

オンライン授業での初学者用プログラミング学習環境の検討

布施 泉¹⁾, 増本 広和¹⁾, 山本 裕一¹⁾, 平林 義治¹⁾

1) 北海道大学 情報基盤センター

ifuse@iic.hokudai.ac.jp

Investigation of Programming Learning Environment for Beginners in Online Class

Izumi Fuse¹⁾, Hirokazu Masumoto¹⁾, Yuichi Yamamoto¹⁾, Yoshiharu Hirabayashi¹⁾

1) Information Initiative Center, Hokkaido University

概要

プログラミング初学者は様々なミスをしながら学習を進める。本稿では、学習者の学習状況に応じた適切な支援をできるよう、オンライン授業における初学者のためのプログラミング環境について検討する。著者らがこれまでに構築してきた初学者に対する授業用プログラミング学習環境での学習者状況の可視化情報に加え、ウェブ会議システムを用いた個別の進捗確認、ならびに初学者からの質問対応を踏まえた追加資料の提示を組み合わせた授業実践について報告する。

1 はじめに

2020 年度における大学の教育は、COVID-19 の影響でオンライン授業を中心に実施せざるを得ない状況にある。本報告は、このような状況下において、大学の一般教育としてのプログラミング教育における初学者支援に焦点を当て、望ましい学習環境について検討するものである。

本報告で検討の前提とする授業は、選択科目の一般教育で行う 2 つのプログラミングの授業である。プログラミング言語は、Ruby と Python をそれぞれ取り上げ、受講者は各 100 名程度である。殆どの学習者は、それ以前に行われた必修授業で 3 コマ程度はプログラミングを学んでいるもののその習熟度は様々である。本稿では、それ以外のプログラミング経験が殆どなく、学習開始時点で、基本事項の習得が不十分で、独力でコード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。本授業では、初学者と経験者が混在しており、学習開始時点で習熟度レベルには幅がある。選択科目であり、総じて学習者の興味関心は高い。2020 年度時点の教育カリキュラムは、初学者を前提として組むこととしているため、習熟度に幅がある学習者に対する到達レベルの設定と課題難易度には工夫が必要である。

著者らは授業の最初に履修の理由について学

習者に問うている。「興味があったから」「必修授業でのプログラミングが楽しかったから」という比較的単純な回答の他、「プログラミングに興味があったが自分では何をしたらいいのかよくわからなかった」「プログラミングに興味があるが、自学では躓きそうなのでアドバイスをもらいながら学びたいと思った」「プログラミングを学ぶのは、独学では難しく感じられ、この機会を逃すと学ぶことができそうになかったから」など、初学者において、自学での不安を理由とした内容、授業での支援に期待する内容が例年散見される。

2019 年度までは、Ruby の授業のみを開講しており、対面で 30 名前後の比較的少人数にグループ分けをして授業を実施していた。この場合、授業の中での具体的なやり取りで、学習者の状況を確認し、支援することが可能であった。しかし、2020 年度は、COVID-19 のためオンラインでの実施となったことから、対面での授業運営は行うことができない。授業でのプログラミング教育として、特に初学者の不明点や不安をどのように支援していくのかが効果的であるかが喫緊の課題となったため、2020 年 10 月現在、試行的な実践を行い、その効果の検証を進めている。本稿ではその内容を報告するものである。

2 初学者用プログラミング環境

著者らは、授業での利用を前提とした初学者用のプログラミング学習支援環境を開発し、当該環境を用いた教育実践を行っている[1][2]。この環境を本稿でのオンライン授業にて用いることとする。当該環境の特徴は、教師が自身の授業資料を学習者の状態に合わせて提示でき、学習者は、授業資料の閲覧、プログラムの編集、実行までを一画面で行うことができる点にある。また、Moodleプラグインとして開発しており、教師は学習者のウェブブラウザ上での実行等の履歴を管理できる。学習者は自宅でも大学でも同じ学習環境でプログラミングの学習を行うことができる。

