研究ノート

Moodle を用いた自学自習ベースのプログラミング教育

河　村　一　樹

東京国際大学論叢　人間科学・複合領域研究　第 6 号　抜刷
2021 年（令和 3 年）3 月 20 日

研究ノート

Moodle を用いた自学自習ベースのプログラミング教育

河　村　一　樹

Self-study Based Programming Education Using Moodle

KAWAMURA, Kazuki

Abstract

The author has been in charge of programming education since 2013 in the Faculty of Commerce. At first, in my class, I proceeded to practice after the lecture. But, with this method, problems were discovered that the copy and paste were rampant and the progress of practical training by students was slow. Therefore, I switched to a lesson system that focused on self-study without giving a lecture. In this paper, I describe programming education of self-study method using Moodle. Among them, I will also discuss the new evaluation method that was tried in 2019.

目　次

はじめに

筆者は，商学部において，2013年度からプログラミング教育を担当してきた。当初は，従来型のプログラミング教育ということで，講義（1コマ）の後に実習（1コマ）という形で実施していた。しかし，このやり方では，学生同士のコピペが横行したり，クラス全体の実習進捗が遅いといった問題が発覚したため，講義をせずに自学自習を主とした授業に切り替えた。この理由には，プログラミングスキルの習得は学生によってまちまちであるため一斉授業は適さないこと，2016年度からは全学レベルでMoodleが導入されたことで自学自習ができる環境が整ったことがあげられる。

そこで，本稿では，Moodleを用いた自学自習方式のプログラミング教育について，どのように実施してきたのかを，科目「プログラミング基礎」を事例にして報告する。その中で，2019年度に試みた新しい評価方法（“正確性”に“迅速性”を加味）についても取り上げる。

1. 商学部におけるプログラミング教育

商学部では，情報ビジネス学科時代から，プログラミング教育を実施してきた。[1)] 講義科目では「プログラミング論」「アルゴリズム論」「Webデザイン入門」，実習科目では「プログラミング実習」「応用プログラミング実習」「Webプログラミング」があげられる。

「プログラミング論」は，基礎理論（形式言語理論，プログラム意味論，計算量理論），プログラムの構成要素，プログラミング言語の種類と特徴，データ構造とアルゴリズム，言語処理系について論じる科目である。[2)]

「アルゴリズム論」は，データ構造（配列，リスト，グラフ，木），伝統的なアルゴリズム（整列，探索，表検索とハッシュ法，最短経路問題，巡回セールスマン問題），再帰（階乗，ハノイの塔）について論じる科目である。

「プログラミング実習」は，プログラミング言語を用いてプログラムを作成して実行することによって，プログラムの動作を検証するといった一連の開発工程を実習する科目である。プログラミング言語については，当初C言語を用いていたが，途中からJavaScriptに変更した。

「応用プログラミング実習」は，「プログラミング実習」で扱わないプログラミング言語として，JavaあるいはVisual BASICを用いて実習を行う科目である。いずれも，Javaではオブジェクト指向プログラミングを，Visual BASICではビジュアルプログラミングを，それぞれ実習することになる。

「Webデザイン入門」は，Webプログラミングの基礎（クライアントサーバシステム，データベースアクセス，フレームワーク）を前提に，Webサイトのデザイン，実装方法，評価基準について論じる科目である。

「Webプログラミング」は，サーバサイドとしてPerlあるいはPHPのプログラミングを，フロントサイドとしてHTML/CSS/JavaScriptのプログラミングを，それぞれ実習する科目である。

その後，情報ビジネス学科が改組されたことにともない，いくつかの科目が統廃合されて今日に至っている。その中で，顕著な傾向として表れてきたこととして，Webアプリケーション（Webサーバとデータベースサーバとの連携），サーバサイドプログラム（CGI），フロントエンド（ある

いは，クライアントサイド）プログラムとしてのスクリプト言語（HTML5，CSS3，JavaScript）に重点が置かれるようになってきたことがあげられる。

本稿では，筆者が担当してきた「プログラミング基礎」（2015年度までは「プログラミング実習」）を取り上げ，2016年度から全学レベルで運用が開始されたMoodleを使った自学自習ベースの授業の在り方について報告する。

2. 科目「プログラミング基礎」について

筆者は2013年度から「プログラミング実習」を担当している。2016年度からは「プログラミング基礎」に名称変更され，開講数は週2コマであるが，実習だけでなく講義も含むので単位数を2単位から4単位に変更した。

当初は，2コマ連続で授業を行っていたが，2016年度からペアコマ制が導入されたことにより，月1コマ木1コマという授業編成になった。これにより，授業時間が分散されて繰り返し行えるので，日々の授業においてスキルを育成するような科目においては効果的といえる。また，これをきっかけにして，授業中だけでなく，授業外における自学自習を促進できる可能性が出てきた。

以上のことから，それまでの講義による一斉授業という形態を改め，自学自習をベースとした授業の在り方について探求してきた。また，自学自習をベースとするためには，何らかの学習支援システムが必要になる。そこで，2016年度から導入されたMoodleを利活用することで，自学自習を支援する教育環境を構築してきた。

2.1　科目の教育目標

大学における情報教育においては，高度な情報技術を駆使する開発サイドの専門教育から，情報技術を利用するユーザサイドの教養教育まで幅広い領域を含んでいる。こういった情報教育を実現するためのカリキュラムについては，情報処理学会が策定した「カリキュラム標準J17」[3] があげられる。これは，米国ACM/IEEE-CSのCC2001-2005 [4] をもとに，日本の情報専門教育に応じた形で再編成したカリキュラムであった「カリキュラム標準J07」を，ほぼ10年毎に改定しているものである。

プログラミング教育については，専門教育として，「カリキュラム標準J17」の「コンピュータ科学領域（J17-CS）」（「アルゴリズムと計算量」と「プログラミング言語」）で取り上げている。それに対して，教養教育としては，「一般情報処理教育（J17-GE）」（「アルゴリズムとプログラミング」）で取り上げている。

本学部におけるプログラミング教育は，どちらかというとJ17-GEのカリキュラムを参照した形となっている。その背景には，システムエンジニアやプログラマといった専門的な技術者を育成することではなく，情報技術を健全に利活用できるユーザを育成することを目指していることがあげられる。情報技術が進展している社会の中で，ディジタルデバイドとならずに，情報技術の恩恵を授受できることによって豊かな生活を送れる社会人として送り出すことが求められているからである。

以上より，「プログラミング基礎」の教育目標は，ただ単にプログラミング言語の構文を覚えるのではなく，プログラミングを通して，抽象化能力およびアルゴリズムによる論理的な思考力を育成すること，コンピュータの動作原理（符号化とコード体系，メモリ上の動作，逐次処理，…）を理解すること，コンピュータをブラックボックス化せずにその可能性と限界を知ること，さら

