拓殖大学北海道短期大学研究紀要　創立50周年記念号
平成30年3月 〈資料〉 pp.124-132

情報リテラシのためのプログラミング教育の導入

庄内慶一

Adoption of Programming Education for Information Literacy

Keiichi Shounai

概要：情報リテラシ教育は近年、問題解決能力不足が指摘されている。こうした中、プログラミングは問題解決に効果ある技術として教育されるようになってきた。本研究は、文系の情報科目に導入するプログラミング教育の授業環境を構築する。学習向けのオンラインプログラミング環境と、自習に要するVODのビデオ教材を用いる。これらをeラーニングに実装し、受講生の情報リテラシを向上する。

キーワード：情報リテラシ、問題解決能力、プログラミング教育、オンラインプログラミング、eラーニング

1. はじめに

プログラミング教育は従来、情報系学部の専門的な授業範囲にあった。しかし、近年のプログラミング教育は、論理的思考力や課題解決力を養うものとして、初等・中等教育にも導入される。プログラミング教育の低コストかつ効果的な実施手法等の実証事業 （総務省，2015）や、継続的な活動を目的に受講生の先輩をメンターとして育成し、初等教育に導入した実績 （鈴木ほか，2018）も報告されている。

スマートフォンはコンピュータ・プラットフォームとして一般に定着してきており、スマートフォンをターゲットとしたアプリケーション開発が増えている （情報処理学会，2018)。プログラミング教育向けには、Webベースでは比較的小規模なプログラムの演習をオープン・プラットフォームとしたCloudCoder （Papanceaet al.，2013）や、情報系学部には初学者向けのC言語プログラミングで間違いの修正およびグラフィックス制作支援の機能を有するBit Arrow （長ほか，2017）が開発されている。他方、開発環境が身近になったことで、プログラミングは理系だけの学習分野ではなくなった。実践的なプログラミングの教育は文系学部にも導入され、学習者自身の体験から、プログラミング技法の基本概念を理解するオブジェクト指向プログラミングの授業 （伊藤ほか，2016）がある。さらに、視覚的にプログラム可能なビジュアルプログラミング言語は、処理系の分かりやすさから初等教育など様々な教育機会に用いられている。その一つの Scratch は文系学生にも、コンピュータやインタラクティブメディアの理解に有効であることが確認されている （森秀樹，2010)。このように高等教育では、問題解決に必要な情報を科学的に捉える能力が求められており、情報リテラシとしてのプログラミング教育が不可欠である。

情報リテラシの内容・範囲は、問題解決に求められる情報リテラシ教育が不十分であると指摘され、

学問の分野別に構成し教育することが考察されている （私立大学情報教育協会，2012)。情報リテラシ科目にプログラミングを一つの学習範囲とするならば、他の技術要素や基礎理論などに充てる学習量も考慮し、一定期間に効果的なプログラミング教育を実施する必要がある。本研究の目的は、情報リテラシ科目に導入するプログラミング教育システムの開発である。本稿では、拓殖大学北海道短期大学の社会科学系コース学生を教育の対象に検討する。

以下、本研究のプログラミング教育システムを検討し、情報リテラシ科目に導入する。次の第2章では学習向けのアプリケーション開発環境を述べる。第3章では授業計画を述べ、その授業内容を第4章に述べる。第5章では実際に導入する論点と導入後の課題を考察する。

2. アプリケーション開発環境の検討

拓殖大学北海道短期大学社会科学系コースの学生は在学する2年間に、職業生活や大学生活に通用する実践的な知識と技能の習得が求められる。関連する資格取得に必要な知識は、情報リテラシ科目でも習得を図っている。本研究が導入するプログラミング教育の内容は、職業人が備えておくべき情報技術に関する基礎知識を証する公的資格「IT パスポート」の出題範囲に当てはまる。出題範囲で基礎理論に分類されているアルゴリズムとプログラミングの範囲を知識項目として取り扱うこととする。情報リテラシ科目の授業計画でプログラミングに充てる時間は制約されることから、本稿では半期の演習科目で全15回の授業の一部に導入する。