著者らが開発したプログラミング学習環境を図1に例示する。プログラミング言語は、Ruby, Python, JavaScript に対応している。

本学習環境上では学習者毎にフォルダ領域が割り当てられ、その中で自由にプログラムを作ることができる。作成した各プログラムファイルを

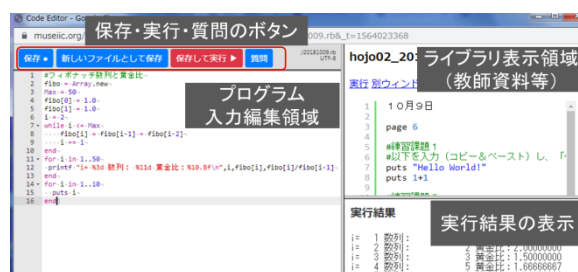


図1 授業で用いるプログラミング環境

ダブルクリックすることで、ファイル毎に図1のようなプログラミング環境が立ち上がる。本環境では、1画面で学習者がプログラム作成に必要な操作ができるように工夫して構築している。画面右上は、学習者の取り込んだライブラリ内のファイル(教師の補助資料等)を表示する領域である。

2.1 学習者状況の可視化

オンライン授業では特に学習者の進捗状況を適切に把握する機能が必要となる。教師は本プログラミング環境の管理画面から、可視化をしたい開始・終了時刻、及び、時間幅を指定し、「表示期

