10.14943/J.HighEdu.28.65
J. Higher Education and Lifelong Learning **28** (2021)
〈Report〉

Implementation of Programming Education as General Education for Beginners in Online Classes

Izumi Fuse[1, 2)*]

1) Information Initiative Center, Hokkaido University

2) Institute for the Advancement for Higher Education, Hokkaido University

初学者を主対象とする大学の一般プログラミング教育のオンライン授業による実施

布施　泉[1, 2)**]

1) 北海道大学情報基盤センター

2) 北海道大学高等教育推進機構研究員

Abstract ― In this paper, we report on the implementation of effective programming education as general education on the premise of the online class. With the promotion of programming education in elementary and secondary education, it is expected that the differences in programming proficiency among university students will increase in the near future. In addition, it is expected that online education will continue to be promoted due to the influence of COVID-19. Classes are conducted using a programming environment originally developed as a Moodle plugin. This environment has a function to check the learner's program execution log and visualization log. Learners can also send questions to teachers in the environment. However, for beginners, there are times when it is difficult to question properly using the questions function of the environment. In such cases, teaching methods that mimic face-to-face lessons using both a web conferencing system and the programming environment are considered effective.

(Accepted on 4 January, 2020)

1. はじめに

新学習指導要領では初等中等教育段階でのプログラミング教育が強化され，2020 年度より小学校での実施が開始されている（文部科学省 2020）。今後 2021 年度には中学校，2022 年度から高等学校で新学習指導要領による授業が進められ，それらの授業を受けた学習者は 2025 年度から大学に入学することになる。一方，2020 年度現在は，大学入学時におけるプログラミング経験者は入学者の 2 割程度に過

*) Correspondence: Information Initiative Center, Hokkaido University, Sapporo 060-0811, Japan
E-mail: ifuse@iic.hokudai.ac.jp
**) 連絡先：060–0811　札幌市北区北 11 条西 5 丁目　北海道大学情報基盤センター

ぎない。そのため，主に大学初年次に行う一般教育におけるプログラミング教育では，初学者を前提とした授業を実施する必要がある。また，2025 年度以降においても旧学習指導要領を履修して入学した学生もいることから，このようなプログラミング教育の過渡期において，学習者の習熟度に差がある状態での授業の実施が今後も求められていくことが予想される。

2020 年度は covid-19 の影響により，大学の授業をオンラインで行わざるを得ない状況が生じている。著者らが担当してきた一般教育としてのプログラミングの授業でもそれは例外ではない。今後もしばらくは同様の影響が考えられることから，オンライン授業として効果的にプログラミング教育を実施する方法の検討が不可欠である。

著者らはこれまで，初学者を対象とした授業用のプログラミング学習環境を開発し，実践を行ってきた（布施 2016, 布施 2018）。当該環境は，少数ながら同時双方向の遠隔教育でも用いてきた経験があり，オンライン授業との親和性は高いと考える。一方で，当該環境は，基本的には対面での授業を前提として利用してきたものである。

以上の背景から，2020 年度は，当該プログラミング環境をフルオンラインの授業内で利用することとした。本報告は，この 2020 年度の教育実践を通し，学習者のプログラミング習熟度に差がある状態の中で，一般教育としてのプログラミング授業をオンライン授業として効果的に実施する方法について検討することを目的とする。

2. 初学者を対象とした授業用プログラミング環境

著者らのプログラミング環境は，学習支援システムの一つである Moodle のプラグインとして開発を行っている。そのため，Moodle コースの権限に応じた管理ができる。例えば，Moodle コースで教師権限を有する利用者は，参加者の各種履歴を確認できる。

本プログラミング環境で使用可能なプログラミング言語は，Ruby，Python，JavaScript である。

2.1 学習者機能

本プログラミング環境は，学習者がログインした Moodle コース上からアクセスする。図 1 は，アクセス直後に遷移した画面において，マウス右クリックで「新規ファイル」を生成する際の画面例である。

図 1 の画面上部にはタブが 5 つあり，左から「フォルダ」「ライブラリ」「ギャラリー」「質問」「コースへ戻る」としている。アクセス直後は「フォルダ」タブにある。学習者毎に固有のフォルダ領域を割り当てており，学習者はその中で自由にプログラムを作ることができる。