同コース2年次では従来、授業計画の一部で Visual Basic によるデスクトップ・アプリケーションを作成している。前提知識を基礎的なコンピュータの知識と利用技術として実施している。過去に受講生はアプリケーションを完成したが、テキストエディタの操作やファイル処理のほか、警告メッセージに対し、インストラクションなしに対処することが難しかった。これを踏まえ、2017年度の同コース2年生を対象に、プログラミングに関する理解の傾向を把握するために、プログラミングの受講前に自己評価に回答してもらった。回答者は1年次に基礎的なコンピュータの知識と利用技術を習得する「情報技術の基礎Ⅰ・Ⅱ」を受講している。質問はITパスポートのアルゴリズムとプログラミングの知識項目に準じた24項目とした。質問文に対する回答は、肯定回答の「はい」、否定回答の「いいえ」のどちらかを選択する。集計は肯定回答をカウントした。実施した期間は2017年11月6日から同月17日までeラーニング・システムのMoodleを利用し、51名が回答した。

自己評価の質問項目と肯定回答数を表1に示す。試験範囲はアルゴリズムとプログラミングそれぞれの系統により4つのカテゴリに分類される。自己評価結果は肯定回答を回答者全員でまとめ、カテゴリ毎で100ポイントに換算した。結果、すべてのカテゴリで50ポイントを下回った。データ構造の42ポイントを最高に、アルゴリズム27ポイント、プログラミング・プログラム言語23ポイント、そしてマークアップ言語22ポイントは最も否定回答を多くした。

自己評価結果から、アルゴリズムとプログラミングの出題範囲全般に、特に否定回答を多くしたプログラム言語に関するカテゴリに重点を置き、プログラミング教育内容を検討する。

表 1　自己評価の質問項目と肯定回答数

Table 1　Self-assessment items and affirmative answers.

n=51

中分類(Scope)	質問項目(List)	受講前(Before PE)
データ構造	プログラム作成において、データ構造を選択する必要性を理解している。	32
(Data structure)	データの構造に関し、キューとは何かを理解している。	14
	データの構造に関し、スタックとは何かを理解している。	18
	データの構造に関し、木構造とは何かを理解している。	13
	データの構造に関し、リストとは何かを理解している。	28
	データの構造に関し、配列とは何かを理解している。	24
アルゴリズム	プログラム作成に関し、アルゴリズムとは何かを理解している。	16
(Algorithm)	アルゴリズムに関し、順次構造について理解している。	12
	アルゴリズムに関し、選択構造について理解している。	15
	アルゴリズムに関し、繰返し構造について理解している。	16
	アルゴリズムに関し、バブルソートについて理解している。	10
	アルゴリズムに関し、二分探索法について理解している。	11
	プログラム作成に関し、流れ図の記号と処理手順の表現方法について理解している。	13
	構造化プログラミングとは何かを理解している。	15
プログラミング・	機械語とプログラミング言語の特徴について理解している。	17
プログラム言語	C言語の特徴について理解している。	11
(Programming language)	Javaの特徴について理解している。	16
	スクリプト言語の特徴について理解している。	9
	COBOLの特徴について理解している。	6
	機械語へ翻訳するインタプリタの特徴について理解している。	11
	機械語へ翻訳するコンパイラの特徴について理解している。	6
	プログラム作成に関し、デバッグの機能について理解している。	16
その他の言語	マークアップ言語の特徴について理解している。	10
(Markup language)	HTMLを記述する基本的なルールについて理解している。	12

プログラミングにより記述するプログラムは、所望の動作を実現する処理手順を成す。この処理でプログラミング言語は、文法に従ったプログラム記述やエラーメッセージなど様々なメッセージをプログラマーに示す。このメッセージそのままでは初学者には理解が難しい。理解を容易にするには、これら処理系のメッセージをモデル化することで簡易に提示する。また、スマートフォンは身近なコンピュータ・プラットフォームであるため、スマートフォン・アプリケーションの作成は、プログラミング初学者の学習の動機づけにも期待できる。

表2はアプリケーション開発環境を特徴で分類した例である。学習向けのScratchとMIT App Inventorは、処理系を視覚的にプログラムできるもので、ビジュアルプログラミング言語と呼ばれる開発環境である。Web アプリケーション向けの JavaScript はスマートフォンなどのブラウザでアプリケーションを実行可能にする。これらの開発環境は、IT パスポートに出題されるアルゴリズムとプログラミングの試験範囲に該当することから、本研究のプログラミング教育用の開発環境に選定する。

表 2　アプリケーション開発環境の例

Table 2　Example of application development environment.

