K-035

問題文並び替えとそれを反映したコード作成による学習者能力の判定

Judgment of Learner Ability from Exercise Sentence Sorting and Corresponding Coding

板戸 陽子 † 原田 史子 ‡ 島川 博光 ‡
Yoko Itado Fumiko Harada Hiromitsu Shimakawa

1. はじめに

プログラミングに必要な能力は，文法知識とコンピュータの知識，制御構造に落としこむ力の 3 つと考えられる．文法知識とコンピュータの知識は演習せずに学べるが，制御構造に落としこむ力は演習の過程で体得するものである．そのため大学のプログラミング教育では，文法知識とコンピュータの知識は座学で，制御構造に落としこむ力は演習で教える．そのさい，学習者が制御構造に落としこむ力を身につけられたかを判別する必要がある．

本論文では，学習者の制御構造に落としこむ力を，処理の流れを考える力とソースコードに落としこむ力に分割して評価する手法を提案する．本手法を用いた演習において，学習者はソースコードを書く前に処理の流れを考える．さらに指導者は，学習者が処理の流れを考える段階で行き詰まったのか，ソースコードを書く段階で行き詰まったのかを把握できるため，指導が効率化される．

2. プログラミング教育における機能分割能力

2.1 大学でのプログラミング教育における問題点

プログラミング演習において支援すべき能力は，制御構造に落としこむ力である．制御構造に落としこむ力は，処理を考える力と，考えた処理をソースコードに落とす力の 2 つに分かれる．ここで，制御構造に落としこむ力を機能分割能力，考えた処理をソースコードに落とす力を表現力とする．演習授業において，それぞれの力を伸ばす必要がある．しかし現在のプログラミング演習は，学習者が完成させたソースコードのみを提出する課題が，大多数である．そのため，機能分割能力が低い学習者は処理を考えきる前にソースコードを書き始めてしまい，計画性のないプログラミングをしてしまう．それにより，学習者は本来の出題意図に即した学習ができなくなる．また表現力が低い学習者は，処理を考えられても，その処理をソースコードに表すことができない．それにより，指導者が学習者の作成したソースコードを参考にして指導するとき，その学習者が処理を考えられなかったのか，処理をソースコードに落とせなかったのか判別しづらい．

2.2 機能分割の重要性

一般的にプログラムを作成するさいには，実装すべき内容を考え，その内容をどのような処理を組み合わせて実現すべきかを考える．その後，各処理に対してソースコードを記述する．演習授業の場合，実装すべき内容は問題文に記載されているため，学習者は問題文の意味を読み取り，処理の流れを考える必要がある．処理の流れを作るためには，どのような処理をするべきか考え処理を切り分けた後，切り分けた処理を並び替える．このように，処理を切り分けて考えることを機能分割という．

機能分割を用いたプログラミングでは，処理の流れを考えるさい言語仕様を考える必要がない．よって，処理の流れを構築することに集中できる．また，ソースコードを記述するさい，局所的な問題解決のみに絞って考えられる．そのため，ソースコードを書くさいには，機能分割をしてからソースコードを書くことが求められる．

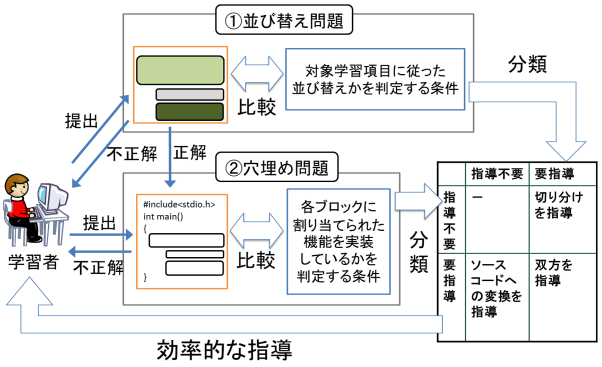

図 1: 手法の全体像

2.3 関連研究

関連研究 [1]–[2] では，学習者が提出したソースコードを解析し，学習者がどのような処理の流れを想定しソースコードを記述したか評価する．しかし，学習者がソースコードを完成させるまで解析できないため，ソースコードを書き始められない学習者に対応できない．また，提出したソースコードのみで評価をするため，処理の流れを考えられない学習者なのか，処理の流れをソースコードとして表現できない学習者なのかを判別できない．