には，（浮きこぼれの一部の学生に対して）自分がデザインしたプログラムをコンピュータに実装できることとする。

2.2　言語の選択

本科目において，どのプログラミング言語を取り上げるかについては，時代の流れに応じて変わってきたといえる。当初の「プログラミング実習」においては，C言語を採択していた。これは，本来C言語はシステム記述言語であり，OSを記述できるような高度なプログラミングができる一方で，機能を限定すればプログラミングの入門言語としても使うことができるからである。

その後，Web系の技術革新が進み，Webプログラミングが主流になってきた。それに伴い，2013年度からC言語からJavaScriptに変更することとした。この理由としては，次のようなことがあげられる。

1）学習のしやすさの差

C言語では，型宣言の中にポインタ型を含む。これによって，直接的にメモリを操作することも可能となる。また，計算の基本単位である関数をオリジナルで記述するプログラミングとなるが，それ以外にも豊富なライブラリ関数が用意されている。これらを使いこなすことによって，より高度なプログラム開発が可能になるが，それに至るまでの学習の困難さを感じ，初心者の学生からみると敷居が高くなる。

一方，JavaScriptはスクリプト言語に位置づけられることから，台本を書くように簡潔にプログラミングができる言語である。たとえば，型宣言（数値と文字だけ）や命令規約（文の終端記号の有無）もゆるやかであることから構文エラーが減ったり，連想配列が使えたりすることから，C言語よりも学習しやすい言語といえる。

2）学習意欲の差

C言語を学ぶことによって（浮きこぼれの学生の中で）プログラマを目指すことはできるが，そもそも学科の教育目標ではない（プログラマ育成ではない）ため，C言語を学んでいる学生にとっては，何のために履修しているのか疑問に思うことがあるかもしれない。また，経産省の国家試験である基本情報処理技術者の出題にはC言語が含まれるので有利ではあるが，プログラマを指向しない学生にとっては意味のないことになる。

これに対して，JavaScriptを学ぶことによって，自分がやりたいこと（たとえば，自分のオリジナルなホームページやゲームプログラム，スマホアプリを開発して公開する）が具体的にイメージできる。これによって，学生の学びへのモチベーションが向上するといった効果が期待できる。

3）学習環境の差

C言語の処理系では，テキストエディタとコンパイラおよびリンケージエディタを組み込んだプログラム開発環境が必要になり，あらかじめ大学のPCにインストールしなければならない。また，実習では，ソースコードの入力後，翻訳と連係編集を行うことで実行可能プログラムを生成してから実行（デバッグ）を行う。このため，プログラムの実行までに手間がかかる。

一方，JavaScriptは，ブラウザにインタプリタが組み込まれており即時実行ができる。これより，独自のプログラム開発環境を用意する必要もなく，テキストエディタとブラウザさえあれば実習環境が整うことになる。また，ブラウザ（たとえば，Internet Explorer11やGoogle Chrome）によっては，デバッガまでもが組み込まれている。このことは，大学以外（自宅等）のPCでも実習が可能になるということでもあり，自学自習を推進するきっかけにもなり得る。

2.3　科目の概要

半期科目であるが，上述したようにペアコマ（月・木）で実施している。授業で取り上げる内容については，自著[5]をベースにしている。その自著では，**図1**のような章立てとしている。

なお，第7章については，当初はJavaScriptによる動的な振舞い（ウィンドウの動き，文字の動き，画像の動き，フォーム）をプログラミングするという実用編としていた。しかし，その後JavaScriptの仕様そのものが改変（動的な振る舞いに対する一部制限）されたことにより，別のアプローチが必要になった。そこで，CSSとJavaScriptの連携を図るとともに，簡単なゲームプログラミングを扱うように，Moodleにアップしたディジタル教材の変更を行った。

具体的に取り上げる内容と課題については，**表1**のようになる。

2.4　評価方法

講義が中心となる知識獲得型の授業では，毎回の授業で論じた事柄をどれだけ理解できたかを期末の筆記試験で評価する形が適しているといえる。これに対して，実習を含むスキル習得型の授業では，毎回の授業の中で，段階的にスキルを習得していくことになり，学習過程そのもの（学習時間，実習の早さ，課題に対する正答率，課題提出数，…）を評価する方が有効といえる。

本科目では，後者による評価を行っているが，これを手作業で実施するとなるとかなり多くの時間と手間がかかることが問題となる。そこで，2016年度に全学レベルで導入されたMoodleを使うことにした。

Moodleでは，教材（解説文，例題，ヒント，アドバイス）や実習課題をアップロードしておくことで，学生はいつでも学ぶことができる。実習課題については，出来次第Moodleにアップロードできるとともに，その時刻もログデータとして取得できる。これによって，各課題における実習時間（課題をダウンロードしてからアップロードするまで）を目安として算出することも可能になる。

第1章　情報[処理]教育	第5章　JavaScriptの基本編	4．フォーム
1．情報処理教育と情報教育	1．画面への出力	⬇（途中から）
2．情報処理教育におけるプログラミング	2．変数の扱い	
3．情報教育におけるプログラミング	3．演算式の扱い	第7章(新)　CSSを利用したWebサイトへの実用的適用
第2章　プログラム	4．画面からの入力	1．Webサイトの基本的構造とCSS
1．プログラムとは	5．選択文	2．CSSについて
2．プログラミングパラダイム	6．繰返し文	3．CSSの記述
3．プログラム言語	7．配列の扱い	4．CSSを外側に記述する
4．プログラム動作環境	8．関数の扱い	5．JavaScriptを使ってCSSをコントロールする
第3章　アルゴリズム	第6章　JavaScriptの応用編	6．文字列を挿入する
1．アルゴリズムとは	1．整列のアルゴリズム	7．アニメーション
2．アルゴリズムの記述	2．探索のアルゴリズム	8．簡易ゲームの制作
3．アルゴリズムの評価	3．再帰のアルゴリズム	
第4章　JavaScript	第7章(旧)　JavaScriptの実用編	
1．JavaScriptとは	1．ウィンドウの扱い	
2．JavaScriptの動作環境	2．文字の編集	
3．JavaScriptの書き方	3．画面の編集	

