Vol.2016-CLE-19 No.6
2016/5/21

大学のプログラミング教育のためのルーブリックの検討

渡辺 博芳[†1,†2]　水谷 晃三[†1]　盛 拓生[†1]　荒井 正之[†1]
佐々木 茂[†1]　古川 文人[†1,†2]　高井 久美子[†1,†2]

概要：大学でのプログラミングの授業で活用するルーブリックの作成と予備的な評価について述べる．プログラミングの力は一つの科目で修得できるものではなく，関連する科目群を通して，学生が自分の力を把握しながら，学修を進めることが重要であると思われる．そこで，プログラミング教育のカリキュラムを通して利用するプログラミングルーブリックを作成した．また，在学生を対象とした 2 日間のプログラミング勉強会を実施し，その前後で学生にルーブリックを使った自己評価をしてもらった．その結果から，実際の授業で活用できる見通しを得た．

Development and Preliminary Evaluation of Programming Rubrics for Higher Education

HIROYOSHI WATANABE[†1,†2]　KOZO MIZUTANI[†1]　TAKUO MORI[†1]
MASAYUKI ARAI[†1]　SHIGERU SASAKI[†1]　FUMIHITO FURUKAWA[†1,†2]
KUMIKO TAKAI[†1,†2]

Abstract: We have developed a rubric available in the whole curriculum of programming education. This is because, it is important for students to learn while checking their own understanding through a group of programming subjects, in order to acquire the programming ability. We held a study session of programming for 2 days. Students who participated in the study session made self-assessments using the developed programming rubric before and after the session. The results showed that we could use the rubric in the actual classes.

1. はじめに

本学情報電子工学科では, 平成 28 年度入学者を対象に教育カリキュラムの大幅な改訂を行うべく，検討を進めてきた．その中でプログラミング教育についても教授内容と教授方法の見直しを行った．これまで，学習管理システム(LMS)の活用やインストラクショナルデザインによる授業設計[1], 学生補助員参加の教材作成[2]などの工夫を行ってきたが，4 年次の研究室配属時に十分なレベルのプログラムを作成できるようになっていない学生も少なくはない．そこで，今回のカリキュラム改訂において，プログラミング教育の改訂に注力した．

受講者が確実にプログラミングの力を獲得するためには, 自分で自分の力を把握しながら，主体的に学習することが重要である. そこで, 2 年次や 3 年次には PBL(Project Based Learning)型の科目を設置すると伴に，プログラミングの基礎的な教育プログラムにおいては次のような方針で授業を設計することとした．

- 受講者が定期的に自己評価を行い，自身の理解状況を把握しながら学習を進められるようにすること．
- 全員一律の課題ではなく，複数のレベルの課題を用意しておき，受講者が把握している自己のレベルに応じた課題に取り組むことができるようにすること．
- 反転授業の導入と学生補助員(SA)による少人数担当制を採用することで，授業時間に受講者が対話的で主体的な学習活動を行えるようにすること．

このような教育プログラムにおいて，受講生が自身の理解状況を把握するツールとして, ルーブリック(Rubric)を利用することが有効であると考えられる．ルーブリックは，縦方向に評価の観点，横方向に到達レベルあるいは尺度を持たせた表で，各セルに，各観点についてそのレベルの達成度を示す基準を具体的に記述したものである．

そこで，本学のプログラミング教育のカリキュラム全体で利用するルーブリックを作成し，在学生を対象としたプログラミング勉強会で試用して予備的な評価を行った．

2. 関連研究

プログラミング教育においては，個々のプログラミング課題を評価するためのルーブリックの作成と活用の例が多い．例えば，Bailie らは教育プログラムの学習目標から学生が達成すべき学習成果を明確化することで，それらに適合するプログラミング課題のための一般的なルーブリックを作成し，それをテンプレートとして具体的課題に合ったルーブリックを作成する例を述べている[3].

†1 帝京大学理工学部
Faculty of Science and Engineering, Teikyo University.
†2 帝京大学ラーニングテクノロジー開発室
Learning Technology Laboratory, Teikyo University.