関連研究 [3]–[5] では，擬似言語やフローチャートを用いて処理の流れを構築させ，学習者の処理を考える力を養う．しかし，新たにこれらの表記法などを習得する必要があり，新規学習者への負担が大きい．

関連研究 [6]–[8] では，学習者に自然言語で処理の流れを記述させ，機能分割を支援している．しかし，記述の自由度が高いため，出題者の意図と異なった処理の流れを作成していないか，指導者が判断することが難しい．

3. 学習者の機能分割能力と表現力の把握

3.1 機能分割能力と表現力による学習者の分類

本論文では，学習者のプログラミング能力を機能分割能力と表現力に分割し評価する手法を提案する．本手法では，ソースコードを記述する前に文のブロックを並び替えさせる．ブロックとは，意味の切れ目で区切られた文を表す．次に，ブロックごとに対応するソースコードを記述させる．これらの解答を分析し，学習者の機能分割能力と表現力を評価する．これにより，指導者は学習者の機能分割能力と表現力を踏まえた指導ができる．

本手法の全体像を図 1 に示す．まず，学習者は機能分割能力を判別する日本語の並び替え問題を解き，システムに提出する．不正解と判定された学習者は，正解するまで並び替え問題を解き続ける．正解した学習者は，表

†立命館大学大学院情報理工学研究科
‡立命館大学情報理工学部

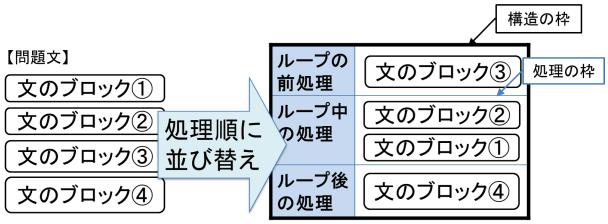

図 2: 並び替え問題

現力を判別する穴埋め問題を解く．これらの解答において，指導を必要とする間違いをしたかを判定し，学習者を分類する．学習者を分類することで，指導者は学習者が指導を必要とする点に絞った指導が可能となる．

3.2 並び替え問題

まず，学習者に機能分割能力を判別する並び替え問題を解かせる．並び替え問題とは，処理が書かれた文のブロックをソースコードで書く順番に並び替える問題である．ソースコードを書く前に処理の流れを組み立てさせ，処理の流れを考えられたか判定する．学習者の機能分割能力を計るさい，問題文の並び替えだけでは，不十分な場合もある．なぜなら，反復構造や関数を含む問題は，繰り返す処理や関数化する処理を選ぶことが重要であるからだ．そこで，これらの問題を階層構造を持つ問題とし，このような構造を持つ問題を解かせるさいは，並び替えをする前に必要な構造を選択させる．それにより学習者は，プログラムの構造について段階的に考えられる．

並び替え問題の概観を図 2 に示す．並び替え問題では，文のブロックと処理を表す枠の集合である構造を表す枠が与えられる．段階構造を持つ問題の場合は複数の処理の枠が与えられる．段階構造を持たない問題の場合，処理の枠はひとつとなる．学習者は文のブロックを処理を示す枠に当てはめ，その中でソースコードで書くべき処理の順番に並び替える．たとえばループ構造に振り分ける場合，ループの前処理，ループ中の処理，ループ後の処理と 3 つの処理に振り分ける．このように，適切な構造に文のブロックを当てはめ並び替えることで，学習者が実際の処理の流れを考えられたかを判定する．