フォルダ内に生成したプログラムファイルをダブルクリックすることでプログラムの編集画面となる（図 2 参照）。学習者は，初学者を想定してきたため，集中力をもってプログラムを作ることができるように，授業資料の閲覧（図 2 の右上領域），プログラムの編集（図 2 の左側領域），プログラムの実行（図 2 の左上のボタン），実行結果の確認（図 2 の右下領域）を，一つの画面で行えるように工夫している。オンラインのプログラミング環境であるため，学習者は，どこからでも同じ学習環境で利用でき，いつでも続きからプログラムを作ることが可能である。機能詳細は参考文献（布施 2018）を参照されたい。

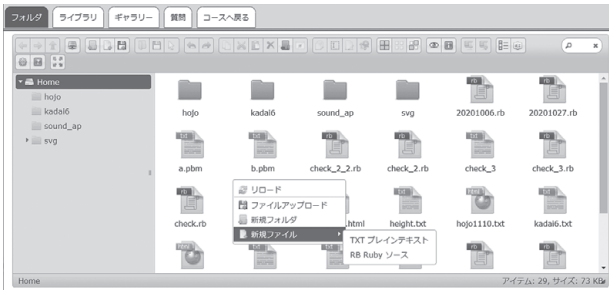

図 1. プログラミング環境に遷移後の画面例（マウス右ボタンで新規ファイルを生成する画面例）

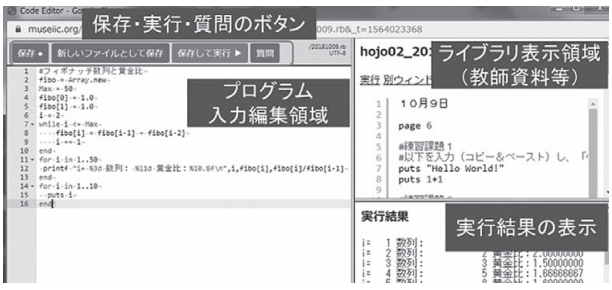

図 2. 編集画面例．ボタンは「保存」「新しいファイルとして保存」「保存して実行」「質問」の 4 つある

本環境では 2019 年度に質問機能を追加した。Moodle コースの教師権限を有する担当者（教員・TA）に向け，図 2 の左上「質問」ボタンから質問をすることができる。質問ボタンを押下した際の画面例が図 3 である。質問メッセージは，質問時に編集していたプログラムとともに Moodle 教師権限を有する担当者に送付される。なお，図 3 では対面授業を想定して座席を記載できるようにしているが，2020 年度はオンライン授業のため，実際の質問では，座席情報は記載していない。教授者からのコメントや回答は，図 1 の質問タブから確認できる。

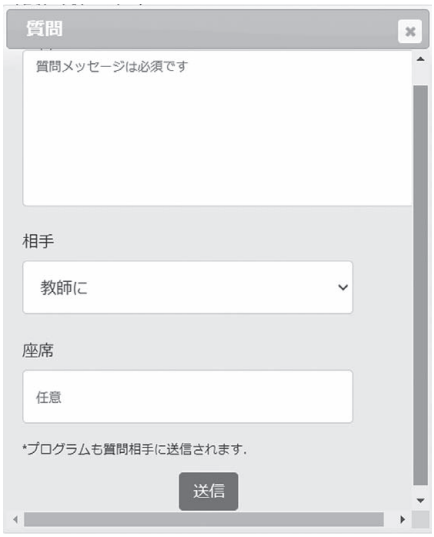

図 3. 質問ボタン押下の際にポップアップする画面

2.2　教師機能

Moodle コースの教師権限を有する利用者は，本プログラミング環境では管理タブも付与されており，計 6 つのタブがある。管理タブを押下した際の画面例を図 4 に示す。

図 4. 教師権限における管理機能の画面例

管理タブ下にある学習者の状況把握機能として，「レポート」「実行ログ」「可視化ログ」の 3 種がある。「レポート」では，学習者毎のライブラリ取得数（本実践では教師資料の取得数に対応する），ギャラリーへのファイルアップロード数，プログラムの実行回数，ライブラリの閲覧数の一覧が確認できる。「実行ログ」では，学習者の最新実行ログ一覧に加え，学習者毎の実行ログをプログラムとともに確認可能である。「可視化ログ」では，時間を区切り，プログラミング環境の利用状況が確認できる。図 4 は「実行ログ」を押下した直後の画面であり，学習者の最新の実行ログ一覧が表示されている。左列のユーザ名はリンクとしてクリックでき，各学習者のリンクをクリックすることで当該学習者のプログラムの実行ログの履歴を確認できる（図 5 参照）。