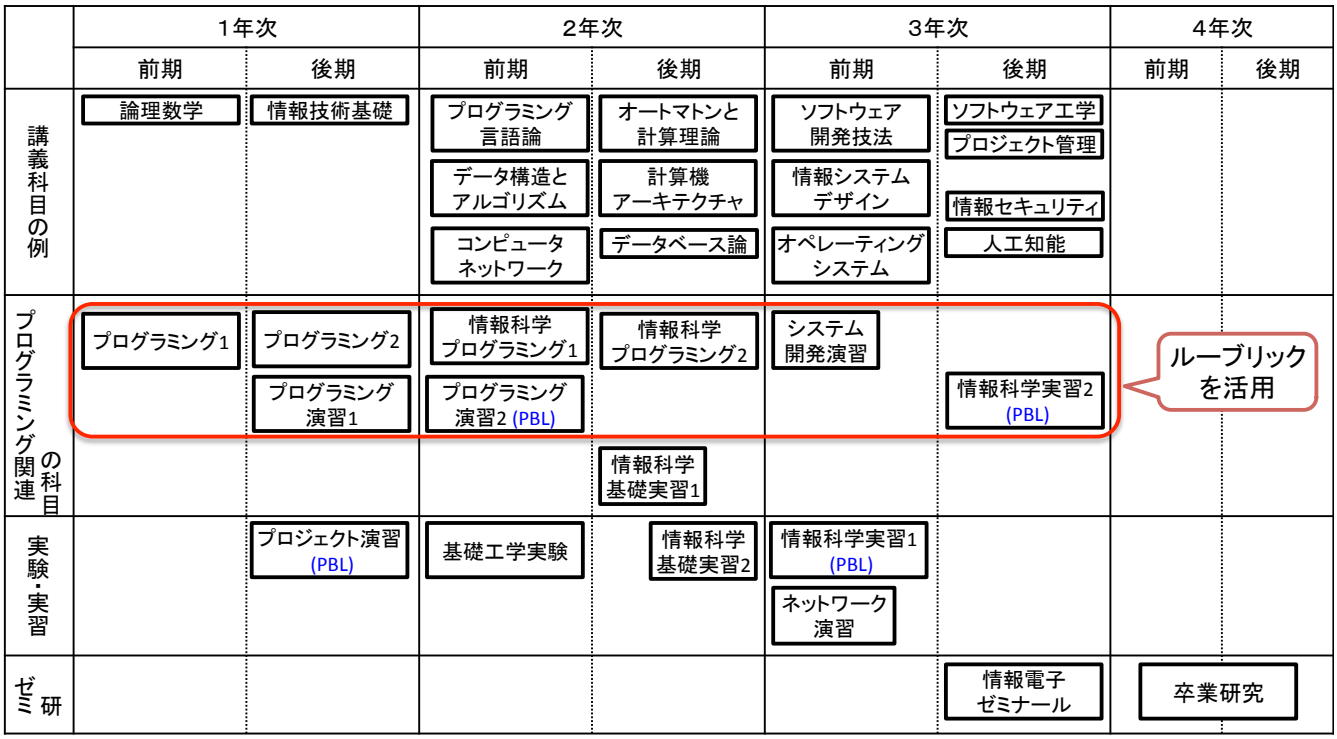


図 1　対象となるプログラミング教育カリキュラム

Figure 1　The target curriculum of programming education.

また，Mustapha らは，複数の教授スタッフによる評価の一貫性を確保するために，教育プログラムの学習成果と授業の学習成果のマッピングを基に，プログラミング課題の評価ためのルーブリックを作成した．授業では同じルーブリックを使って 2 名の教員または TA が評価を行った．実際の授業での約2500の提出物に対する2名の評価結果を重み付きカッパ係数により分析したところ，一致度が十分に高いことが示された[4].

このような研究は，ディプロマ・ポリシーやカリキュラム・ポリシーで示される学生が大学教育において獲得すべき力との整合性を保つように，プログラミング科目でどのような力を身に付けるべきかを明確化して，それにそった課題の評価を行うと言う点で重要なアプローチである．また，複数の評価者による評価結果のばらつきを抑える効果も示されており，有用性も高い．

これらに対して，本研究ではプログラム関連科目の受講者が自己評価を行う際に用いることを主な目的としており，著者らが調べた範囲では，同様なアプローチによるルーブリックの作成と活用の例は見あたらない．

3. 対象となるカリキュラム

対象となるカリキュラムを図 1 に示す．これは平成 28 年度入学者からの改訂後のカリキュラムである．図 1 における講義科目はカリキュラムの一部である．

1 年次に配置されている「プログラミング 1・2」と「プログラミング演習 1」では Processing 言語を使って，基礎的なプログラミングを学ぶ．

2 年次の「情報科学プログラミング 1・2」では Java 言語を使ってオブジェクト指向プログラミングを中心に学ぶ．「プログラミング演習 2」は，チームまたは個人で独自のプログラムを開発するような PBL 形式の授業を考えている．個人で開発する場合も，類似のプログラムを作成するメンバーで構成されるグループでの活動を取り入れる予定である．「情報科学基礎実習 1」はアセンブリ言語によるプログラミングを学ぶ．

3 年次は「情報科学実習 1」でシステム設計を PBL 形式で，「情報科学実習 2」でシステム開発を PBL 形式で行う予定である．「システム開発演習」では「情報科学実習 2」で行うシステム開発に必要な技術の演習を行う．

このようなプログラミング関連科目のうち，高級言語によるプログラミングを伴う科目を対象としたルーブリックを作成する．

4. プログラミング教育用ルーブリック

4.1 作成するルーブリックの目的

先に述べたように，本研究で作成するルーブリックは，プログラミング関連科目の受講者が自己評価を行う際に利用することを主要な目的としている．受講者が学習活動を振り返り，解いた問題や作成したプログラムをエビデンスとして，今の自身の力がルーブリックのどのレベルに該当