また，課題の意図を満たした正しい処理の順序は 1 種類とは限らない．そのため，文のブロックを正しい処理の枠に当てはめられたかと，処理の枠内でのブロックの順番が設定した条件を満たしているかの 2 つを正解条件とする．またこれらの条件には，出題した問題において要となる条件と，入力の後に出力をするなどの自明な条件がある．出題した問題において要となる条件を，対象学習項目とする．自明な条件に対する間違いは，問題文を読み直せば改善できる可能性が高いため，間違いを通知することによって改善できると考えられる．しかし，対象学習項目は出題した問題特有の構造を持つため，一度正しいと思い込んでしまうと自力で修正することが困難である．そのため，対象学習項目を間違えた学習者を指導が必要な学習者とする．このような学習者は，問題文から出題者の意図を読み取れなかったと考えられるため，全体的な処理の構築方法を指導する必要がある．

3.3 穴埋め問題

並び替え問題に正解した学習者には，表現力を判別する穴埋め問題を解かせる．穴埋め問題とは，各ブロックに対し，対応するソースコードを書く問題のことである．この結果より，学習者がひとつの処理の流れに対して過不足なくソースコードを記述できるかを判定できる．

穴埋め問題では，並び替え問題で作成したブロックに対応した部分が空白になっているソースコードのひな形が与えられる．学習者はブロックごとにその内容のみを考え，ソースコードを書く．ここで個々の処理をソースコードで表せない学習者は，個々のブロックを理解していないと考えられる．そのため，処理ごとに正誤を見ることで，学習者の各処理に対する表現力を計ることができる．また部分的に個々の処理をソースコードで表現できていれば，学習者がソースコードとして表現できる処理とそれ以外の処理を指導者は容易に判別できる．

各ブロックが表す処理以外の処理を記述している学習者は，各ブロックの処理を細かく理解していないと考えられる．そのような学習者は正解に辿り着いたとしても，サンプルソースを改変して作成したなど，問題文の本質を理解していない可能性が高い．そのため，各ブロックに対応したソースコードが記述できていない学習者を指導が必要な学習者とする．このような学習者には，各処理の内容を理解できるよう，指導する必要がある．

3.4 学習者の分類

並び替え問題と穴埋め問題，それぞれの正誤をあわせて学習者を分類することで，学習者を 4 つに分類することができる．両問題とも指導不要であった学習者は，機能分割能力と表現力，両方の力を併せ持つと考えられる．そのため特別な指導の必要はない．並び替え問題は指導不要であったが，穴埋め問題で要指導とされた学習者は，全体的な処理の流れは構築できるが，個々の処理に対して必要なソースコードを表現できないと考えられる．そのため，各処理に対して適切なソースコードを書けるよう，指導する必要がある．ここで，並び替え問題で要指導とされた学習者は，処理の流れを考えられていない．そのためその時点で問題の切り分け方を指導をする必要がある．並び替え問題に対する指導後，穴埋め問題で指導不要とされた学習者は，処理の流れを考えられさえすれば，ソースコードとして表現できる．そのため，処理の流れを考えることに重点をおいた指導をすれば良い．並び替え問題に対する指導後，穴埋め問題でも要指導とされた学習者は，表現力も不足していると考えられる．そのため，両方の点において指導する必要がある．

4. 実験

4.1 実験概要

本研究の有用性を検証するために並び替え問題と穴埋め問題を提示，評価するシステムを実装し，立命館大学情報理工学部 1 回生 9 名に対し実験をした．そのさい，以下のような二重ループを用いて実装することが求められている選択ソートの課題を出題した．処理の流れとして書かれた文が学習者が切り分ける文である．学習者が二重ループ構造を含む問題において，どの処理を繰り返すべきか正しく読み取ることができるかを題材とした．

> ソートすべき数字を標準入力から読み込み，選択ソートをした後，昇順で表示するプログラムを作成せよ．なお，ソートすべき数字の個数は 10000 を超えないものとしてよい．

処理の流れ　まず，ソートすべき数字の個数を標準入力から読み込む．次に，改行区切りで表された数字を標準入力から受け付ける．そして，まだ整列されていないデータ列の 1 番目の要素を，もっとも小さい値として記憶する．次に，まだ整列されていないデータ列を繰り返し走査することで，もっとも小さい値を探した後，もっとも小さい値をまだ整列されていないデータ列の先頭の値と入れ替える．これをデータ列の最後まで繰り返す．最後に，整列が終了した配列を小さい方から順に表示する