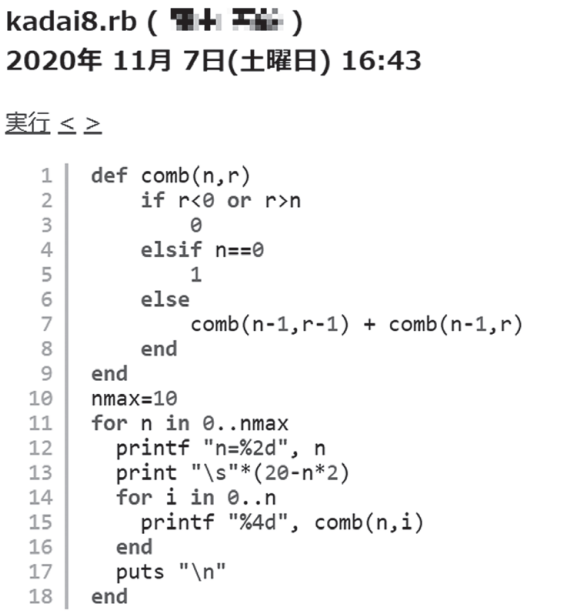

図 5. 上：ある学習者の実行ログ一覧．下：実行プログラムの表示例と同名ファイルの実行ログ遷移リンク

各学習者の実行ログは，図 5 上図の右列の操作欄から確認する。操作の種類は，プログラム表示，プログラム実行，実行時のフォルダ表示の 3 種あり，アイコンで示している。プログラムの表示アイコンをクリックした例が図 5 の下図である。図 5 の下図では，学習者が当該プログラムを実行した時刻が明示され，その下にある「実行」リンクをクリックすると，当該プログラムを実行できる。また，その右にある，＜，＞リンクから，同名のプログラムファイルの実行プログラムの変遷を確認できる。

「可視化ログ」例は図 6 に示す。

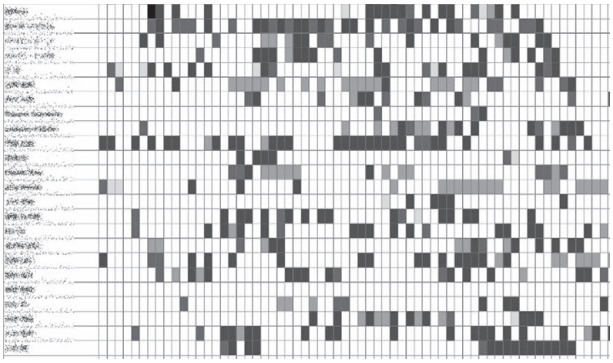

図 6. 指定時間での学習者状況の可視化（60 秒毎）の例

開始時刻，終了時刻。時間幅を指定して可視化する期間を設定する。縦軸は学習者一人ひとりに対応し，横軸が時間の流れとなっている。学習者状況は色分けされており，各色の意味は以下の通りである。

緑：実行および実行完了（エラーなし）
赤：実行エラー
薄青：編集開始
橙：質問
黄：質問に回答
濃青：ファイル保存
紫：ギャラリー閲覧
桃：ライブラリ閲覧

緑が続いている場合には，プログラムを問題なく実行している状態であるが，赤が続いている場合には，プログラムのエラーが解消できていないことを意味し，詳細確認をした方が良い場合がある。なお，可視化ログにおいて，実行エラーである赤マスのみクリック可能となっており，赤マスをクリックすることで，当該エラーをしているプログラムを直接表示させることができる。当該プログラムは，図 5 下図と同様に，「実行」リンクの右側にある＜，＞のリンクから，同一ファイル名における実行時の変化を確認することができ，学習者が，当該プログラムをどのように修正しようとしたのか，その際にどのようなエラーの遷移となっているか等を確認することで，当該学習者が理解できていない箇所を把握可能であると考える。

前節では，学習者が発した質問は，プログラムとともに教師権限を有する利用者に送付されると記した。送付された質問は，教師権限ユーザの質問タブにリスト化されて表示される。図 7 に例示する。