するかを判断する．そうした自己評価に基づいて，以降の学習に受講者自身が主体的に取り組むことを期待している．逆に言えば，受講者が自己評価をしながら，主体的に学習活動に取り組めるような授業を目指している．

本研究で作成するルーブリックは，次のような特徴を持つことになる．

(1) 3 年間の教育カリキュラムで共通に使用する

1 年次から 3 年次のプログラミング関連科目で利用するため，個々の課題でなく，プログラミング能力全般を評価するルーブリックである必要がある．また，学び初めの段階から自分でプログラムが作成できる段階までを表す．

(2) 毎回の授業で日常的に使用する

一方で，毎回の授業でも使用するため，毎回の授業で学習目標となる事項に関して自己評価をする際にも使える必要がある．

(3) 学生が自己評価において使用する

学生が使用するため，学生にとってわかりやすいルーブリックである必要がある．また，自分が該当するレベルを学生自身が判断しやすい仕組みも必要である．これについては，学生に課する課題にルーブリックの対応するレベルを明記しておくことで，その課題を解決できれば，ルーブリックの対応するレベルに到達できていることがわかるようにする．

4.2 作成したルーブリック

自分で十分なレベルのプログラムを作成できることを最終的な目標として，ルーブリックを作成した．ルーブリックの作成にあたっては，教育目標の分類体系[5]や ICE モデル[6]を参考にしてレベル分けを行った．しかし，ルーブリックの作成作業はシステマティックに行うことができた訳ではないことから，作成したルーブリックを示して，考慮した点を説明したい．

表 1 に作成したルーブリックを示す．本学におけるプログラミング教育全体で使えるように，3 年次終了時に達成すべきレベルまでを段階化することとした．さらに上級もあり得るが，このルーブリックでは対象としない．

4.2.1 評価の観点

図 2 の縦方向に配置した評価の観点として，まず，「コーディング」と「コードリーディング」を設定した．プログラムコードを書けるようになること，そのために他のプログラムコードを読んで理解できることが重要だからである．

次に，正しく動作するプログラムを作成するための「テストとデバッグ」，読みやすいプログラムを作成できるようにするための「可読性」という項目も設けた．これらは，本来「コーディング」に含まれるが，コーディングで考慮すべき事項が多くなってしまうことから，別の観点として位置付ける方が望ましいと判断した．

表 1　プログラミング教育用のルーブリック
Table 1　The Rubric for Programming Education.

規準	レベル0	レベル1	レベル2	レベル3	レベル4
データ構造とアルゴリズム	レベル1に満たない	(基本的なデータ構造と処理を理解している．) ・データ構造にはどのようなものがあるか言える． ・典型的な処理にはどのようなものがあるか言える．	(基本的なデータ構造と処理について深く理解している．) ・どのようなときに，どのデータ構造を使うかを言える． ・どのようなときに，どの処理を使うかを言える．	(一般的な表現で処理手順を構成できる．) ・与えられた仕様に対するデータ構造を構成できる． ・与えられた仕様に対するアルゴリズム・処理手順を書ける．	(問題を解決するデータ構造とアルゴリズムを分析できる．) ・与えられた仕様に対する複数のデータ構造，複数の処理手順をあげて，それぞれの長短を説明できる．
コーディング	レベル1に満たない	(基本的な文法を理解している．) ・1行程度の日本文での表現に対応するコードが書ける．	(基本的な文法と処理手順の関連を理解している．) ・与えられたデータ構造と処理手順を基にプログラムコードを書くことができる． ・プログラムの空欄を埋めることができる．	(基本的なプログラミングができる．) ・与えられた仕様に対して，データ構造と処理手順を自分で考えて，プログラムを作ることができる．	(実際的なプログラムを作ることができる．) ・問題状況に対して，プログラムの仕様を決めることができる． ・与えられた仕様に対して，可読性が高く，正しく動作するプログラムを作ることができる．
コードリーディング	レベル1に満たない	(基本的な文法を理解している．) ・1行程度のコードを説明する日本文を書ける．	(基本的な文法と動作の関連を理解している．) ・プログラムの動作をトレースすることができる．	(プログラムコードを読んで理解することができる．) ・コメント無しに，プログラムコードの仕様を推定することができる．	(実際的なプログラムコードを理解し，説明することができる．) ・大規模なプログラムのコードを読んで，プログラムの仕様や処理内容を，他者に説明できる．
テストとデバッグ	レベル1に満たない	(プログラムの実行とテストについて基本事項を理解している．) ・テストデータとは何か説明できる． ・プログラムを入力し，(コンパイルし，)実行できる． ・コンパイラ・インタプリタが報告する文法エラーを取り除くことができる．	(プログラムの実行とテストについて理解している．) ・与えられた仕様をテストするテストデータを構成できる． ・容易に想定される原因による論理的エラーを取り除くことができる．	(基本的なテストとデバッグを行うことができる．) ・テストデータを使って，デバッグをすることができる．	(実際的なテストとデバッグを行うことができる．) ・特殊な条件下で発生するなど，容易には想定できない原因による不具合を解消することができる．
可読性	レベル1に満たない	(プログラムの可読性について基本事項を理解している．) ・可読性が高いプログラムがどのようなものか言える．	(プログラムの可読性について理解している．) ・可読性を高めるため，なぜそうするのかを説明できる．	(可読性の高いプログラムを書くことができる．) ・読みにくいプログラムコードの可読性を高めることができる． ・実際のプログラミングにおいて可読性の高いプログラムを書くことができる．	(プログラムの可読性について分析できる．) ・可読性を高めるコーディングルールを決めることができる．

表 2　教育目標の分類体系
Table 2　Taxonomy of Educational Objectives.

