者派遣事業の平成11年度事業報告の集計結果についてー,”労働省職業安定局民間需給調整事業室

[14] 厚生労働省 home page,
＜http://www.mhlw.go.jp/wp/hakusyo/roudou/01/2-6.html＞
“「平成 13 年版労働経済白書の分析」要約 第６節 情報通信技術革新に伴う雇用・就業形態の多様化,”

[15] Hideo.Kudo, Yuji.Sugiyama, Mamoru.Fujii, and Koji.Torii: Quantifying a design process based on experiments, *The journal of system and software 9*, pp.129-136, 1989.

[16] 鐵　健司：離散分布，応用統計ハンドブック，第 1 章，養賢堂，pp14-15, 1992.

[17] 鐵　健司：離散分布，応用統計ハンドブック，第 1 章，養賢堂，pp22-23, 1992.

[18] 桑原恒夫:例題中心学習における教材の知識構造の複雑さと理解の困難さとの関係，電子情報通信学会論文誌，Vol.J80-D-Ⅱ，No.11，pp.3039-3047, 1997.

[19] 松田憲幸，柏原昭博，平嶋　宗，豊田順一：プログラムの振舞いに基づく再帰プログラミングの教育支援，電子情報通信学会論文誌，Vol.J80-D-Ⅱ，No.1，pp.326-335, 1997.

[20] 溝口理一郎:知的教育システム, 情報処理学会誌, Vol.36, No.2, pp.177-178, 1995.

[21] 溝口文雄：概観：知識表現と知識プログラミング，認知科学ハンドブック，第Ⅵ編，共立出版，pp.253-268, 1995.

[22] 文部省委託調査:情報技術人材に対する産業界ニーズの動向に関する調査研究, 日本工業教育協会, pp.4-8, 1990.

[23] 中井節雄：日本的経営と人材確保・育成戦略，日本労務学会年報，pp132, 1993.

[24] 中山　茂：Java プログラミング徹底演習，日刊工業，pp.1-173，1998.

[25] 西田豊明，荒木雅彦：空間的知識の表現と利用，認知科学ハンドブック，第Ⅵ編，第 3 章，共立出版，pp.293-294, 1995.

[26] 日本情報処理開発協会編:情報化白書 1990：コンピュータ・エージ社, pp391, 1990.

[27] Joseph O'Nell：独習 Java, 武藤健志 監修, 株式会社翔泳社, pp.1-76, 1999.

[28] 大谷卓史, 武藤健志：はじめての Java, 技術評論社, pp.1-131, 1996.

[29] 佐々木 整, 小野慶一, 佐竹 誠：IM 法を利用した学習目標系統図の作成, 電子情報通信学会技術研究報告, Vol.96, No.431, ET96-87～102, pp.87-94, 1996.

[30] Roger C. Schank: Conceptual Information Processing, *New York: North-Holland*, 1975.

[31] 政府関係資料：高度情報化政策と新技術株式会社, 第 2 編 2 章ソフトウェア市場の環境の整備, 産業技術会議, pp204-208, 1993.

[32] 政府関係資料：高度情報化政策と新技術株式会社, 第 5 編 3 章情報化のための人材像, 産業技術会議, pp723-724, 1993.

[33] 政府関係資料：高度情報化政策と新技術株式会社, 第 5 編 3 章情報化のための人材像, 産業技術会議, pp725-730, 1993.

[34] 芝 祐順, 渡辺 洋, 石塚智一：統計用語辞典, 新曜社, pp.11-12, 1999.

[35] Elliot Soloway and Kate Ehrligh: Empirical studies of programming knowledge, *IEEE Transactions on Software Engineering*, SE-10 (5), pp.595-609, 1984.

[36] Sun Microsystems, Inc.：Java プログラミング講座, 株式会社アスキー, pp.12-79, 1998.

[37] 武村泰宏, 下左近多喜男: コンピュータ技術者の技術教育と派遣労働について, 日本経営システム学会論文誌, Vol.12, No.1, pp.1-7, 1995.6.

[38] Yasuhiro Takemura and Haruhisa Yamagichi: An evaluation of learning effect for education form by means of computer communication with foreign country, *Proceedings of the International Conference on Technology Education in School around Asian Countries*, pp.165-168, Sep. 1995.