特徴(Feature)	開発環境(Development environment)
学習向けに直感的なプログラミングと実行(Learning application)	Scratch, MIT App Inventor
Webアプリとして開発するプログラムの連携(Web application)	JavaScript
実践的なアプリケーションの提供(Practical application)	Swift, Android Studio, Unity

上記に選定したオンラインプログラミング環境は、導入する情報リテラシ科目において、教室の対面授業形態と、それを補強する e ラーニングで利用可能にする。半期の演習科目でプログラミングの導入教育を完了するために、受講生は毎回の授業時間外に自習をする。教室での対面授業を効果的にする遠隔教育の先行研究では、授業に関連する内容を VOD のビデオ教材としたものがある（INAGAKIet al., 2016)。受講生が家庭でビデオを視聴し、対面授業を補強した成果が得られている。本研究の導入教育では受講生の前提知識を踏まえた VOD 教材を用いる。オンラインプログラミング環境と VOD 教材とを e ラーニングに実装する。

以上を本研究のプログラミング教育の枠組みとし、具体的な授業計画を次章に述べる。

3. プログラミングを導入する授業計画

本章は、情報リテラシ科目にプログラミング教育を導入する授業計画を策定する。対象となるアルゴリズムとプログラミングの授業設計仕様を表 3 に示す。半期の全 15 回、1 回 80 分とする。(1)プロ

表 3　アルゴリズムとプログラミングの授業設計仕様

Table 3　Instructional design specifications of algorithms and programming.

項目(Item)	記述(Details)
名称(Title)	アルゴリズムとプログラミング
授業の目的 (Purpose)	本授業計画は、情報リテラシ科目のためのプログラミング教育を導入する。本授業の目的は、プログラミングの基礎を学ぶことにより情報の科学的な理解を図り、情報活用能力を向上することである。
授業の概要 (Outline)	本授業は次のように進める。 1. 科目の半期全15回のうち4回、1回80分 2. 対面授業を補強するためにeラーニングを用いる 3. 基本的な知識と操作についてはeラーニングで自習する 4. 教室では理解の確認と次の学習をする
授業展開 (Program)	1回1章とし、次の(1)から(4)の全4章を構成する。(1)プログラムの流れをつかむ、(2)プログラムの論理構造の発見、(3)構造化プログラミングの理解、(4)アルゴリズムの発見、これらを順次学習する枠組みとする。
授業内容 (Contents)	文系初学者の前提知識を踏まえる。
授業時間 (Course hours)	対面授業4時間、eラーニング8時間、計12時間
前提知識 (Prerequisite)	基礎的なコンピュータの知識と利用技術
使用メディア (Educational Materials)	オンラインプログラミング環境 動的教材：VOD教材（SCORM2004対応） 静的教材：各開発環境教材（オフィシャルサイトへのリンク） クイズ：択一式（SCORM2004対応）
開発計画 (Development program)	授業前の3ヶ月間、VOD教材、静的教材、クイズの作成。実施環境のテスト。
説明方法、説明図、事例等 (Notes)	具体的操作方法は講師が実演で伝える。学習内容の要点の理解はクイズで確認する。オンラインプログラミング言語を教室内外で利用可能とし、受講生同士教えあい理解を促す。eラーニングを用いて予習・復習を可能とするほか、欠席などで不足した学習内容を補う。

グラムの流れをつかむ、(2)プログラムの論理構造の発見、(3)構造化プログラミングの理解、(4)アルゴリズムの発見、これらを各章に順次学習する枠組みとする。

プログラミングの導入教育に制約された時間で、受講生は授業内容を習得しなければならない。そのために、受講生同士が教え合い相互評価のできる場を、e ラーニングに用意する。e ラーニングはビデオ等の教材を提示できるほか、学習で扱う情報を受講生同士や受講生と教員とで共有できる。学習状況を蓄積し、受講生自身も教員も確認できる。この e ラーニング・システムに Moodle を用いてプログラミング教育システムを構築する。こうして教室外の学習空間を整備することで、情報リテラシ科目全体の学習時間を単純に増やさないように配慮した。

図 1 に e ラーニング各章の学習順序を示す。受講生は教室でも教室外でも、本教育の学習に e ラーニングを用いる。順次学習する各章は、学習開始時に選択する。選択した章のオンラインプログラミング環境で学習する。また、教室との遠隔で、オンラインプログラミング環境のインストラクションのみでは、それを受講生単独で理解するには難しい場合がある。そのため、授業内容の理解不足の受講生は、VOD のビデオ教材を閲覧して学習する。ビデオ教材は自習に要する内容として、操作方法の説明、講義、その他に教室で扱う情報を代替する。クイズは、学習内容の要点の理解を確認する。フォーラムは、受講生同士や教員との質問や意見を交わす。評定は、受講生各自の学習状況を確認する。これらの機能は受講生が学習状況に応じて選択する。