レベル	ブルーム	マルザーノ&ケンドール
1	知識	取り出し
2	理解	理解
3	応用	分析
4	分析	知識の応用
5	総合	メタ認知システム
6	評価	自律システム

表 3　ICE モデルの 3 段階
Table 3　Three Levels of ICE Model.

I (Ideas)	一つ一つの知識を学ぶ段階.
C (Connections)	学んだことと既に知っていることをつなげ，深い理解に至る段階.
E (Extensions)	学びの最終段階で学んだことを十分に自分のものにしている段階.

さらに，プログラムを作成するために前提となる「データ構造とアルゴリズム」についての理解も観点に含めることにした．

4.2.2 各観点での到達レベル

表 2 にブルームの分類体系とマルザーノらが構築した新しい分類体系[5]における 6 つのレベルを示す．レベル 1 が低いレベル，レベル 6 が高いレベルである．

作成したルーブリック(表 1)は 4 つのレベルとレベル 1 に満たないレベル 0 から成っており，レベル 1 からレベル 4 が概ね表 2 のレベル 1 からレベル 4 に対応する．具体的には，レベル 1 は文法等の知識を覚えるレベル，レベル 2 は例題等の理解ができているレベル，レベル 3 は知識を応用して一通りのことができるレベル，レベル 4 は分析することでより高度に知識を活用できるレベルであり，作成したルーブリックは表 2 のブルームの分類体系に近い．

次に ICE モデルとの関連を述べる．ICE モデルは，学びの段階を表したもので，I(Ideas)，C(Connections)，E(Extensions)の頭文字をとったものである．各段階の説明を表 3 に示す．詳しくは文献[6]を参照されたい．

作成したルーブリック(表 1)では，レベル 1 とレベル 2 が I(Ideas)の段階，レベル 3 が C(Connections)の段階，レベル 4 が E(Extensions)の段階に対応させたつもりである．

以上のように，作成したルーブリックの到達レベルの段階分けは，教育目標の分類体系や広く用いられている ICE モデルに適合していると言える．

4.2.3 学習項目ごとの評価

プログラミング能力の全体としては表 1 のルーブリックを用いて評価を行う．これは中間試験や期末試験などのタイミングで自己評価することを想定している．一方，毎回の授業における個々の学習項目についてもルーブリックを用いて自己評価を行う．具体的には，各回の授業において学んだ学習項目について「コーディング」と「コードリーディング」の 2 行を抜粋したルーブリックで，学生が自己評価を行う活動を取り入れる．

後述するプログラミング勉強会においては,「コーディング」のみに関して，学習項目ごとに自己評価を行ってもらった．具体的には，表 1 のルーブリックを表面に，付録 A.1 の評価シートを裏面にした自己評価シートを試用した．

5. プログラミング勉強会の実践

5.1 プログラミング勉強会の概要

在学生を対象とした 2 日間のプログラミング勉強会を実施した．プログラミング勉強会の目標は次の通りである．

- 次の年度から行う授業の学生補助員(SA)の候補にプログラミング言語 Processing に慣れてもらうこと．2015 年度時点の在学生は，1 年次からのプログラミング科目を Java 言語で学んでおり，Processing は授業では扱っていないためである．
- 在学生にプログラミングの学び直しの機会を与えること．SA の候補だけでなく，プログラミングを復習しておきたい在学生も対象とした．
- 作成したルーブリック，問題，および教材の改善のための情報を得ること．

まず，1 年次から 3 年次までの必修科目の授業の中で，上に述べた目的でプログラミング勉強会を行うことを周知し，ガイダンスの日程を告知した．ガイダンスは 2015 年 1 月 26 日(火)，1 月 27 日(水)の 2 回に分けて実施し，勉強会の進め方とプログラミング言語 Processing の概要を説明した．その後，参加を希望する者からルーブリックを使った自己評価シートを提出してもらった．他に，研究室に配属が決定した学生へガイダンスを行った研究室もあった．ガイダンスの参加者数を表 4 に示す．

プログラミング勉強会は全体として以下の流れをとった．

(1) 自己評価シート(ルーブリック)による事前の自己評価
(2) LMS(学習管理システム)に掲載された教材・課題による学習
(3) 自己評価シート(ルーブリック)による事後の自己評価，およびアンケートへの回答

表 4　プログラミング勉強会ガイダンスへの参加者数
Table 4　Numbers of Participants in Guidance Meetings for the Programming Study Session.