図 7. 質問リスト例

質問毎に区切られており，下部にあるアイコンから，当該質問時のプログラムの表示，プログラムの実行，当該質問時点の学習者のフォルダ状況，コメントの付与等を行うことができる。教授者（教員もしくは TA）が回答する際には，コメント付与ボタンを押下し，コメントを付与する。コメントを付与した例を図 8 に示す。本コメントは何度もやり取りが可能であり，教員からのコメントに対し学習者は再度質問もできる。

図 8. 学習者の質問と教授者によるコメント回答例

3. 一般プログラミング教育のオンライン授業実践

本章では，前章で紹介したプログラミング環境を用い，オンラインでの授業実践の状況を報告する。

3.1　授業概要

著者らは，2020 年度後期に 2 単位の一般教育としてのプログラミング授業を 2 つ担当している。一つはプログラミング言語として Python を用い，もう一つは Ruby を用いて行うもので，ともに選択科目である。前者は 2020 年度に初めて開講し，後者は 15 年以上にわたり行ってきた長い経験を有する授業である。ともに履修者数の制限をしており，各科目 100 名弱が履修している。担当教員 2–3 名で実施しており，TA が数名ついている。

毎年度前期には必修の情報教育が行われており，2019 年度からは Python を 3 週ほど行っている。その結果を踏まえ学習者は選択として履修しているものであり，興味関心が高いことが推定される。しかしながら，その習熟度は様々である。基本事項の習得が不十分で，独力でプログラムを作成することが困難な学習者を初学者とみなすこととすると，2020 年度後期における授業開始時点で，初学者とみなせる学習者が相当程度存在すると考えられる状況である。第 1 章で述べた通り，昨今のプログラミング教育の推進で，初学者は 2025 年度を境に減少することが想定されるが，今後もゼロにはならないと思われる。そのため，初学者のオンライン授業での振る舞いの把握し，効果的な対応についての検討が不可欠である。一方で，初学者と経験者が混在し，授業開始時点ですでに習熟度レベルには大きな幅がある状態のため，初学者のみに焦点をあてると経験者にとってつまらない授業となる可能性が高い。そのため，本授業では，基礎課題＋応用課題の組み合わせとして構成することとし，応用課題では，多数提示されている課題から各自が興味のある課題を選択し（あるいは自由課題を設定し），自身でプログラムを作成していくこととしている。また，応用課題の提出の際はプログラムを提出させるだけではなく，自身のプログラムにおいて，どこを工夫し何に苦労したか等をレポートとしてまとめることを課している。

このような授業構成を Ruby 言語の授業で継続的に行ってきたが，習熟度に幅がある状況でも，ある程度柔軟に対応可能であることが分かっている。そのため，2020 年度は Python を含む先の 2 科目で同様の構成で授業を行うこととした。基礎課題としては，以下の 10 個の課題を提示し，必ず完了することを課している。

表 1. 基礎課題内容（Ruby と Python は原則同構成）

基礎課題番号	概要
基礎課題 1	四則演算
基礎課題 2	実数の絶対値の上限・下限
基礎課題 3	変数
基礎課題 4	素数判定プログラム（穴埋め）
基礎課題 5	平方根の逐次近似
基礎課題 6	九九の表 Ruby：HTML のテーブルとしてファイル出力 Python：各行・各列の和と対角和を追加して出力（標準出力）
基礎課題 7	配列・リストとファイル入出力
基礎課題 8	再帰：パスカルの三角形
基礎課題 9	bit 演算
基礎課題 10	正規表現

3.2　オンライン授業におけるプログラミング環境

前節のように，基礎課題と応用課題の組み合わせによる授業構成とした場合，前章のプログラミング環境をどこまでどのように使わせるかという問題に

到達する。

まず，基礎課題については，科目の履修者全員が，基礎課題に取り組む過程を含め，すべて本プログラミング環境のみで行うこととした。その上で，プログラミング環境における各種ログを確認し，プログラミングを行っていないと判断される場合には，当該週の授業は欠席とみなすことを含め周知することとした。

一方，応用課題では，SVG や Tk を用いた描画を伴う課題をいくつも設定している。また，音の生成や錯音を確認する応用課題では，ある程度の計算量を要し，集団でサーバを使うには負荷が高いものも含まれる。なお，本プログラミング環境は，構築上の制約から，以下を行うことはできない。