図1　教科書の章立て

表1　授業で取り上げる内容

取り上げるテーマ	JavaScriptの構文	課題数	課題番号
自学自習の進め方 *1	―	0	
画面への出力	document.write	1	問5-1-1
変数の扱い	var、代入文(=)	2	問5-2-1～問5-2-2
演算式の扱い	演算子、代入演算子、++、--	2	問5-3-1～問5-3-2
画面からの入力	prompt、eval関数	5	問5-4-1～問5-4-6
選択文	if文、if else文、else if文、switch文	7	問5-5-1～問5-5-7
繰返し文	for文、while文、do while文、break文	11	問5-6-1～問5-6-11
配列の扱い	new Array、連想配列	11	問5-7-1～問5-7-11
関数の扱い	function文、引数/戻り値	6	問5-8-1～問5-8-6
整列のアルゴリズム	選択/バブル/挿入/マージ/シェル/ヒープソート	6	問6-1-1～問6-1-6
探索のアルゴリズム	線形/2分探索	4	問6-2-1～問6-2-4
再帰のアルゴリズム	階乗計算、ハノイの塔	3	問6-3-1～問6-3-3
JavaScriptの応用	HTML，CSS，ゲームプログラミング	6	問7-3-1～問7-8-1
	合計	64	

*1：Moodleの課題ダウン/アップロード、評点の見方、テキストエディタ（TeraPad）、ブラウザ(IE)とデバッグ、評定について

アップロードされた課題については，教員の方で確認した上で，Moodleの［評定］において評点をつけるとともに，学生にフィードバックコメントを戻すことができる。評点については，実習課題の仕様通りの実行結果が得られていれば10点，軽微なエラーが含まれていれば5点，重度なエラーであれば0点，をそれぞれ与えることにする。10点であればその実習課題を合格とし，10点未満であれば指摘された間違えを修正して再提出させるようにする。

半期の間，以上のような形で［評定］を繰り返すとともに，毎回の授業開始時には全学生の進捗状況を公開するとともに，個別指導の際に各人の進捗についても確認する。これによって，学生の学習に対するモチベーションを維持するように心がけた。また，期末において，学生毎に累計された評点がMoodleから取得でき，規則的に評価をつけることができる。

3. Moodleの各モジュール利用について

もともと筆者の研究室では教育ベンダの有償LMS（SATT社のsmart FORCE，ファカルタス社のSeLPS，キャスタリア社のGoocus）を使い，LMSに関する調査研究を行ってきた。2016年度に全学レベルでMoodle[6]が導入されたことによって，無償でLMSが利用できるようになった。これに合わせて，Moodleを取り入れた自学自習方式の授業を，本科目で実践することにした。[7]

自学自習を前提にしたのは，そもそもプログラミングにおける学生のスキルはまちまちであり，習得時間にも習得レベルにも差が生じるからである。このため，一斉授業は適さないことになり，個別指導の方が，教育効果が高いといえる。

個別指導を進めるためには，何らかの学習環境を提供し，そこで自学自習を行うことが有効となる。そこで，すべての教材コンテンツ（テキスト，実習課題，実行結果，ヒント）をリソースモジュールの［ファイル］でアップしておくとともに，学生が自学自習している際に，わからな

いことをいつでも質問できるように活動モジュールの［フォーラム］を用意した。

実習課題を終えた学生は，活動モジュールの［小テスト］（当初は［課題］）を用いてアップロードすることで課題提出を行う。教員は，課題の動作確認をしたのち，ナビゲーションの［評定］で評点を出すとともに，学生にフィードバックを行う。学生は，それを見て，課題が合格したか否かを確認し，不可の場合は再度課題をやり直してアップロードする。

以上の形で，授業中でも授業外（1号館5階の共用パソコン室や自宅）でも，学生が自学自習できるようにするわけである。

なお，教室での授業時間中においては，教員と学生1対1での個別指導を行う。教卓に学生を一人ずつ呼び出し横に座らせて，学生の進捗状況の確認およびわからないことについての質問とアドバイスを，フェースツーフェース（face to face）で行う。その際には，教卓にあるPC画面に学生のプログラムを表示させて動作確認を行う場合もある。これによって，学生はクラス全体の中での自分の進捗状況を把握するとともに，わからないことがわかるようになり，他人のプログラムをコピペする必要もなくなる。[8)]

ただし，最初からMoodleのすべてのモジュールを使って授業を進めたわけではなく，多少の試行錯誤を経験しながら，年度ごとに順次取り込む形で進めてきた。それをまとめたのが，**表2**である。次からは，表の内容について取り上げる。

表２　Moodle の利用

			Moodle					
			ナビゲーション[評定]	活動モジュール[フォーラム]	リソースモジュール[ファイル]		活動モジュール[課題]	活動モジュール[小テスト]
開講時期	受講者数	授業の進め方	課題の評価	質疑応答	教材	ヒントと実行結果	実習課題	実習課題
2016年春	23	講義をせず教科書（自著）を自学自習，授業中は個人指導（教員対学生の1対1）	授業回毎にExcelの表で公開	新規ディスカッションでアップ，全員に公開	自著	全演習問題の実行結果画面とヒントをPDFで提供	ソースコード＋実行結果画面を，docxファイルでアップロード	↓
2017年春	39	教科書の内容はPDFに変換しMoodleで公開	同上	同上	章毎にPDFで提供	同上	ー	ソースコード＋実行結果画面を，docxファイルでアップロード
2017年秋	62	同上	同上	同上	同上	同上	ー	同上
2018年春	19	同上	Moodleの[評定]を利用	同上	同上	同上	ー	同上
2018年秋	50	同上	同上	全員ではなく教員のみに公開	同上	同上	ー	同上
2019年春	44	評価基準に迅速性を追加，これに合わせてアップロード時刻を記録	評点＋提出順としExcelの表で公開	同上	同上	同上	ー	ソースコード＋htmlファイルをアップロード
2019年秋	50	同上	同上	同上	同上	同上	ー	同上

3.1　リソースモジュールの［ファイル］

ここでは，教材コンテンツ（解説文，ヒント，実行結果）をMoodleでどのように扱ったかについて述べる。

2016年度については，自著を教科書に指定し，学生はそれを講読する形で自学自習をさせた。このため，講義は一切行っていない。2017年度以降からは，教科書の一部分をPDFファイル（PCだけでなく，すべてのスマホに対応できるため）に変換して，Moodleのリソースモジュールの［ファイル］を用いてアップロードすることにした。これによって，Moodleだけで自学自習ができるようにした（図2）。

実習課題は各章末に掲載してあるが，講義レスの自学自習だけでは達成できない可能性があるので，ヒントと実行結果をPDF化してMoodleのリソースモジュールの［ファイル］を用いてアップロードした（図3）。