[39] Yasuhiro Takemura and Haruhisa Yamagichi: An evaluation of learning effect for education form by means of applying both international personal

computer communication and translation system, *Proceedings of the International Conference on Technology Education in School around Asian Countries*, pp.169-172, Sep. 1995.

[40] Haruhisa Yamagichi and Yasuhiro Takemura: Learning result evaluation by fuzzy learning model, *Proceedings of the International Conference on Technology Education in School around Asian Countries*, pp.181-184, Sep. 1995.

[41] Yasuhiro Takemura and Takio Shimosakon: A study on home use information system by multimedia, *Proceedings of the 14th International Conference on Production Research*, Vol.2, pp.1344-1347, Aug. 1997.

[42] Yasuhiro Takemura, Kazuyuki Shima, Ken-ichi Matsumoto, Katsuro Inoue and Koji Torii: Factor analysis of comprehension states in the learning phases of a programming language, *Proceedings of the 6th Asia-Pacific Software Engineering Conference (APSEC'99)*, pp.136-143, Dec. 1999.

[43] 武村泰宏, 島 和之, 松本健一, 井上克郎: Java 言語の理解順序に関する関連構造の統計的分析, 電子情報通信学会技術研究報告, ソフトウェアサイエンス研究会, SS2000-53, Vol.100, No.677, pp.1-9, 2001.3.

[44] 竹谷　誠, 佐々木　整：学習者描画の認知マップによる理解度評価, 電子情報通信学会論文誌, Vol.J80-D-Ⅱ,No.1, pp.336-347, 1997.

[45] 田村進一, 柳原圭雄, 唐沢博：人工知能の世界, 技術評論社, pp.169-170, 1992.

[46] 寺井淳裕, 内田眞司, 島 和之, 武村泰宏, 松本健一, 井上克郎, 鳥居宏次: ソースコードの並び替えによるソフトウェアの問題発見手法, 電子情報通信学会技術研究報告, ソフトウェアサイエンス研究会, SS2000-52, Vol.100, No.570, pp.81-88, 2001.1.

[47] 寺野隆雄：常識とコンピュータ・システム, 認知科学ハンドブック, 第Ⅵ編, 第 2 章, 共立出版, pp.284-285, 1995.

[48] 地域経済総覧'93：東洋経済, pp.8-9, 1993.

[49] 通商産業省通商産業省機械情報局監修：ソフト人材の地域展開, ”商産業調査会, pp.11-16, 1990.
[50] 通商産業大臣官房調査統計部編:特定サービス産業実態調査報告書：通商産業調査会, 情報サービス業編　pp.10-11, 1992.
[51] 通商産業省通商産業省機械情報局監修 情報サービス産業協会編：情報サービス産業白書 1992, コンピュータ・エージ社, pp.115-117, 1992.
[52] 通商産業省通商産業省機械情報局監修 情報サービス産業協会編：情報サービス産業白書 1993, コンピュータ・エージ社, pp.304-305, 1993.
[53] 通商産業省通商産業省機械情報局監修 情報サービス産業協会編：情報サービス産業白書 1993, コンピュータ・エージ社, pp.62-64, 1993.
[54] 植野真臣, 河竹千春, 大西 仁, 繁桝算男：ネットワーク型テスト理論における構造最適化法,　電子情報通信学会技術研究報告, Vol.93, No.541, ET93-124〜145, pp.101-108, 1994.
[55] 内田眞司, 寺井淳裕, 武村泰宏, 阪井 誠, 島 和之, 松本健一: ソフトウェアの理解性向上によるデバッグ時間の短縮, 情報処理学会第 63 回全国大会講演論文集, 6H-05,pp.1-101-102, 2001.9.
[56] 吉川　厚：プログラム理解課程を調べた認知実験報告(1), 人工知能学会研究会資料, SIG-IES-9202, pp.53-61, 1992