・キーボードからの入力
・Tk を用いての描画

応用課題は，基本的には学習者の選択に応じ，プログラムの作成の自由度を高めることが望ましい。できるならばローカル PC 上に個別にプログラミング環境を構築して応用課題に取り組むことが望まれる。一方で，ローカル環境を学習者が各自の PC で問題なく設定できる確証はない。もしも問題が生じた場合には，その状況を回復するための労力は，オンラインのみの授業では，より多くかかることが想定され，それは科目の本来の目的であるプログラムを検討する時間にも制約を与える可能性がある。

これまでは，大学設置の PC 教室で授業を行っていたため，このような固有の環境構築を考える必要はなかった。本授業は全てオンライン授業になったものの，2020 年度後期の授業では，大学の行動指針レベルの状況により，大学設置の PC を使うことも可能であるため，以下の方針で応用課題を進めることとした。

・応用課題はプログラミング環境を用いて実施可能なものとそうでないものに分けて提示する。
・ローカル環境を構築する必要はないが，構築したい人向けに手順書は用意する。
・環境設定に自信がない場合は，原則，第 2 章で示したプログラミング環境で行うことのできる応用課題を選択する。
・環境設定に自信がないものの，プログラミング環境を用いては実施できない応用課題を行いたい場合には，大学の PC を利用することを検討する。

なお，基礎課題を早々に終了し，早く応用課題に取り組む場合であっても，授業 15 回のうち前半 8 回が経過するまでは，必ず本プログラミング環境を用いることとした。早く基礎課題が終わった学習者に対しては，第 8 回までは，まずはプログラミング環境で行うことのできる応用課題に取り組むよう指導した。この措置により，前半終了時点での各学習者のスキルと状況把握が行える状況を確保している。

3.3　オンライン授業の実施

授業は基本的に，オンデマンドで資料を提示し，本プログラミング環境を用いて基礎課題に取り組むこととした。オンデマンドではあるが，必ず授業時には取り組み，取り組み状況の記録を作業記録として毎時間提出することを課した。また授業時はウェブ会議システム上により質問対応を設定しており，学習者の質問に直接対応できるようにした。その上で，学習者への個別の対応をきめ細かく行うために個別の進捗確認を短時間確保して行うこととし，担当教員が分担対応することとした。担当者と履修者との数の関係で，各学習が，個別の進捗確認を 3–4 週毎に 1 回実施できるようにし，確認スケジュールを学習者に周知して対応した。

3.4　学習者が躓いた際の対応

学習者が躓いた際の対応は基礎課題と応用課題で異なる。基礎課題についてはきめ細やかな対応を行うように工夫し，応用課題については自身で考えるためのヒントを適宜出す程度にとどめている。

基礎課題における学習者の理解度の把握は，毎時間の作業記録における質問内容，プログラミング環境における質問ボタンからの質問状況，プログラミング環境におけるエラー状況等から確認し，総じて理解できていない項目がある学習事項に関しては，補助的な追加資料を用意した。例えば，print 文の使い方や繰り返しの利用による和の計算等についてである。テキスト資料，ビデオ資料などを追加資料と

して提示したことに加え，関連の小テスト（成績には関係しない）を設定し，それを行うことで，理解を促そうとした。

しかしながら，そのような追加資料や小テストの仕組みを作っても，Python における基礎課題 6 に苦戦した学習者が特に多かった。追加資料や小テストを確認しない学習者もいるのであるが，追加資料を確認しても理解できない学習者も存在した。苦戦した内容は，2 次元リストにおける行・列の和と対角和を求める箇所である。

当該課題に特化した内容を扱ったビデオ資料を提示したにもかかわらず，それを視聴しても理解できないという学習者が複数いたため，当該課題特化型のウェブ会議システムを設定し，理解できなかった学習者へのフォローアップを行うこととした。後述するが，この対応は対面授業にかなり近い形で行うことができ，効果があったと推察している。

4. 考察

本章では，学習者の躓き解消の手法について考察を行う。まず，ある程度のプログラミングの文法を理解している状況で，何らかの理解が足りないために問題を起こしている場合には，質問ボタンから，適切な質問ができると考えられる。学習者自身が，何ができ，何が問題なのかを示すことができるため，教授者からのコメントのやり取りで問題は解消可能であると考える。