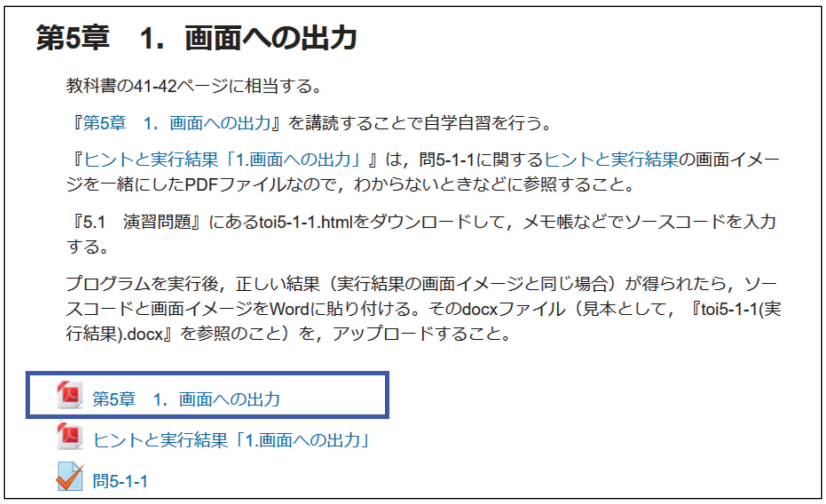


図２　教科書の PDF 化

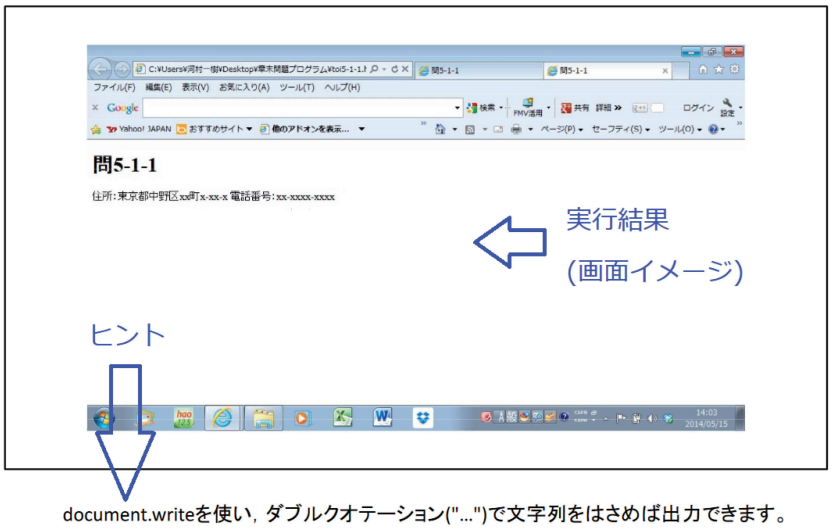


図３　実習課題の「ヒント」と「実行結果」

ヒントとは，実習課題毎に，どのようにプログラミングすればよいかについて解説したものである。具体的には，構文の使い方，該当のアルゴリズム，よく間違える箇所などについてまとめた。実行結果とは，実習課題を実行した結果の画面イメージを表示したものである。これによって，プログラムの仕様とともに，デバッグを含めてどのような実行結果が表示されれば合格とみなすかをわかるようにした。

3.2　活動モジュールの［課題］［小テスト］

ここでは，実習課題をMoodleでどのように扱ったかについて述べる。

2016年度については，Moodleの活動モジュールの［課題］を用いて実習課題を提示した。例えば，問5-1-1の場合，toi5-1-1.htmlのソースコードを用意しておき，学生はそれをダウンロードする（図4）。

次に，そのhtmlファイルを，テキストエディタで開く。その際に，メモ帳ではなく，TeraPadを推奨している。TeraPadにした理由は，行番号が自動的に付番されること，ソースコード中の空白をオプション指定により表示できることがあげられる。とくに，空白については，プログラミングの際に注意が必要となる。ソースコーディングにおいて，字間として空白を使う場合，半角でないと構文エラーを起こす。仮名漢字変換モードを切り替えながらコード入力をしていると，全角の空白が混ざる場合があり，メモ帳ではその個所を見つけ出すのが煩雑（カーソルを動かしながら移動の幅を目視）となる。一方，TeraPadであれば半角と全角がすぐに判別できるので，その分デバッグしやすいといった効果があるからである。

学生は，ダウンロードしたhtmlファイルにおいて，実習課題に応じたJavaScriptのソースコードを指定した箇所（図5の矢印）に埋め込む形でプログラミングする。その後，デバッグを繰り返してプログラムを完成させる。完成したかどうかは，上述した図3の「実行結果」と自分のプログラムを実行した結果の画面イメージが一致したかどうかで判断する。

実習課題が完成したら，そのプログラムのソースコードと実行結果をそれぞれMS-Wordにコピー＆ペーストし，そのdocxファイルをアップロードすることで提出終了となる。

図４　［課題］による実習課題の提示

一方，教員は，次回の授業の前日までに，各学生からMoodleにアップロードされた実習課題をチェックした上で評点をつける。また，エラーについては「提出コメント」に入力することで，学生が後で修正できるようにフィードバックの機会を与える（図6）。

2017年度からは，［課題］ではなく［小テスト］に変更した。［小テスト］に変更した理由は，学習ログデータの取得項目が異なるからである。［課題］の場合は［最終更新日時］しか取得できないが，［小テスト］の場合は［開始日時］［受検完了日時］［所要時間］を取得できる（図7）。こ