まず，立命館大学情報理工学部 4 回生と大学院情報理工学研究科 1 回生，計 4 名に本課題を出した．その結果，2 名の学生は 10 分以内に実装を終えた．10 分以内に実装が終わらなかった学生は，実装途中で行き詰まり，指導者に質問をした．これより，自力で解ける学習者は 10 分程度の時間があれば解くことができると考えられる．

そのため，1 回生に対する実験においても 10 分程度の時間を設け，本課題を解けるかテストした．その結果，すべての学習者が正解できなかった．正解できなかった学習者は，機能分割能力または表現力が不足している学習者が含まれていると考えられる．その力を判定するため，正解できなかったすべての学習者に，並び替え問題と穴埋め問題を出題した．

4.2　並び替え問題

まず学習者は，使うべき繰り返しの構造を最初に選択する．学習者は本課題が何重ループを用いて実装すべき問題かを考え，使用する構造を選択する．構造を選択した学習者は，各ブロックをクリックして選択し，処理を示す枠内にドラッグ & ドロップさせて並び替えをする．このとき，学習者は任意のタイミングで構造の選択をやり直すことができる．これは，学習者が並び替え問題を解く過程で問題文を読み直すことにより，構造の選択の過ちに気づいた場合，やり直せることが，機能分割能力を向上させるうえで有効であると考えたためである．また，構造の選択を誤った上で並び替え問題を解き続けるなど，自力で直すのが困難と考えられる間違いをしている学習者に対して，指導者を呼ぶように警告を出す．その結果，指導者は適切なタイミングで学習者に指導できる．

4.3　穴埋め問題

並び替え問題に正解した学習者は穴埋め問題に取り組む．最終的にシステムの支援なしで学習者がプログラミングをできるようになることが理想である．そのため，穴埋め問題を解く手順を，システムのサポートなしにコーディングする状態に近づけることが重要である．そこで，学習者に処理の流れを表す文のブロックをソースコード上にコメントとして埋め込み，コード自体は空白としたソースコードのひな形を提供した．学習者は，このひな形を普段使用しているエディタを用いて編集し，ソースコードを完成させる．また，学習者は普段の演習で用いているデバッグツールなども使用する．最終的に学習者が提出したソースコードを評価の対象とする．

5.　評価

5.1　並び替え問題の結果

本問では，内側のループを抜けた後に，変数同士を入れ替えるべきである．そのため，内側ループの外で変数の入れ替えができているかを対象学習項目とし，指導が必要な学習者を判定した．その結果，9 名の被験者のうち，4 人が指導を必要とする学習者と判別された．

表 1: スコア比率に対する T 検定の結果

分類方法	スコア比率平均		p 値
	指導不要	要指導	
$\frac{GE}{NGE}$	0.2	0.17	0.049
（GE）$\frac{400\text{ 字以上}}{400\text{ 字以下}}$	2.8	4.18	0.034
（NGE）$\frac{300\text{ 字以上}}{300\text{ 字以下}}$	0.56	0.62	0.039
$\frac{\text{数式なし}}{\text{数式あり}}$	7.71	6.78	0.051
$\frac{\text{コードや実行例なし}}{\text{コードや実行例あり}}$	0.24	0.21	0.029

機能分割能力が低い学習者は，説明が短い問題などの処理の流れを考える必要性が高い課題が苦手と考えられる．そこで，C 言語の基礎的な演習をする授業の課題 153 題を，要する機能分割能力の観点から分類した．ここで，立命館大学では，文献 [9] で示されているグラフィカルツールを用いる課題 (以下 GE とする) がある．GE では，問題文中に図を用いた説明があり，その図をグラフィカルツールを用いて描くことが要求される．出題方法の傾向がグラフィカルツールを用いない課題（以下 NGE とする）と異なるため，しきい値を分け検証した．また，GE において学習者は絶対座標で表されている問題文を読み解き，相対座標で実装しなければならない．そのため，処理の順番を考える力がない学習者は GE を解くことができないと考えた．