学年	1/26	1/27	研究室	合計
4 年次	0	1		1
3 年次	8(1)	1	9	18(1)
2 年次	1	8(7)		9(7)
1 年次	5	10(4)		15(4)
合計	14(1)	20(11)	9	43(12)

( )内は SA 希望者

表 5　プログラミング勉強会の時間割

Table 5　The Time Table of Programming Study Session .

10:00-12:00	学習時間(1)
12:00-13:00	お昼休み
13:00-15:00	学習時間(2)
15:00-15:30	休憩(おやつ)
15:30-17:00	学習時間(3)

表 6　プログラミング勉強会の参加者数

Table 6　Numbers of Participants in the Programming Study Session.

	2/9 出席	2/10 出席	両日出席	参加合計
人数	34	20	18	36

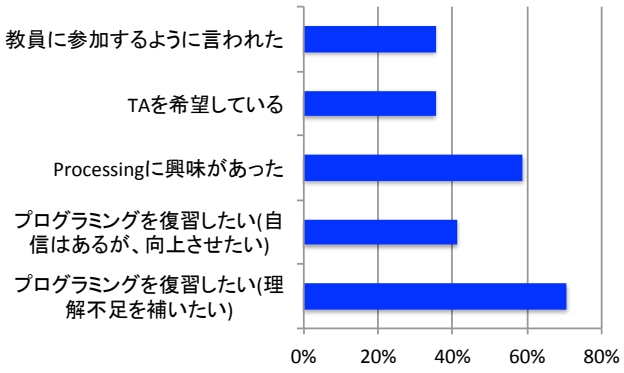


図 2　勉強会に参加した動機

Figure 2　Motivation to Participate in The Programming Study Session.

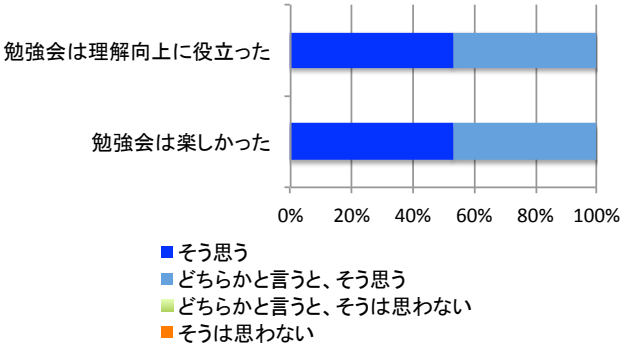


図 3　プログラミング勉強会についての質問への回答

Figure 3　Answers for Question on The Programming Study Session.

LMS には次の学習項目について，教材と課題を掲載した．1 つの学習項目はおよそ授業 1 回分の内容である．

(1)データ型・演算子，(2)変数，(3)for 文，(4)while 文，(5)if 文，(6)switch 文，(7)二重ループ処理，(8)関数，(9)一次元配列，(10)二次元配列，(11)テキストファイルの入出力，(12)テスト駆動開発，(13)デバッガの使い方，(14)文字列操作(課題のみ)

付録 A.1 に示したルーブリックの学習項目は新しいカリキュラムで 1 年次の「プログラミング 1・2」で学ぶ項目をリストアップしたものであるが，それら全ての学習項目はカバーすることができなかった．

勉強会の 1 日の時間割を表 5 に示す．おやつの時間は学生食堂の一角において，担当教員が用意したお菓子等をつまみながら，参加者の交流を図った．

学習時間(1)から(3)では，個人またはグループで学習に取り組み，課題に対するプログラムを作成したら，教員や友人に確認してもらうように指示した．上で述べた学習項目は，どれをやってもよいことにし，事前の自己評価の結果を基に，苦手なところを選んで学習するとよいとアドバイスした．ただし，ほとんどの学生は LMS に掲載されている順に学習を進めていた．また，学習時間毎に学習した内容や疑問点・教材の誤字脱字・分かりにくい点などを LMS の日誌機能へ投稿するように指示した．

2 日間の各学習時間には，3~5 名の教員がコンピュータ教室に待機し，来年度に使用する問題や教材の作成をしながら，ときどき巡回を行う程度で，積極的な教授活動はしなかった．

5.2 プログラミング勉強会に関するアンケート結果

プログラミング勉強会への参加者数を表 6 に示す．プログラミング勉強会後に行ったアンケートへの回答者は 17 名であり，回答者は 2 日目の勉強会に参加した学生ばかりである．これは，LMS 上のアンケートを回答可能な状態にしたのが，2 日目の最後であったためだと思われる．回答者 17 名のうち 2 日目のみに参加した学生は 2 名であった．勉強会に参加した動機(複数回答)の質問への回答を図 2 に示す．プログラミングの復習をしたいと言う学生が多いことがわかる．

プログラミング勉強会に関する質問への回答を図 3 に示す．有用性の質問，楽しさの質問とも全員がポジティブな回答であった．自由記述からは「課題の量もちょうど良く，学んだことを生かしてオリジナルのプログラムを組む時間ができたため楽しめた」「あたたかい感じで，自分のペースで進められたので勉強しやすかった」など自分で学べることが良いという意見が 4 件，おやつや交流タイムが良かったとする意見が 4 件あった．これらのことから，プログラミング勉強会全体としては成功だったと言える．