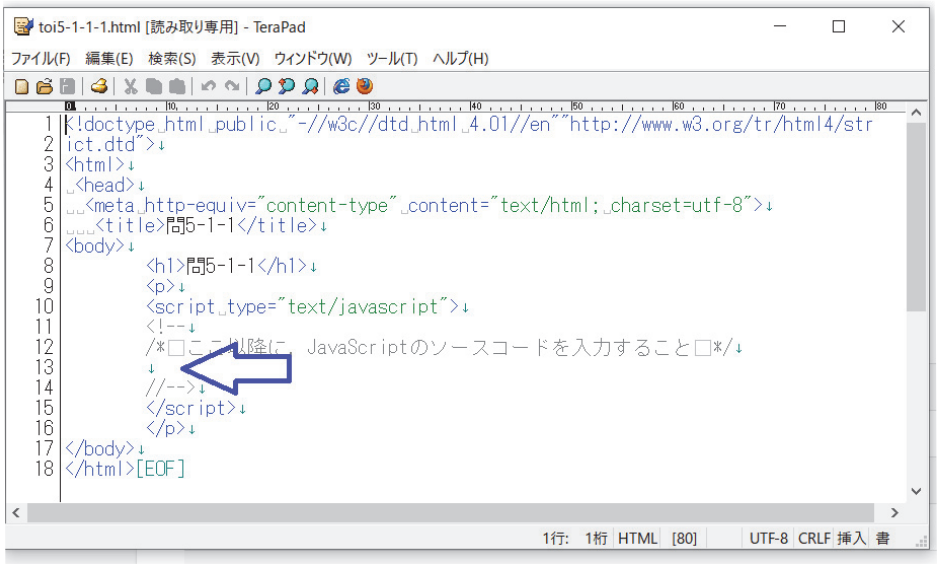


図５　実習課題の進め方（JavaScript の埋め込み）

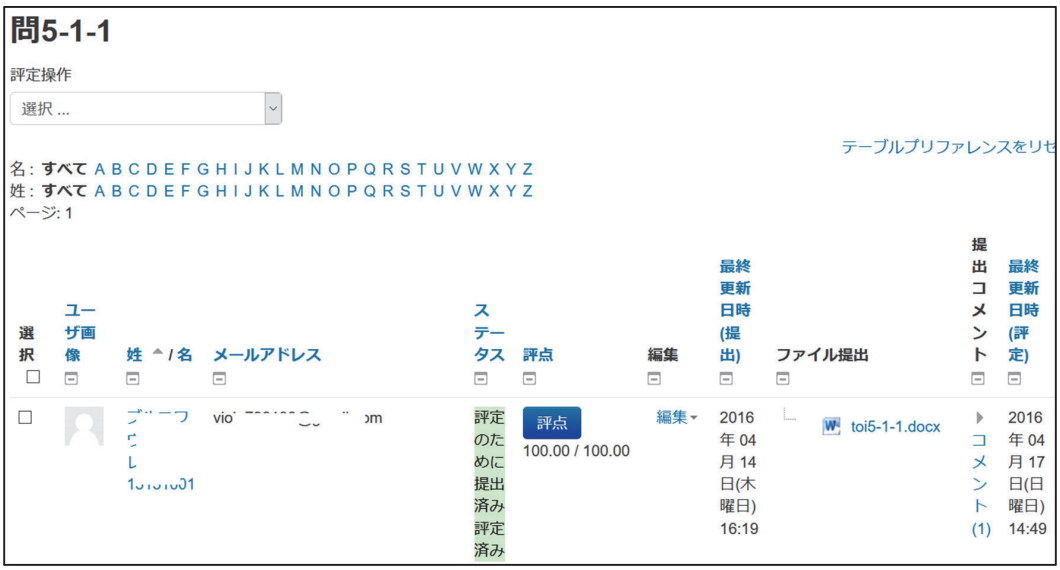


図６　提出された実習課題のチェック

れによって，学生が実習課題をダウンロードしてからアップロードするまでの時間が明らかになり，学生の実習時間の目安を把握することが可能になる。このため，学生には科目ガイダンス時に，課題1問を終えてから次に進むようにと指示した。なお，［所要時間］については，評価対象ではなくクラス全体の進捗状況を把握するための参考にするとした。

［小テスト］における［問題タイプ］は［作文］とし，［問題テキスト］には実習課題の仕様とhtmlコードを入力し，［デフォルト評点］は「1」および［回答オプション］の［解答形式］は「HTMLエディタ」とした（図8）。

2019年度からは，実習課題の提出を変更した。それまでのソースコードと実行結果を一緒にしたdocxファイルではなく，JavaScriptを組み込んだhtmlファイルそのものをアップロードさせることにした。これによって，教員の方でプログラムのデバッグが可能になり，より正確な評価ができるからである。とくに，論理エラーを起こすようなテストデータ（境界値，上/下限値，…）でデバッグすることができ，学生が見落としているようなケースを指摘できるようになる。これ

		姓 / 名	メールアドレス	状態	開始日時	受験完了	所要時間	評点 /10.00	Q. 1 /10.00
☐		受験をレビューする	yu	終了	2017年 04月 12日 10:21	2017年 04月 13日 16:18	1 日 5 時間	5.00	5.00
☐		受験をレビューする	tal ～ ik.jp	終了	2017年 04月 13日 15:14	2017年 04月 13日 16:12	57 分 51 秒	5.00	5.00

図７　［小テスト］の学習ログデータ

問題 1

未解答

最大評点 1.00

問題にフラグを付ける

問題を編集する

```
<!-- 自宅の住所と電話番号を，次のように表示するJavaScriptのプログラムを作成せよ。
   住所：・・・，電話番号：・・・
作成にあたっては，次のソースコードを利用すること。 -->
<!DOCTYPE html>
<html lang="ja">
 <head>
  <meta charset="UTF-8">
  <title>問5-1-1</title>
 </head>
 <body>
 <h1> 問5-1-1 ○○○○(←自分の名前) </h1>
 <hr>
  <p>
  <script>
  /* ここに，JavaScriptのソースコードを入力のこと */

  </script>
  </p>
 </body>
</html>
```

図８　小テストの構成

によって，デバッグに関して踏み込んだ助言ができる。

3.3　活動モジュールの［フォーラム］

ここでは，自学自習に伴う学生からを質問について，Moodle でどのように扱ったかについて述べる。

教室での授業では，講義はぜずに個別指導だけを行っている。教卓に一人ずつ学生を呼び出し，隣に座らせて，わからないことやプログラムを実行してもうまくいかないことなどについて確認する。場合によっては，PC の画面を共有しながら学生のプログラムの動作確認なども行う。これによって，わからないことをそのままにすることがないようにするとともに，独力でプログラミングする力を養う。

本科目では，授業外における自学自習を推奨しているが，その中でわからないことや原因不明なエラーにより実習ができないといった可能性もある。そこで，電子掲示板の機能を持つ Moodle の活動モジュールの［フォーラム］を使うことにした。

［フォーラム］では，［新規ディスカッショントピックを追加する］でスレッドを立て，質問内容をテキストで入力する。そのときに，添付ファイルもつけることができるので，自分のソースコードを添付してもよい。多少のタイムラグは生じるが，教員は質問の回答を返信することができる。これによって，授業外でも自学自習を支援できるようにした。

この形での［フォーラム］であれば，履修者全員が閲覧できることによって，同じようなことでつまずいている学生も参考になる。しかし，授業期間も終わりに近づくと，課題提出をあせるあまり［フォーラム］の中身をまねて提出するという問題が生じた。

そこで，2018 年度の秋学期からは，［フォーラム］を学生一人と教員だけのグループ編成に変更した。そのためには，学籍番号ごとに［フォーラム］を設置することになる。［モジュール共通設定］の［グループモード］を「分離グループ」とし，［利用制限］の［アクセス権限］において「学生［合致する必要がある］>以下の条件に対して］［グループ「学籍番号」］とし［制限を追加する］とする。これによって，学生は自学自習中に，他の学生を気にすることなく，自分一人だけで質問できるようにした（図 9）。