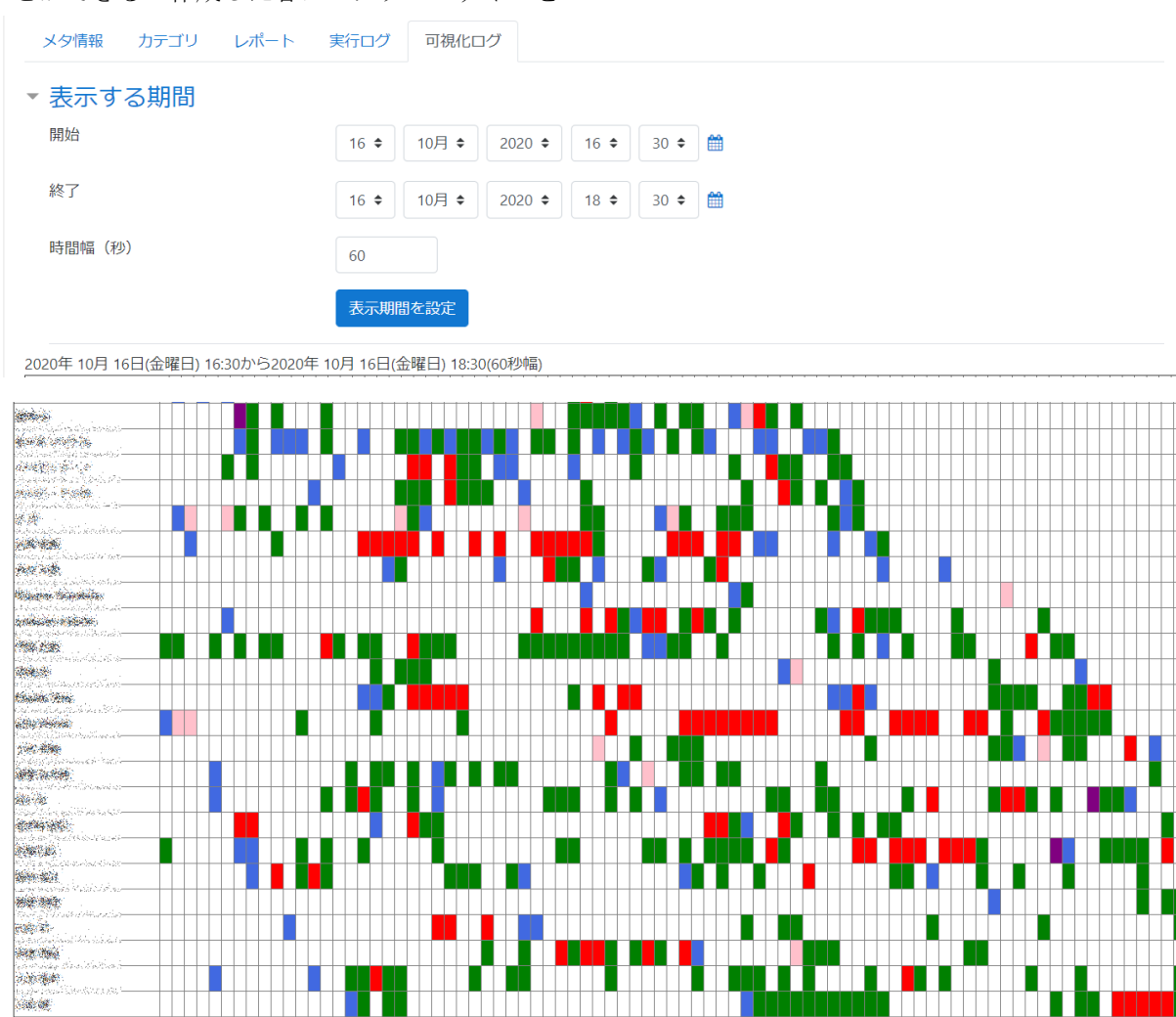


図2 指定時間における学習者状況を可視化した画面例

間を設定」ボタンを押下することで学習者の状況が表示される。図2は、60秒毎の可視化例（実際の画面を一部抜粋したもの）である。横軸が時間の流れであり、縦軸が各学習者に対応する。つまり、1分毎の各学習者の状況が示されている。学習者の状況は、色で区別しており、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、プログラムの実行を行っている状態であり、各人は基本的に問題なく授業を実施していることが推察される。

図2では2020年度におけるある授業コマの記録を可視化させたものである。本来、基本的な流れとして、学習者がライブラリを閲覧し、プログラムを編集し、プログラムを実行するというものを想定するものの、エラーを連続して何度も発生させていたり、色が殆どつかずスカスカな状況である学習者もいる状況が見て取れる。学習者には授業コマでは、プログラミング環境を使って授業を実施するように伝えても、授業コマでは特段の作業をせず、授業コマ以外で実施しようとする学生がいることが想像され、指導が求められよう。

連続エラーをしている学生支援については、本プログラミング環境では、エラーが発生した赤マスをクリックすることで、当該学習者のエラーコードを確認できるようにしている。図3に、赤マスをクリックした際に立ち上がる実際のエラーコード例を示す。本画面から教師はエラーコードの

実行 ≤ ≥

```

1 | print(2==3)
2 | print(3>1)
3 | print(-1>0)
4 | print('a'>'b')
5 | print('こんにちは'=='こんにちは')
6 |
7 | a=15,b=12
8 | print(a<b)

```

図3 赤マスクリックからのコード表示例

実行を上部の「実行」リンクから直接できる。また、実行リンクの右側にある”<” ”>”のリンクから、同一ファイルでの当該コードの実行時の変化を確認することができる。学習者が、当該コードをどのように修正しようとしたのか、実際にどのようなエラーを出しているのかを確認することで、当該学習者の理解できていない箇所を把握することを想定している。

図2で、気になる状況がある学生が表出した場合、個別の学習者のプログラムの実行状況を、時系列を追って確認することができる。図4にその画面例を示す。図4は、管理画面の「実行ログ」タブを表示させたものである。指定された期間における最新の実行プログラムの一覧が確認できる。縦軸は学習者に対応する。図4の画面右には、プログラムコードの操作が3種類用意されている。操作欄左のメモ帳のアイコン  を押下すると、プログラムコードが閲覧できる。中央のアイコン  は、押下することで、当該プログラムコードの実行を行い、実行結果を表示する。右のアイコン  は、押下することで当該学習者のフォルダ内を確認することができる。図4における左列のユーザ名にあるリンクをクリックすると、当該ユーザにおける表示期間内での実行履歴を確認することができる（図5参照）。

実行ログ

直近の実行プログラム一覧

表示する期間

開始 10分 10月 2020年 20分 30分 Yes

終了 10分 10月 2020年 20分 30分 Yes

表示期間を設定

ユーザ名	ファイル名	日付	操作
ユーザ名	kada5.py	2020年 10月 16日(金曜日) 17:48	  
ユーザ名	20201016.py	2020年 10月 16日(金曜日) 17:51	  
ユーザ名	kada4.py	2020年 10月 16日(金曜日) 17:51	  
ユーザ名	kada5.py	2020年 10月 16日(金曜日) 17:49	  
ユーザ名	kada3.py	2020年 10月 16日(金曜日) 17:53	  
ユーザ名	kada5.py	2020年 10月 16日(金曜日) 17:54	  
ユーザ名	新規ファイル.py	2020年 10月 16日(金曜日) 17:55	  

図4 学習者の最新実行コード一覧

実行ログ

ユーザの実行コードリスト

表示する期間

開始 10分 10月 2020年 20分 40分 Yes

終了 10分 10月 2020年 20分 40分 Yes

表示期間を設定

戻る

ファイル名	日付	操作
kada1.py	2020年 10月 16日(金曜日) 18:08	  
kada1.py	2020年 10月 16日(金曜日) 18:08	  
kada1.py	2020年 10月 16日(金曜日) 18:08	  
kada1.py	2020年 10月 16日(金曜日) 18:08	 