6. 作成したルーブリックの評価

6.1 ルーブリックに関するアンケート結果

先のプログラミング勉強会についてのアンケートの中で，ルーブリックについても質問した．ルーブリックに関する質問への回答を図 4 に示す．ルーブリックはわかりやすいと感じる学生やルーブリックが有用であると感じる学生が多い．自由記述においても以下のような意見があった．

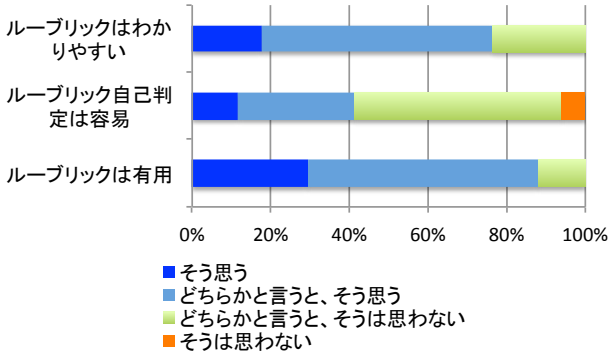


図 4　ルーブリックについての質問への回答

Figure 4　Answers for Question on Rubric.

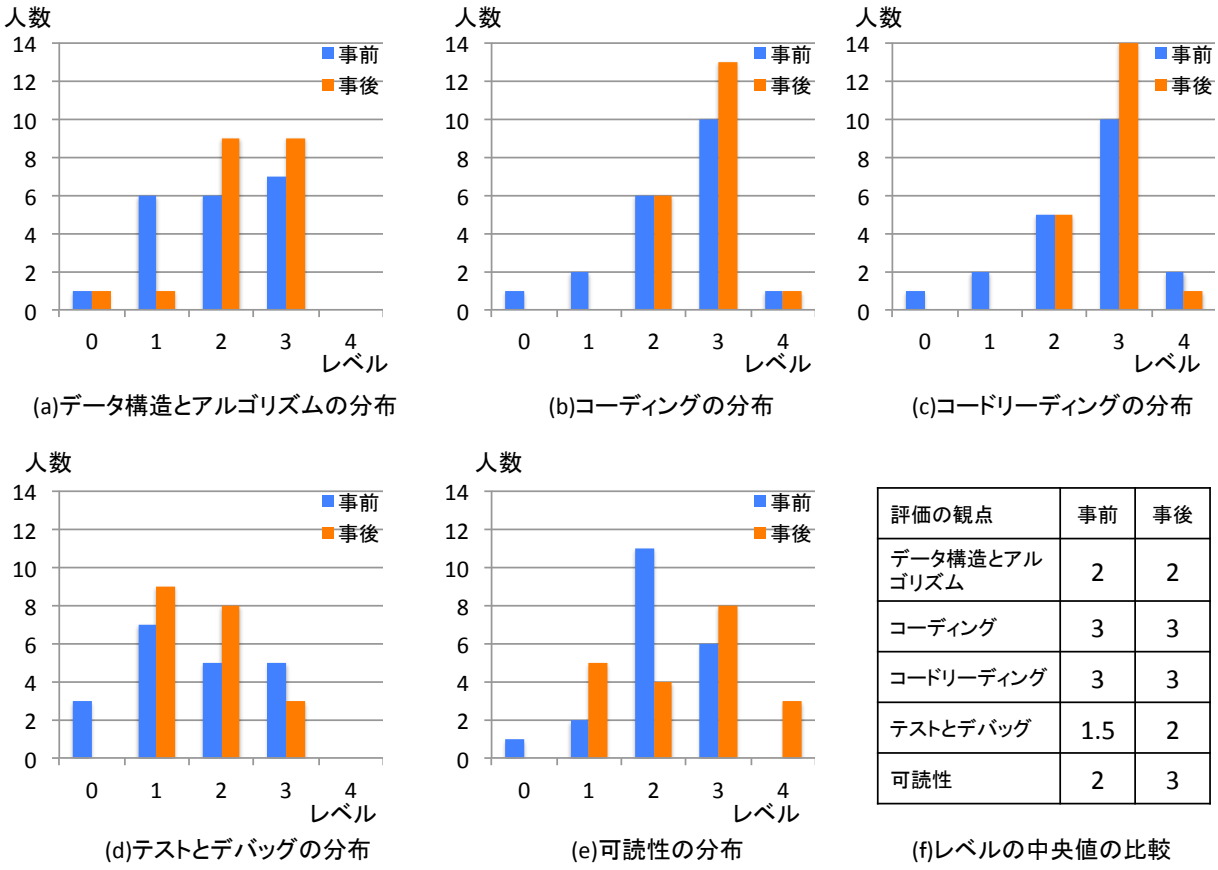
- 自分の能力確認するためにとてもいいと思った．また目標設定もルーブリックをもとに立てることが可能なのでよかった．
- 自分のプログラミングの苦手な部分を改めて確認することなどができて，ルーブリックは自分のプログラミングスキルチェックに有効だと思いました．
- 自分のレベルを再確認することができた．