図 9　フォーラムの構成

3.4　ナビゲーションの［評定］

ここでは，学生が提出した実習課題の評価方法について，Moodleでどのように扱ったかについて述べる。

3.4.1　Excelによる評価

2016年度については，［課題］としてアップロードされたdocxファイル（ソースコードと実行結果画面）をもとに，評点（合格：100点，軽微なエラー：50点，重度なエラー：0点）をつけた。それとともに，学生/課題毎に提出日時（ファイルのアップロード日時）をつけた進捗状況表をExcelで作成し，Moodleの［ニュースフォーラム］に授業回毎にアップした。これによって，学生同士でお互いの進捗を確認することができるとともに，個別指導のときに学生毎の進捗を確認し，遅延が見られる学生については助言を与えた（図10）。

3.4.2　［評定］による評価

2017年度については，実習課題を［課題］ではなく［小テスト］としたことによって，［開始日時］だけでなく［受験完了］日時と［所要時間］がログとして取得できるようになった。このため，進捗状況表において，これらも追記することにした。ただし，このときはあくまで所要時間を学習している時間の目安に留め，個別指導の際に参考にするだけとした。

2018年度については，Moodleの［評定］を利用することとした。［小テスト］でつけた点数がそのまま［評定］に反映されるとともに，クラス全体の評定をcsvファイルで出力できるので，それを進捗状況表としてそのまま利用できるからである。

3.4.3　“正確性”＋“迅速性”による評価

2019年度については，新たな試みを実践した。[9] それは，実習課題を［小テスト］に変えたことにより，受検完了日時が取得できるという点に着目したことにある。

いままでの評定では，“正確性”，つまり，プログラムの仕様通りの実行結果が得られていることを評価対象にした。これに対して，より多面的な評価をする[10]ために，“正確性”だけでなく“迅速性”を加味することにした。

“迅速性”とは，プログラムを完成するまでの早さの度合いのことである。つまり，実習課題に応じたアルゴリズムをデザインしコーディングかつデバッグにかかる合計時間が短いほど，プログラミングスキルが高いという評価を与える。これによって，より多くの課題を提出した学生の評価を高くしようとする目論見である。

図 10　クラス全体の進捗状況

また，これについては，授業外の自学自習を促進するという狙いもあった。Moodleにアクセスできる環境さえあれば，授業中だけでなく授業外でも自学自習に取り組めるとともに，わからないことについては常時フォーラムで対応できるようにしてあるからである。

(1) 平常点の算出

“正確性”については，実習課題の仕様通りの結果が得られていれば10点を与える。

“迅速性”については，提出順位を10点満点から0点までに換算することにした。具体的には，ある課題について，提出した順番に「順位」をつける（**図11**）。その「順位」は，{全提出者数-(n-1)}とし，nは提出した順番（早い順に，1，2，…，全提出者数，なお同時刻の場合は同順）とした。それを，ExcelのRANKS関数を用いて算出している。例えば，セル2Dについては，問5-1-1の受検完了日時（課題をアップロードした日時）が，全提出中何番目の提出だったかを降順にして「順位」を求めている。「10点換算」は，「順位」を10点から0点までに換算している。ここで，「C46」は問5-1-1の提出総数をCOUNTA関数で算出している。「評点」は“正確性”から求めた点数であり，10点（合格）か空白（未提出）となる。

以上をもとに，「10点換算」と「評点」を合計して「平常点」を求めている。

(2) 実施結果

2018年度の秋学期（“正確性”のみの評価）と2019年度の春学期（“正確性”＋“迅速性”による評価）におけるクラス全体の進捗度合いや課題提出数がどのように変化したかについて調べてみた。

①授業回毎の課題提出状況

授業回毎に，クラス全体として課題提出数の累計をとりグラフ化した（**図12**）。

提出率としたのは，年度毎に履修者数が異なるので，<i回目の課題提出数>/64×<履修者数>［%］として算出した。なお，2018年度の13回目は未表示になっているが，これはクォーター試験期間中のため休講となったためである。

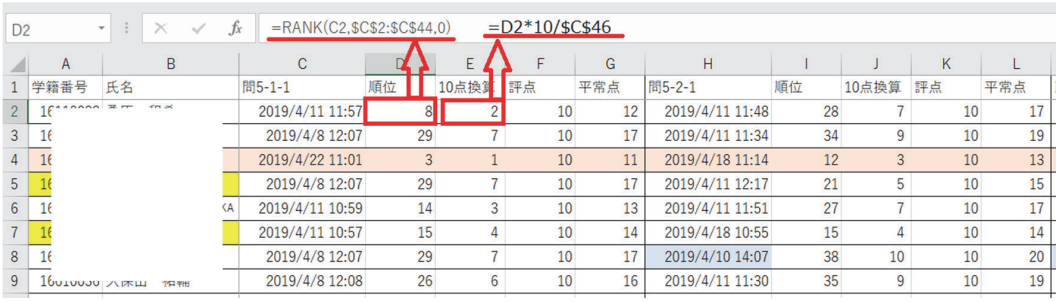
図12より，2019年度の方が毎回の課題提出率が多いことが明らかになった。

②全授業終了後の課題提出状況

すべての授業が終了してから得た課題提出結果をもとに，課題毎の提出率（**図12**）とクラス全体のヒストグラムを作成した（**図13**）。

図13より，2018年度は問5-8-1から課題提出が半分程度となったのに対して，2019年度は問5-8-6までほとんどの学生が提出しただけでなく，問6-3-3まで提出した学生が数名いた。

図14の提出率は，<提出者数>/<全履修者数>［%］として算出した。いずれの年度においても，64問すべてを終えた学生はいないとともに，2018年度は最も多く提出した学生の課題数は49問なのに対して，2019年度は58問となった。また，2018年度は51問以上の提出は全くなかった。



	A	B	C	D	E	F	G	H	I	J	K	L
1	学籍番号	氏名	問5-1-1	順位	10点換算	評点	平常点	問5-2-1	順位	10点換算	評点	平常点
2	16		2019/4/11 11:57	8	2	10	12	2019/4/11 11:48	28	7	10	17
3	16		2019/4/8 12:07	29	7	10	17	2019/4/11 11:34	34	9	10	19
4	16		2019/4/22 11:01	3	1	10	11	2019/4/18 11:14	12	3	10	13
5	16		2019/4/8 12:07	29	7	10	17	2019/4/11 12:17	21	5	10	15
6	16		2019/4/11 10:59	14	3	10	13	2019/4/11 11:51	27	7	10	17
7	16		2019/4/11 10:57	15	4	10	14	2019/4/18 10:55	15	4	10	14
8	16		2019/4/8 12:07	29	7	10	17	2019/4/10 14:07	38	10	10	20
9	16		2019/4/8 12:08	26	6	10	16	2019/4/11 11:30	35	9	10	19