図5 ある学習者の実行コード履歴の表示

2.2 学習者との本プログラミング・システムを通じた質問応答

初学者がプログラミングを行う際には、想定しない結果が出たり、エラーが消えなかったりする場合がある。前節では、学習者の可視化により全体状況を確認するシステム機能について紹介した。しかし、本来は学習者の能動的な質問に対し支援を行いたいと考える。学習者がエラーを起こしているとしても、本人がその内容を自身で考えて解決したいと考えている場合があり、一律の支援が望ましいとは限らないからである。一方で、理解が不十分な学習者はどこがわからないかを適切に伝えることが難しい場合もある。

このような状況に対応するため、本プログラミング環境は、学習者の能動的な「質問」を当該コードに紐づけて行う機能を有する。質問ボタンとそれに対する回答を行う機能であり、ポイントは以下の3点である。

- ・ 各プログラムファイルと質問フォームを紐づける（図6参照）。
- ・ 質問者自身が、教師のみか、他の学習者を含めてかを選択して質問できる（図6参照）。
- ・ 他の学習者宛ての質問であれば、他の学習者が確認し、コメントすることも可能である（図7、図8参照）。

図6は質問画面例である。図1に示したプログラム実行画面上部に、「質問」ボタンがあり、当該ボタンを押下した場合に、図6の表示が出る。各プログラムで押下した質問フォームであり、質問とプログラムが紐づけられて管理されている。

図6 質問フォーム画面例

質問フォームは、質問内容、質問を答える対象（「教師に」「みんなに」を選択）、座席（記載は任意）の3項目から構成される。座席は、対面を想定した機能であり、オンライン時には用いない。質問を求める対象「みんなに」は学習者間の相互解決を意図して設定したものであるが、2019年度は「教師に」のみが選択されている状況である。

質問を送信すると、質問タブにその内容が表示される。また、教師はMoodleトップのメッセージで、質問が届いたことを確認できる。図7は2019年度の授業における実際の質問タブでの質問リストの表示例である。質問とともにプログラムが紐づけられており、実際に質問のあるコードを実行して確認し、結果を踏まえて、アイコンから応答コメントが可能である。2019年度の最終授業では、質問機能を用いた遠隔地からの支援の試行も行ってきたが、順調に機能していることを確認している。図8は図7の一覧の質問にあるアイコンをクリックし回答コメントを出した際の質問応答の画面例である。

図7 質問リスト例

図8 質問への回答例

3 学習者の状態把握と学習支援

「はじめに」で述べた一般教育における選択科目であるプログラミング授業をオンラインで実施するにあたり、前章のプログラミング学習環境を用いることとした。

一般に、オンライン授業では、同時配信授業もしくはオンデマンド授業を想定することとなるが、授業における学習者の習熟度の幅を踏まえ、同時配信授業を用いることは避けた。また、近年は反転授業をプログラミング教育に活用する事例が増えているが、対面することができない中で、初学者が1週間という短い時間で、期待すべき到達目標まで達することができるかが見極められない可能性が高いと考え、反転学習の実施は控え基本的には資料を確認しながらのオンデマンド型で行うこととした。但し、オンデマンド型と言うだけであると、学習者は授業時間でも授業時間外の時間でも取り組まず、授業後半に固めて学習内容を進めようとする可能性がある。このような事態を避けるために、当該授業時間内に行った内容について記録する「作業記録」を設け、各コマの授業後半でのみ提出できることを課した。「作業記録」の提出率が悪い場合には、出席不足として単位を認めないようにする。

また、授業時間は、TAを含め、ウェブ会議システムを用いた質問対応を行うこととした。しかし、単に質問対応を行っているというだけであると、学習者が質問に来たがらないことが想定されるため、学習者と必ず話すことを前提とした担当者による「進捗確認」を、ウェブ会議システムを用いて行うこととした。

3.1 進捗確認（ウェブ会議システム）

担当者毎にスケジュールを決め、学習者と5分程度の個別面談を行う。「面談」と称すると学習者が委縮する可能性があることから、授業に対する「進捗確認」と称している。ウェブ会議システムは、BigBlueButtonを用いている。このウェブ会議システムは、当該授業の質問対応にも用いている。進捗確認と質問対応に使うシステムと同じものとするすることで、システムに慣れること、また担当者と音声のやり取りを行うことを通じ、その後の質問対応へのアクセスへの敷居を下げる意図もある。進捗確認の主な流れは以下の通りである。