しかしながらその状態に至らない初学者が，一定程度存在する。基礎課題の実施過程で，プログラミングの基本事項を一つずつ積み上げるようにテキストを構成しているものの，各課題や関係の例題を漫然と行っているだけでは，全体の理解が不十分な場合があるためである。例えば，変数を用いて和を求める例題は行っているものの，各命令の意味を確実に確認していない場合には，自身で似たようなプログラム（ここではある値の和を求めるプログラム）を一から作成することはできない。

このような初学者の躓くポイントは，対面でもオンラインでも原則同様ではあるが，その躓きの解消方法は，その授業形態によって，効果は当然変わり得る。本授業は，オンライン実施であることから，まずは各種資料提示や小テストにより解消を行おうとしたが，状況の改善は芳しくなかった。これは，前述の学習者自身の課題の取り組み方の問題でもあるが，どの学習項目の理解が足りていないかの判断を学習者自身ができていないことが原因であるためと思われる。そのため，本授業実践では，当該課題に特化したウェブ会議システムを設定し，丁寧に指導することを連絡し，希望者がアクセスする形式としたところ 14 名の参加があった。対面授業のように，まずは担当教員が当該課題の趣旨，求めるべきもの，必要な要件などを示しながら，学習者たちに説明を行う形式とした。その上で，必ずプログラミング環境を各自が立ち上げ，教授者の説明とともにプログラムを実行させながら，課題に取り組む演習型とした。こちらで用意した内容をうまく実行できない場合には，その時点で必ず声を上げるように徹底させた。実際にうまく行かないプログラムが生じた際には，そのプログラムをチャット上に掲載させ，教授者が，当該プログラムがなぜうまく行かないかのコメントをするなど，何が問題であるかを示すように心がけた。結果として，行の和と対角和については 14 名とともに 40 分程度取り組み，列の和については 20 分程度を使い，個別で課題を行わせた。最終的にできなかった場合には質問ボタンから必ず質問するように促したところ，11 名が当該授業時にほぼ完成させた。残り 3 名は，プログラムのエラーの状況を確認の上，個別に問題を指摘することで，次回までに参加者全員が課題をクリアすることができた。

このように課題をクリアできない学習者が多い課題については，課題の意図の説明を含めウェブ会議システムを用いて行うとともにプログラミング環境を併用することで，対面授業に近い形で実施できることが確認された。また，授業終了後に参加した学習者のログを確認することで，うまくできなかった学習者へのフォローアップも可能であることが確認できた。なお，授業後に参加者 14 名のプログラミング状況を可視化したところ，授業時に完成できなかった 3 名のうちの 1 名は，指示とは別の手順で取り組んでいたことが判明した。課題を達成させるプログラムの作成方法は一通りではないため，各自の

流儀で作成したい気持ちは理解できる。このような学習者には本来の質問対応として対応することが望ましいと考える。

5. まとめ

本稿では，大学の一般教育としてのプログラミング教育において，習熟度が異なる学習者に対し，効果的なオンライン授業の実施について検討を行った。プログラミング環境の制約により，応用課題についてはローカル PC での実施も対象とするが，基礎課題の取り組みはすべて本プログラミング環境を使って行うこととして実施した。本プログラミング環境による実行ログや可視化ログを用いることで，特に初学者に対する状況把握が行いやすくなっている。

一般的な質問対応は，プログラミング環境やウェブ会議システムを用いて行うことで有効に機能するが，さらに初学者に対しては，分からない段階に応じ，対面授業のような形式でのウェブ会議システムの利用も効果があると考えられる。これは従来，対面授業で行っている能力別クラス編成を課題型に特化したものとも考えられる。

これらの経験を踏まえ，次年度のオンライン授業の構成について検討を進めていきたい。

参考文献

文部科学省（2020），「小学校プログラミング教育の手引（第三版）」，https://www.mext.go.jp/content/20200218-mxt_jogai02-100003171_002.pdf（2021 年 1 月 1 日閲覧）

布施泉・中原敬広・岡部成玄（2018），「プログラムの相互利用と相互評価が可能な初学者用プログラミング授業支援環境の構築」，『教育システム情報学会誌』35 (2)，221–226

布施泉・中原敬広・岡部成玄（2016），「授業での利用を前提とした初学者用プログラミング学習環境の開発」，『大学 ICT 推進協議会年次大会』，FE22，2016.