図11　平常点の算出

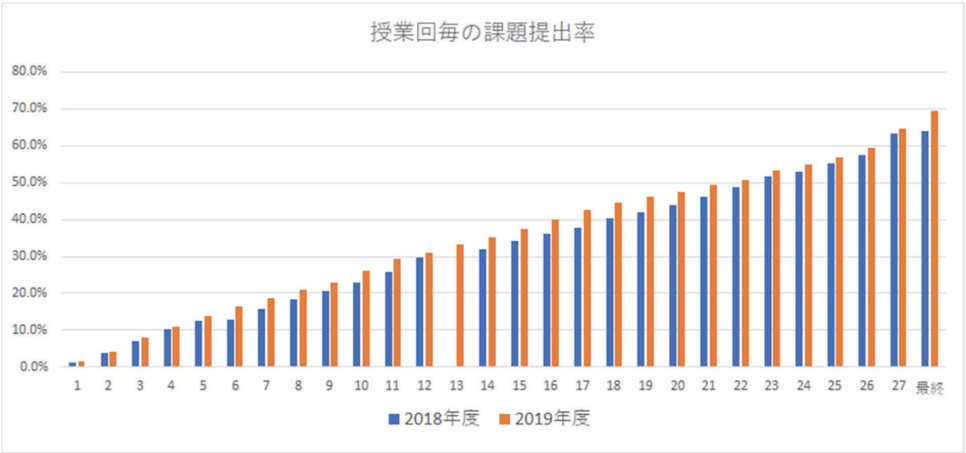


図 12　年度別の課題提出状況の比較

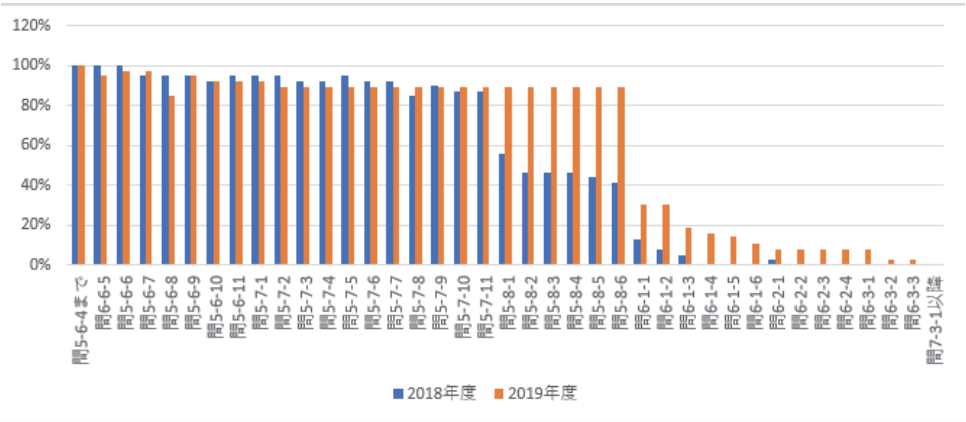


図 13　課題毎の提出率の比較

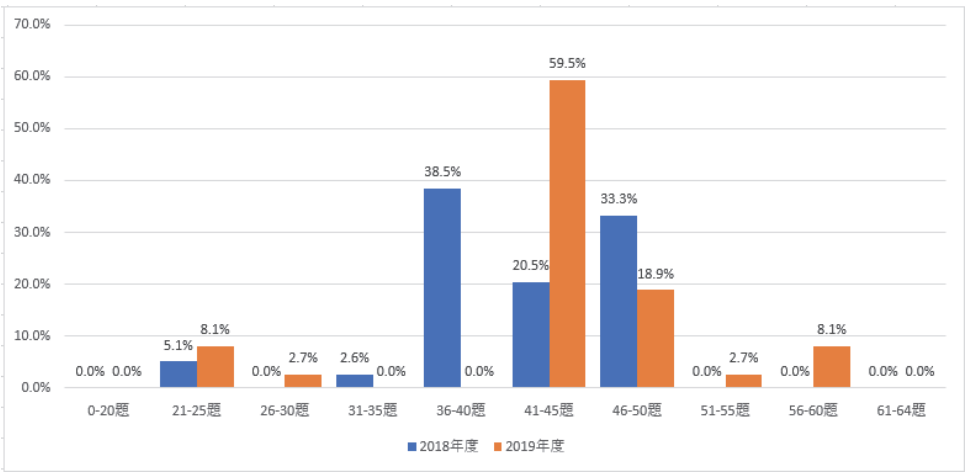


図 14　クラス全体の課題提出状況の比較

以上より，2018年度に比べて2019年度の方がクラス全体として課題達成度および進捗率が向上したことが明らかになった。このことは，評価対象を“正確性”だけとするのではなく“迅速性”も含めることで，より多くの課題をできるだけ早く終えるという学生の意欲ややる気が向上したことを表している。

なお，“迅速性”を評価に含めるということは，科目ガイダンス時に説明するとともに，毎回の授業開始時にクラス全体の進捗状況をMoodleで公開し，かつ，個別指導において学生個々人の進捗を確認しながら遅延が生じている場合にはその原因を探りながら助言を与えた。これらによって，学生の自学自習に対するモチベーションを下げないように指導したことも影響しているかもしれない。

4. オンライン授業に向けて

COVID-19の影響により，本学でも2020年度春学期からZoom[11]によるオンライン授業が開始されている。ただし，コンピュータの実習を含む科目については春学期開講せずとなっている。その理由としては，次のようなことがあげられる。

①実習室において，密集を避けるために座席の間隔をあけるとなると，実習室に入れる学生数が制限されることになり，別立てで授業を開講しなければならなくなる。その結果，教員不足が生じる可能性がある。

②実習室に設置されているPCは共用となっており，電源ボタン，キーボード，マウス，プリンタなどは不特定多数の人が触ることになるので，そのための消毒作業が頻繁に生じ，別途要員を配置しなければならなくなる。

③授業では学生の質問に対して教員がそばに行き学生のPC画面を見ながらのアドバイスとなるが，そこで教員と学生間の密接が生じる。

④実習科目では，履修する学生数によってTA（Teaching Assistant）やSA（Student Assistant）を入れることができるが，Zoomによるオンライン授業では実習における補助作業にうまく対応できない。