ここで，課題の難易度によって重み付けされた点数の総計をその課題種別に対するスコアとする．並び替え問題において指導が必要とされた学習者は，全体的にスコアが低い．そのため，分類した課題に対するスコアの比率を取り，比較した．その比率をスコア比率と呼ぶ．たとえば，GE におけるスコアが 100，NGE におけるスコアが 500 の学習者がいたとする．その場合，GE のスコアを NGE のスコアで除算した 0.2 を $\frac{GE}{NGE}$ と表す．これより，スコア比率が要指導学習者の方が低い場合はスコア比率の分子となった課題種別が苦手であると解る．また，スコア比率が要指導学習者の方が高い場合はスコア比率の分母となった課題種別が苦手であると解る．

要指導な学習者と指導不要な学習者のスコア比率の平均を表 1 に示す．要指導学習者のスコア比率が指導不要学習者に対し有意差があるか調べるため，スコア比率の高低に基づき，以下の項目を T 検定を用いて検証した．表 1 には T 検定における p 値の結果もあわせて示す．

- $\frac{GE}{NGE}$ のスコア比率が要指導学習者は，指導不要学習者に比べて低いか
- GE の $\frac{\text{問題文の長さが 400 文字以上}}{\text{問題文の長さが 400 文字以下}}$ のスコア比率が要指導学習者は，指導不要学習者に比べて高いか
- NGE の $\frac{\text{問題文の長さが 300 文字以上}}{\text{問題文の長さが 300 文字以下}}$ のスコア比率が要指導学習者は，指導不要学習者に比べて高いか
- $\frac{\text{数式なし}}{\text{数式あり}}$ のスコア比率が要指導学習者は，指導不要学習者に比べて低いか

- $\frac{\text{ソースコードや実行例なし}}{\text{ソースコードや実行例あり}}$ のスコア比率が要指導学習者は，指導不要学習者に比べて低いか

この結果，並び替え問題で要指導とされた学習者は以下の種別の課題が苦手と解った．

- GE．特に，処理の流れがあまり問題文に書かれておらず，図から処理を考える課題
- 問題文の文字数が少ない課題
- 実行例やサンプルとなるソースコードがない課題

また，数式のない課題も苦手な傾向が見られた．並び替え問題における要指導学習者は，処理の流れを自分で考える必要がある課題が苦手とであると考えられる．

5.2 穴埋め問題の結果

個々のコメントに従った処理を書いていないソースコードを提出した学習者を，表現力を高める指導が必要な学生とした．その結果，7 名中，2 名が指導を必要とする学習者と判別された．また，2 名の学習者は，最初の 10 分間に最後まで解けなかったが，書いた処理が多かったため，穴埋め問題に挑戦せずソースコードを完成させた．

不正解だった 2 名の学習者は，変数の値を入れ替える部分を内側ループの中に書くべきところ，内側ループの外に書いていた．この 2 名について，処理をソースコードとして表現することが苦手かを検証するため，C 言語の基礎的な演習をする授業におけるソースコードや取り組み方と比較した．その特徴を以下に示す．

- スコアやテストの得点は高い
- オプション課題にも積極的に取り組んでいる
- ブロックの先頭以外で変数宣言をするなど，C 言語の記法にない記述が見られる

これより，要指導学習者のプログラミング能力は高いと考えられる．しかし，授業で指導した記法に従わないなどの行動から，プログラムを完成させさえすれば良いと考えていると推定される．穴埋め問題における要指導学習者は，プログラミング能力は高いが，細かな指示に従わず，我流で取り組んでいる可能性が高いと解った．

5.3 分類された学習者ごとの特徴

実験の結果，並び替え問題，穴埋め問題，共に要指導となった学習者は 1 名もいなかった．これは，各ブロックの処理を理解した上でソースコードとして表現することができない学習者が被験者にいなかったと考えられる．