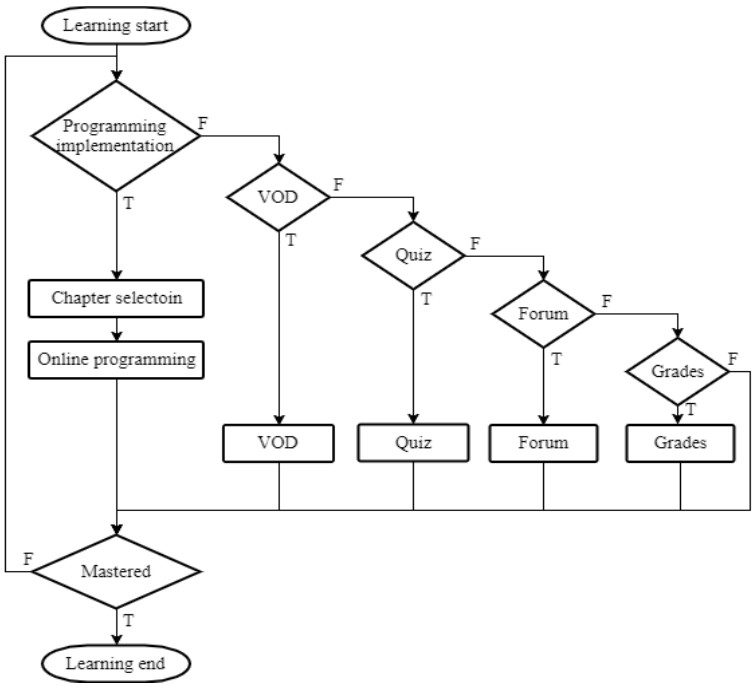

図 1　e ラーニング各章の学習順序

Figure 1　Learning order of e-learning chapters.

成績評価の方法は、各章の学習内容から課題を用意し、受講生の提出した課題の得点により行う。基礎知識はクイズの回答で評価する。

4. 授業内容

導入するプログラミング教育に扱う授業内容は、次の(1)から(4)を各章として順次学習する手順をとる。各章の授業内容を以下に述べる。

(1)プログラムの流れをつかむ

本章では、学習向けのビジュアルプログラミング言語の Scratch により、プログラムが論理的な処理手順を構成していることを理解する。類似したプログラミング環境には MIT App Inventor がある。作成したアプリケーションがスマートフォンで動作する点が Scratch と異なる。作成するプログラムの実行環境は、スマートフォンまたはエミュレータを用いる。ただ、教室ではネットワーク環境により失敗することがある。小規模な LAN 環境での実習に向いている。Scratch と MIT App Inventor は、視覚的な分かりやすさを前提に処理系の制約はあるが、プログラム規模の比較的小さいアプリケーションを簡易に作成できる。

先ず、Scratch の Top ページに紹介されるアプリ（例えば「neon tunnel」）を任意に開き、ブロックで視覚的に組み立てられた手順に従って動作していることを理解する。Scratch の基本操作の学習は、ツールの操作方法を知るだけでなく、プログラミングの注意点も理解する。最も簡単なアプリケーションの場合、オブジェクトとメソッド、オブジェクト名、スクリプト、順次処理、座標、演算、そして初期化が挙げられる。この例題には、例えば、障害物を避けて目標に到達するゲームが当てはまる。また、Scratch は構造化プログラミング、アルゴリズム、比較演算、論理演算、関数などの処理にも対応している。これらも例題に組み込み学習する。

(2)プログラムの論理構造の発見

本章では、(1)の Scratch で扱ったプログラムの論理を、フローチャートにより表現する。個人が作成したプログラムの論理は、組み立てられた記述やブロックそのままでは、他人が見ても論理を理解しにくいことがある。フローチャートを用いることにより、処理の流れを一般化する。使用する流れ図記号は、端子、準備、入出力、選択・判断、ループ端、処理の 6 種である。(1)で作成した、例として障害物を避けて目標に到達するゲームの流れ図を表現する。フローチャート作成ツールは、オンラインサービスの draw.io を用いる。登録不要の無料、日本語で使うことができる。フローチャート用のパーツが予め用意されているので、簡単に使い始めることができる。完成した図をファイルとして保存できる。もし受講生が利用できない環境にある場合、draw.io よりも操作は複雑になるが、Excel アドインの Flowchart_Excel_AddIn を用いることもできる。