⑤受講する学生がPCを所有していなかったり，大学以外の自宅や下宿先等でのネットワーク環境が不十分（WiFiなし，キャリア契約による通信量の制約）である場合，受講に支障をきたす可能性がある。

⑥PCを所有していない，あるいは，ネットワーク環境が不十分な学生に対して，学内の実習室にあるPCやWiFiを利用させるという策もあるが，学生が登校する際の交通機関や学内（食堂やロビーなど）での感染の可能性がある。

⑦PCではなくスマホだけで受講しようとしても，プログラミングの実習ではソースコードの入力作業が生じるため難しいといえる。外付けのキーボードをBluetooth経由で接続する策もあるが，別途購入費がかかる。

一方，5月下旬には，緊急事態宣言が解除されたことから，2次・3次感染を警戒しながら，徐々にではあるが日常の生活に戻りつつある。ただし，本学では，春学期が終了するまでオンライン授業を継続することになっており，対面授業は一切行わないことになっている。その上で，秋学期からは，オンライン授業を中心に，一部対面授業の実施を予定している。

そこで，本科目がオンライン授業に移行できるかについてだが，結論から言うと，可能である。上述したように，本科目に関する教材コンテンツ一式（テキスト，実習課題，ヒント，実行結果）

はMoodleにアップロードしてあり，学生の課題提出も評定もすべてMoodleで行っている。学生からの質問についてもMoodleで対応している。このため，Moodleの教員画面を，Zoomの「画面を共有」により表示しながら講義を進めることができる。

それだけでなく，もともとは個人による自学自習ベースでの授業を想定していることから，Zoomによるオンライン授業にも移行しやすいといえる。学生は，Moodleにアップロードされている教材を自学自習し，実習課題を完成させてMoodleにアップロードする。途中でわからないことやデバッグ時に生じた疑問点などは，Moodleのフォーラムを使うことで理解できるようになる。これらは，すべて個人の作業プロセスとなり，自分のペースで進めることができるので，Zoomとの相性もよいといえる。

ただし，教室で行っていた個別指導をどうするかという問題が残る。これについては，Zoomの［ブレークアウトセッション］[12] を使うことで解決できる。Zoomのミーティングルームを開設後，履修者数分のセッションを［自動］で設定する。学生が指定されたセッションに参加することで，学生一人だけの［ブレークアウトセッション］が作られる。ホストの教員は，出席番号順に学生の［ブレークアウトセッション］に参加することで，教員と学生の1対1でのコミュニケーションができるようになる。また，学生とのやりとりの中で，学生の作成したソースコードを参照したいときには，学生に［合同ホスト］の権限を与え，学生のテキストエディタ画面を共有することによって可能になる。

以上のように，ZoomとMoodleを併用することで，本科目をオンライン授業として実施することができそうなので，秋学期から試みる予定である。

おわりに

以上，Moodleを用いた自学自習ベースのプログラミング教育を実践してきた。本来，プログラミングスキルの習得は，個々人によって異なる。そのため，一斉授業では適さないといえることから，別のアプローチとして自学自習の授業スタイルを提案した。

自学自習を実践するためには，LMSが有効といえる。LMSでは，ディジタルコンテンツ（教材，副教材，課題，テスト）を管理することができるとともに，学習データ（学習回数，学習期間，学習時間，評定）を自動的に取得できる。これによって，学生個々人がどのように学習を進めているのか，どの程度の学習レベルに達したかなどを把握することでき，多少のタイムラグは生じるが学生からの質問に答えることもできる。つまり，LMSは，自学自習を推進するために有用なツールとなり得るわけである。

本学では，2016年度から全学レベルでMoodleを導入し，学内での利用を始めた。これに合わせて，筆者もいくつかの授業の中で徐々にMoodleの機能を使い込んできた。[13–15] そして，本稿では，「プログラミング基礎」という科目において，Moodleを用いた自学自習ベースの授業について報告してきた。その中で，2019年度春学期に実施した評価方法において，新たな知見を得ることができた。それは，プログラミングスキルを評価する基準に，“正確性”だけでなく“迅速性”を加味することによって，クラス全体の実習課題の達成度合いが向上したということである。

今後の課題としては，2020年度秋学期から実施予定のZoomとMoodleを併用したオンライン授業がどのような学習効果を導き出せるのかどうかについて検証する予定である。

参考文献

1）河村一樹：情報ビジネス学科におけるプログラミング教育，東京国際大学論叢商学部編，第84号特集「情報ビジネス学科」，pp. 23–39，2011年.
2）河村一樹，斐品正輝：文科系のためのプログラミング論，日刊工業新聞社，2000年.
3）情報処理学会編：カリキュラム標準J17，2017年.
https://www.ipsj.or.jp/annai/committee/education/j07/curriculum_j17.html
4）The Joint Task Force for Computing Curricula 2005: Computing Curricula 2005 The Overview Report covering undergraduate degree programs in Computer Engineering Computer Science Information Systems Information Technology Software Engineering, 2005.
5）河村一樹：JavaScriptによる情報教育入門，大学教育出版，2011年.
6）Moodle教育管理システムについて――日本での実践――.
https://moodle.org/course/view.php?id=14
7）河村一樹：Moodleを用いたプログラミング教育の事例，TIU学内報 FD Newsletter“SEED”，2018年.
8）河村一樹：開講コマの違いによる学習進捗の相違について――自学自習ベースのプログラミング教育の場合――，情報処理学会コンピュータと教育研究報告，2017-CE-139(8)，2017年.
9）河村一樹：Moodleを用いたプログラミング教育における評価方法――「正確性」に「迅速性」を加味した結果，e-Learning教育研究，第14巻，pp. 34–42，2020年.
10）朽木　拓，山田敬三，佐々木淳：プログラミングスキルレベル評価手法の研究，情報処理学会創立50周年記念全国大会論文集，1–521–1–522，2010年.
11）Zoomアカデミージャパン：Zoomなら実現できます！あなたのセミナーやミーティングをオンラインで開催してみませんか？
https://zoomy.info/
12）Zoomヘルプセンター：ブレークアウトルーム入門.
https://support.zoom.us/hc/ja/articles/206476093
13）河村一樹：Moodleを用いたルーブリック評価の試み――初年次演習での実践――，東京国際大学論叢人間科学・複合領域第3号，2018年.
14）河村一樹：Moodleをプラットホームにした学生の合同開発事例，日本教育工学会第34回全国大会，2018年.
15）河村一樹：Moodleをプラットホームにした学生による教材開発，e-Learning教育学会第17回研究大会，2019年.