付録

A. テストの内容(a)

A. Java 言語に関する設問について適切な回答を，回答群 (a) ～ (e) から選びなさい.

1. 自転車クラスにおいて，フレーム長，車輪径，ギアは何を意味しているか.
   (a) 情報隠ぺい (b) 属性 (c) 継承 (d) 機能 (e) 分類
2. 犬クラス，猫クラス，馬クラスのスーパークラスはどれか.
   (a) シェパードクラス (b) 哺乳類クラス (c) サラブレッドクラス (d) ペルシャ猫クラス (e) 足クラス
3. クラスごとに異なる値をとる変数はどれか.
   (a) クラス変数 (b) インスタンス化 (c) クラスメソッド (d) クラスライブラリ (e) パッケージ
4. クラスの階層関係に基づいて，クラス間で属性と操作を共有することを何と言うか.
   (a) アイデンティティ (b) 属性 (c) ポリモーフィズム (d) 継承 (e) 分類
5. あるクラスに含まれるクラスを何と言うか.
   (a) パッケージ (b) スーパークラス (c) サブクラス (d) クラスメソッド (e) インタフェース
6. 多角形クラスにおいて，辺，境界色，内部色は，何を意味しているか.
   (a) 情報隠ぺい (b)属性 (c) 継承 (d) 機能 (e) 分類
7. 野球ボール，バレーボールのスーパークラスはどれか.
   (a) 野球クラス (b) 運動クラス (c) ボールクラス (d) バレークラス (e) 球技クラス
8. オブジェクトの属性を示す「変数」と，振舞を示す「メソッド」を宣言するものを何と言うか.
   (a) メソッド (b) オブジェクト (c)クラス (d)インスタンス (e) プロトタイプ
9. 具体的な実体を示すオブジェクトを何と言うか.
   (a) オブジェクト (b) クラス (c) メソッド (d) プロトタイプ (e) インスタンス
10. オブジェクトの振舞を示すものを何と言うか.
    (a) インスタンス (b) プロトタイプ (c) オブジェクト (d) メソッド (e) クラス
11. インスタンスごとに異なる値をもつ変数を何と言うか.
    (a) オブジェクト変数 (b) メソッド変数 (c) クラス変数 (d) インスタンス変数 (e) パブリック変数
12. あるクラスを含むクラスを何と言うか.
    (a) パッケージ (b) スーパークラス (c) サブクラス (d) クラスメソッド (e) インタフェース
13. クラスの振舞を示すものを何と言うか.
    (a) パッケージ (b) クラスメソッド (c) インタフェース (d) スーパークラス (e)サブクラス
14. 属性の共通性はないが，振舞に共通性があるクラスを示すものを何と言うか.
    (a) パッケージ (b) スーパークラス (c) インタフェース (d) サブクラス (e) クラスメソッド
15. クラスやインタフェースの名前の衝突を避けるために，それらをグループ分けする単位を何と言うか.
    (a) インタフェース (b) パッケージ (c) クラスメソッド (d) サブクラス (e) スーパークラス

B. テストの内容(b)

B. 下記のプログラムについて各設問に答えなさい.

```
//"Point"クラスの定義
 [ B1.1 ]  Point {
 //変数の定義
 int x, y;
      //メソッドの定義
 public String toString() {
          return "("+x+","+y+")";
 }
 void set(int a, int b){          //変数に値を設定するメソッド
          x = a;
          y = b;
 }
 int getx(){//x の値を得るメソッド
          return x;
 }
 int gety(){//y の値を得るメソッド
          return y;
 }
 public static void main( [ B1.2 ]  args[]){
          int x=20, y=30;
          Point obj = [ B1.3 ] Point();        // インスタンスの生成
                    obj. [ B1.4 ] (10, 20);
          System.out.println("obj="+ [ B1.5 ] );// 表示 (B2.1)
          System.out.println("x="+x); // 表示 (B2.2)
          System.out.println("y="+y); // 表示 (B2.3)
          // x,y それぞれの値を得て、表示する
          System.out.println("getx: "+obj.getx());// 表示 (B2.4)
          System.out.println("gety: "+obj.gety());// 表示 (B2.5)
 }
}    //"Point"クラスの定義の終わり
```

B1.1～B1.5　プログラム中の５つの空欄 [ ] を埋めなさい.

B2.1～B2.5　プログラムの５つの出力結果を記述しなさい.