この手法を理解した後、実際に新たなシナリオを作成しフローチャートに表現する。ここでは、(1)

の障害物を避けて目標に到達するゲームのシナリオに、初期化とゲーム停止を追加する。初期化はアプリケーション実行時の状態を、ゲーム停止はシナリオ終了の状態を定めるもので、いずれも重要な命令である。受講生はフローチャートを完成した後、Scratch によりプログラムを作成する。

この後は、テキストエディタを用いたプログラミング言語の学習に進む。プログラムの論理構造を把握するには、プログラムの記述に対応した動作の確認も役に立つ。そのために、受講生は予め用意された実行ファイルを実行し、プログラムの動作を把握する。次にプログラムのソースファイルを閲覧し、動作と記述の対応を確かめる。このときフローチャートも用いることで、理解を深める。なお、実行ファイルを作成する場合は、コンパイルの機能も理解する。ここでは、次章の学習への橋渡しのため、HTML と JavaScript を用いる。

(3)構造化プログラミングの理解

(2)で記述に対応した動作を確かめた後、本章では構造化プログラミングを実施する。パソコンやスマートフォンで利用できるプログラムを作成するには、ブロックを組み立てるだけでなく、より汎用的なプログラミングが必要となる。一般的には、プログラミングのコードを記述することで実現する。この記述は、どのようなプログラムでも記述できる構造化プログラミングで成り立つ。

プログラミング言語は主に機械語と高水準言語がある。用途に応じてプログラム作成者が選択する。機械語は 0 と 1 で記述され、コンピュータが直接理解する。高水準言語（Java や C 言語など）はコンピュータが直接理解できないが、人間の日常的な言語で記述できる。機械語に変換して、はじめてプログラムを実行できる。

構造化プログラムは、マークアップ言語の HTML とスクリプト言語の JavaScript を対象に、いずれもオンラインプログラミング環境の Bit Arrow で利用可能である。先ず、受講生が自習できるために、アプリケーションのシナリオを用意する。受講生の理解度に応じて、教員または受講生のどちらが用意しても良い。このシナリオを実現するプログラムを作成し、動作を確認する。Bit Arrow では、HTML と JavaScript とで例題を通して学習する。作成したプロジェクトに任意の名前を付けて、Bit Arrow 内に保存できる。

(4)アルゴリズムの発見

処理の手順はアルゴリズムと呼ばれている。よいプログラムは効率よいアルゴリズムによる。探索と並び替えは身近な問題解決手段であり、アルゴリズムでも大切な方法である。

先ず、数を探す問題を扱う。その数は何番目か。この問題は線形探索である。整列済みのデータ（配列）の片側から順番に、探したい数を調べ、一致したところで調べるのを止め、何番目かを記録する。次に、最小値はどれか。この問題は配列で、片側のデータを仮に最小値とし、隣り合う要素の方が小さい値の場合に、そのデータを最小値とする。これを配列の反対側まで順次繰り返す。これら各サーチ手順を Scratch に実装して動作確認する。数が見つからなかった場合の無限ループと、ユーザによ

る意図しないデータ入力にも対応する。

さらに進んだ学習は、並び替えの問題を扱う。基本的な並び替え手順は、配列の中で不規則に並んでいるデータを大小により整列する。バブルソートと呼ばれている。プログラムのポイントは、隣り合うデータの交換である。データの退避場所を用意して可能になる。次に、選択ソートは、１巡ごとに最小値の添え字を書き換えることで、データ交換の回数をバブルソートより少なくする。挿入ソートは、整列済み要素の部分に対して、新たな要素を大小により適切な位置に挿入する。バブルソートや選択ソートよりも比較の回数を少なくする。これらの各ソート手順を Scratch に実装して、動作確認する。

5. 実施方法および考察

本研究の導入教育の枠組みは、教室での対面授業と、教室外の e ラーニングとで授業を実施する。それぞれの学習段階は、教室と教室外とで明確でなければならない。授業内容の各章において、基本的な知識と操作を教室外とし、その理解の確認と次の学習を教室で行う。教室外で自習が成立するには、以上に述べたプログラミング教育システムを e ラーニングに実装し、効果的に活用しなければならない。