並び替え問題，穴埋め問題ともに指導不要とされた学習者は，機能分割能力と表現力，両方の力を持つと解った．そのため，特別な指導をする必要はない．並び替え問題は指導不要であったが，穴埋め問題で要指導とされた学習者は，全体的な問題への対処はできるが，細かい規約などを気に留めたコーディングができないと解った．そのため，規約などに従うよう，指導する必要がある．並び替え問題で要指導とされた学習者は処理の流れを考える問題が苦手であると解った．そのため，問題文から処理の流れを読み解く方法を指導する必要がある．

この結果のうち，並び替え問題は指導不要であったが，穴埋め問題で要指導とされた学習者の特徴が，実験前の想定と異なるものであった．これは，実験前に想定したタイプの学習者が被験者にいなかったことが考えられる．また，本タイプにあてはまる学習者には，個々の処理に対して必要なソースコードを考えられない学習者以外に，個々の処理を気に留めない学習者がいると解った．

6. おわりに

本論文では，学習者の機能分割力と，表現力を独立に評価する手法を提案した．本手法では，学習者に機能分割力を計る並び替え問題と表現力を計る穴埋め問題を順に与える．その結果を用いて学習者を 4 つのタイプに分類することで，学習者の力にあわせた指導が可能となる．

本手法に基づく実験の結果，機能分割能力が低い学習者は，処理をより考える必要がある課題が苦手かに対する T 検定の結果において優位な差が認められた．これより，並び替え問題で要指導となった学習者は，処理を構築するのが苦手と解った．表現力が低い学習者の演習授業に対するソースコードや取り組みを検証した．これより，穴埋め問題で要指導となった学習者は，我流でプログラミングに取り組んでいると解った．これらの結果，指導者は学習者の苦手な力を踏まえた指導が可能となる．

今後は，今回の実験で検証できなかった，機能分割はできるが表現力が低い学習者に対して，本手法が有用かを検証する予定である．さらに，本手法を用いた指導がもたらす教育効果についても検証する．

参考文献

[1] 小西達裕，鈴木浩之，伊東幸宏，プログラミング教育における教師支援のためのプログラム評価機構，電子情報通信学会論文誌. D-I, 情報・システム, I-情報処理，Vol.83， No.6， pp.682–692，2000

[2] 渡辺博芳，荒井正之，武井惠雄，事例に基づく初等アセンブラプログラミング評価支援システム，情報処理学会論文誌，Vol.42，No.1，pp.99–109，2001

[3] 新開純子，炭谷真也，プロセスを重視したプログラミング教育支援システムの開発，日本教育工学会，Vol.31，pp.45–48，2008

[4] 岡田信一郎，冨士池均，藤原祥隆，流れ図作成支援システムのための誤り診断機能の性能評価，情報処理学会論文誌，Vol.42，No.3，pp.614–623，2001

[5] 阿部清彦，大山実，大井尚一，コンピュータプログラミング学習のためのアルゴリズム自習システム，工学･工業教育研究講演会講演論文集，Vol.18，pp.140–141，2006

[6] 斐品正照，徳岡健一，河村一樹，構造化チャートを用いたアルゴリズム学習支援システム，情報処理学会論文誌，Vol.45，No.10，pp.2454–2467，2004

[7] 松澤芳昭，青山希，杉浦学，川村昌弘，大岩元,「目的の表現」に注目したオブジェクト指向プログラミング教育とその評価，情報処理学会研究報告. コンピュータと教育研究会報告，Vol.2003，No.123，pp.77–84，2003

[8] 新開純子，宮地功，手作業による体験的アルゴリズム学習の実践，日本教育工学会論文誌，Vol.35，pp.129–132，2011

[9] 谷川紘平，フォンディンドン，原田史子，島川博光，C 言語関数呼出しの記録を用いた演習過程での習得項目の把握，電子情報通信学会論文誌. D, 情報・システム，Vol.95，No.12，pp.2079–2089，2012