これらのことから，学生もルーブリックを有用であると感じていることがわかる．

一方で，図 4 の「ルーブリックを使った自己判定が容易」という点については，60%の回答者が「そうは思わない」「どちらかと言うとそうは思わない」と回答しており，自分がルーブリックのどのレベルに該当するかを判断するのは難しいと感じる学生が多いことがわかる．自由記述では以下のような意見があった．

- 評価は完全に自己判断なので，自分のレベルがどの程度と言えるのか迷うところがあった．
- 自信を持って，「３や４のレベルです」とは言いにくいと思う．
- 客観的な評価にならない．

これらのことから，受講者が自己評価を行う際に何らかのフォローが必要であることがわかる．また，「客観的な評価にならない」という指摘も当たっており，実際にルーブリックを用いた評価は主観評価に分類される．ただし，完全に主観的ではなく，誰が評価しても同じような評価になるように間主観性を重視している点に意義がある．このようなルーブリックを利用する意義についても，折に触れて受講者に伝えるのが効果的かもしれない．

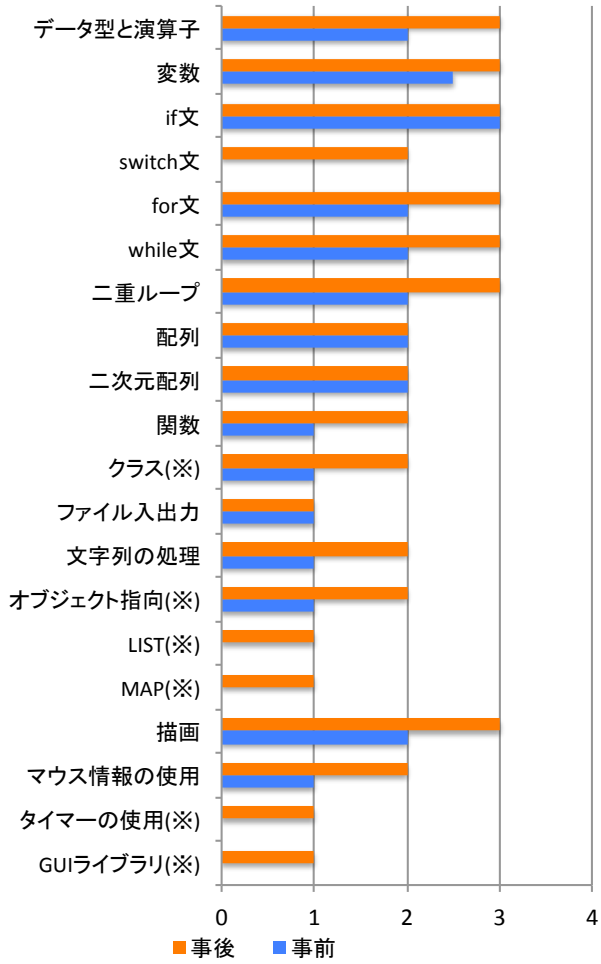


評価の観点	事前	事後
データ構造とアルゴリズム	2	2
コーディング	3	3
コードリーディング	3	3
テストとデバッグ	1.5	2
可読性	2	3

(f)レベルの中央値の比較

図 5　事前と事後のルーブリックを使った自己評価結果の比較

Figure 5　Comparison of Self-Evaluation Results by Students between Pre and Post Programming Study Session.



※はプログラミング勉強会で教材が無かった項目

図 6　各学習項目についてのルーブリックを使った自己評価のレベルの中央値の事前と事後の比較

Figure 6　Comparison of Median Levels of Self-Evaluation by Students for Each Learning Item between Pre and Post Programming Study Session.

6.2 ルーブリックを用いた自己評価シートの分析

プログラミング勉強会において事前と事後の両方とも，自己評価シートを提出した学生は 21 名であった．うち 1 名は事前の自己評価の際に表面(表 1 のルーブリック)への記入をしていなかった．事後の自己評価は事前の自己評価シートを手元に持たない状況で行った．

図 5 に表 1 のルーブリックを使った事前・事後の自己評価結果の比較(20 名分)を示す．図 5(a)から(e)は観点ごとに，各レベルを選択した学生の人数の分布を示したものである．全体的に，事前よりも事後の方が高いレベルを選択した学生が多くなっていることがわかる．図 5(f)は各観点のレベルの中央値を事前と事後で比較したものである．「テストとデバッグ」および「可読性」については中央値の向上が見られ，事前と事後で学生の自己評価結果の向上が著しいことがわかる．「テストとデバッグ」については「テスト駆動開発」と「デバッガの使い方」を教材として配置していたこと，「可読性」については各教材の中で読みやすいコードについて説明があったことによると考えられる．

図 5(a)から(e)において，レベル 4 を選択する学生が少ないのは，ルーブリックのレベル 4 の記述が本学のカリキュラムで扱っている内容よりも高くなってしまっている可能性がある．ルーブリックの表現や 3 年次における教授内容をもう一度見直して検討をする必要があると考えられる．