本研究のプログラミング教育システムは、その教育効果を試行的に確認する。そのために、拓殖大学北海道短期大学社会科学系コースの 2 年次で、情報関連ゼミナールの受講生を対象に試行する。試行による学習ログとコミュニケーションログは、e ラーニングの学習管理システムに蓄積される。本導入教育の前後に自己評価も行い、受講生のプログラミングに関する理解度を把握する。これらのデータを分析することで、オンラインプログラミング環境と VOD 教材による教育効果を確認する。開発したプログラミング教育システムによる導入教育の実施方法も検討する必要がある。また、メンターの育成は受講生から選定する方法を考えるが、本研究では導入教育を 2 年次対象としていることから、実現することが難しい。メンターの代替として、本稿第 3 章の授業計画にした、受講生同士が教え合う場を重視する。

なお、アプリケーション開発環境はその技術の移り変わりが早く、数年経過すると主流のツールそのものが替わっている場合もある。本稿第 4 章の授業内容に影響するので、開発環境の陳腐化や再現不可能なものを教材として扱わないように、実際の導入時には注意しなければならない。一方、アルゴリズムとプログラミングに関する理論的な授業内容は、普遍の基礎理論である。したがって、本研究が導入するプログラミング教育において、問題解決で必要となる情報を科学的に理解する授業展開は、開発環境によらず、アルゴリズムとプログラミングの基礎理論により規定すると考える。

6. まとめ

情報リテラシとしてのプログラミング教育に、IT パスポートのアルゴリズムとプログラミングを当てはめた。科目の授業展開の一部に、効果的なプログラミング教育を実施するために、基本的な知識

と操作を教室外とし、その理解の確認と次の学習を教室で行うこととした。この枠組みを実現するために、学習向けのオンラインプログラミング環境と、自習に要する VOD のビデオ教材とを e ラーニングに実装し、導入する教育方法を策定した。

今後の課題は、本研究のプログラミング教育を試行的に導入し、e ラーニングに実装した各機能と教材による教育効果を確認することである。

[参考文献]

総務省　プログラミング人材育成の在り方に関する調査研究報告書．若年層に対するプログラミング教育の普及推進（平成 28 年度～）．(オンライン)2015 年．(引用日: 2018 年 2 月 24 日．) http://www.soumu.go.jp/main_sosiki/joho_tsusin/kyouiku_joho-ka/jakunensou.html.

鈴木昌和, 小林　真　視覚障害がある生徒達を対象としたプログラミング教育ワークショップの試み. 日本学術振興会 平成 29 年度科学研究費補助金等による研究集会. 情報アクセシビリティをめぐる諸問題に関する研究集会，2018.

情報処理学会　スマホプログラミング．一般社団法人情報処理学会．情報処理，Vol.59 No.2，2018．ページ:114-141．

Andrei Papancea, Jaime Spacco, David Hovemeyer　*An Open Platform for Managing Short Programming Exercises*．Proceedings of the Ninth Annual International ACM Conference on International Computing Education Research．USA，ICER'13, ACM，2013．pp.47-51．

長　慎也，長島和平，堀越将之，兼宗　進，並木美太郎　オンラインプログラミング環境 Bit Arrow を用いた C 言語プログラミングの授業実践．一般社団法人情報処理学会．情報教育シンポジウム，2017．ページ:121-128．

伊藤一成，吉田　葵，安彦智史，竹中章勝，中鉢直宏　過去体験を重視した横断型プログラミング授業の設計と評価．一般社団法人情報処理学会．情報処理学会研究報告，コンピュータと教育，Vol.2016-CE-134 No.13，2016．ページ:1-13．

森　秀樹　Scratch を用いた文系大学生向けプログラミング教育．日本教育工学会．日本教育工学会論文誌，Vol.34, Suppl，2010．ページ:141-144．

私立大学情報教育協会　学士力に求められる情報活用能力の考察．大学教育への提言　未知の時代を切り拓く教育と ICT 活用 2012 年版．公益社団法人私立大学情報教育協会，2012，ページ:254-327．

Tadashi INAGAKI, Yasuhiro SATO　*Analysis of a Flipped Classroom Focusing on Learners' Video Viewing Logs and Notes at Home*．Japan Society for Educational Technology．Educational technology research, 39(1), 2016．pp.125-133．