図 6 に付録 A.1 を使った自己評価において，各学習項目について学生が選択したレベルの中央値を示す．「配列」と「二次元配列」以外は中央値が向上していることがわかる．特に「switch 文」は中央値が 0 から 2 に大幅に向上しているが，プログラミング勉強会に参加した学生の多くが授業では switch 文を学んでいなかったことから，妥当な結果であり，ルーブリックが有効に機能していると考えられる．

一方で，図 6 で※の付いている項目は今回のプログラミング勉強会には教材がなく，事後の自己評価で中央値が向上するほど，ルーブリック上のレベルが高くなるとは考えがたい．これらに関して，的確な自己評価を行えていない可能性がある．

6.3 考察

これまで述べたことから，全体的に見ると，作成したプログラミング教育用ルーブリックは有効に機能しており，学生の多くはわかりやすく，有用であると感じていた．したがって，実際の授業においても利用可能であると考えられる．

一方で，学習項目ごとの自己評価においては的確に判定できていない可能性も示唆された．ただし，これは以下のことから，実際の場面の方が的確な判定を行える状況にあると言える．

- プログラミング勉強会では，学習項目の名称のみを見て自己評価したが，実際の授業においては，特定の学習項目を授業で十分に扱った上で自己評価することになる．
- プログラミング勉強会では，多くの学習項目に対する自己評価を一度に行ったが，実際には授業で扱った学習項目のみについて自己評価をすることになる．

ただし，自分のレベルを判断するのは難しいと感じている学生が多いことから，自己評価において何らかのフォローが必要である．その一つの方策が，授業の課題にルーブリックの対応するレベルを明記することであるが，それでも十分でなければ，学生が自己評価を行う際に，教員や学生補助員(SA)などと相談できるようにすることも考えられる．

今後，ルーブリックの内容，3 年次の教育内容を見直すのに加えて，自己評価におけるフォローの方法を検討したい．

7. おわりに

本稿では，1 年次から 3 年次までのプログラミング教育において共通に用いることができるプログラミング教育用ルーブリックについて述べた．

今後，プログラミングの基礎教育において，反転授業の導入，学生補助員(SA)による少人数担当制，およびルーブリックの活用を特徴としたプログラミング授業を実践して，その効果を評価したい．また，プログラミングルーブリック自体についても引き続き検討を行いたい．

参考文献

[1] 佐々木 茂, 渡辺 博芳. インストラクショナルデザインに基づいた教材開発のための授業・教材設計ツールの開発. 平成 22 年度 ICT 活用による教育改善発表会. 2010, B-14，pp.56-57.

[2] 佐々木 茂, 荒井 正之, 古川 文人, 渡辺 博芳, 武井 惠雄. 学生アシスタント参加による Java プログラミングコース Web 教材の開発． 平成 19 年度情報教育研究集会講演論文集. 2007, C1-4, pp.174-177.

[3] Bailie, F., Marion, B., & Whitfield, D.. How rubrics that measure outcomes can complete the assessment loop. Journal of Computing Sciences in Colleges. 2010, 25(6), pp.15-25.

[4] Mustapha, A., Samsudin, N. A., Arbaty, N., Mohamed, R., & Rahmi, I.. Generic Assessment Rubrics for Computer Programming Courses. Turkish Online Journal of Educational Technology. 2016, pp.53-61.

[5] R.J.マルザーノ，J.S.ケンドール (黒上晴夫，泰山裕 訳). 教育目標をデザインする 授業設計のための新しい分類体系. 北大路書房，2013.

[6] S.F.ヤング，R.J.ウィルソン (土持ゲーリー法一 監訳). 「主体的学び」につなげる評価と学習方法 ~カナダで実践される ICE モデル~. 東信堂，2013.

付録

付録 A.1 学習項目ごとの評価シート

名前

規準	レベル0	レベル1	レベル2	レベル3	レベル4
コーディング	レベル1のレベルに満たない	基本的な文法を理解している. ・1行程度の日本文での表現に対応するコードが書ける. ・キーワード，記号など文法レベルのプログラムの空欄を埋めることができる.	基本的な文法と処理手順の関連を理解している. ・与えられたデータ構造と処理手順を基にプログラムコードを書ける. ・条件や処理内容など，他との関連で考える必要があるプログラムの空欄を埋めることができる.	基本的なプログラミングができる. ・与えられた仕様に対して，データ構造と処理手順を自分で考えて，プログラムを作ることができる.	実際的なプログラムを作ることができる. ・問題状況に対して，プログラムの仕様を決めることができる. ・与えられた仕様に対して，可読性が高く，正しく動作するプログラムを作ることができる.
データ型と演算子					
変数					
if文					
switch文					
for文					
while文					
二重ループ					
配列					
二次元配列					
関数					
クラス					
ファイル入出力					
文字列の処理					
オブジェクト指向					
LIST					
MAP					
描画					
マウス情報の使用					
タイマーの使用					
GUIライブラリ					