- ・ 進捗確認スケジュールを開始1週間前の授

業で提示する（1人5分として割付）。

- ・ 授業時には、同じウェブ会議システムに、進捗確認のための学生と質問に来た学生が混在する。その回の割り付け担当のTAが、入室者の目的（進捗確認か質問か）を確認し、適宜、担当者の小部屋に割り付ける。
- ・ 進捗確認では、可能な限り音声でのやり取りとし、この進捗確認の意図と実際の進度がシステムで予め把握できていることを学習者に理解させるようにする。本授業におけるオンライン授業の内容理解が不十分と想定される場合は、学習者の理解を確認する。
- ・ 具体的な質問が見受けられ、小部屋での担当者では時間が足りない場合には、親会議室で割り付けをしているTA等に質問対応を依頼する。

約100名の受講者がある授業であり、担当者2-3名であることから、1人5分の進捗確認を行う場合には、1コマ11~12名が限界であり、3週から4週をかけて一回りさせることとする。

2020年度後期の授業で実践しているものであり、10月16日現在、進捗確認を1週行った段階であるが、単純な質問対応に比較して、学習者からの質問等を引き出すことができるとともに、学習者への指導を行うことができることを確認している。

今後、授業が進行していくにつれ、前章のプログラミング環境による個別履歴と具体的にどのように組み合わせ、理解が不十分な点を確認し効率的に指導するかが重要になっていくと考えられるため、効果的方法をさらに検討していく予定である。

3.2 質問応答（授業時・授業時以外）

学生へは質問方法として、以下を提示している。

- ・ ウェブ会議システムを用いる（授業時）
- ・ メールで行う（授業時は対応が遅れる旨記載）
- ・ プログラミング環境の質問機能を用いる（授業時を特に推奨）
- ・ Moodle設置のフォーラム

その他、作業記録内にも質問を記載できるようにはなっているが、基本的には上記における質問に対しては、何らかの回答を行っている（作業記

録上の質問は回答するとは限らない)。現状では、授業時のウェブ会議システムと、質問メールを主として利用しているようであるが、それぞれ1週間に4-5件程度であり、全体としての質問はあまり多くはないように思われる。なお、作業記録には何らかの感想や質問を書くように求めているため、作業記録に質問を書く学習者も一定程度いる。これらの学習者が、進捗確認を経ることで、直接能動的な質問を発信できるようにすることが望まれる。

3.3 追加資料の作成と提供

前節で、質問はあまり多くはないと記載したが、その質問内容は重複することは比較的多い。学習者(特に初学者)にとって理解が難しい項目があることを示している。学習者が何を理解しにくいかを理解することは非常に重要である。そのような項目に対しては、学習者全体へのフィードバックが必要と考えており、コラムとしての追加資料の作成を推進している。

これまで利用してきた授業テキストはあくまで対面授業を前提としたものとなっている。2020年度の授業は、オンデマンド型を主体としていることから、担当者がその場で音声を使って説明をすることはしていない。このようなオンラインでの学習では、学習者の習熟度に応じた追加資料を提示していくことが望ましいと考えており、特に初学者向けには、実践しながらより詳しい内容を確認できるコラムを作成していくこととし、例えば、論理式等に関する追加資料を用意している。

4 考察とまとめ

2020年度の授業は進行中であるため、すべての通した評価は行えてはいないが、これまで述べた実践を通した考察とまとめを行う。

オンライン授業としてのプログラミング授業では、特に学習者の状況把握が不可欠である。オンデマンド型の授業となると、授業時に拘束されなくてよいと安直に考える学習者が多く、その認識を改善させる必要がある。プログラミングは一朝一夕では身につくものではないため、あとでまとめて行おうとした場合には、既に手遅れになっていることも想定される。その意味で、学習者がプログラミングに取り組んだ履歴を可視化し、各週でどの程度、取り組んでいるかを早めに把握できることは大きい。図2は授業時を想定し、1分刻みでの学習者毎の状況を示したが、刻みの時間

幅は任意に変更可能である。実際には、授業時の状況確認には1分という細かな刻みで確認するとともに、1週間の状況把握としては、6時間、12時間、1日刻みといった大きな刻み幅で全体確認することも行っている。1週間何も実行履歴がない学習者、授業途中で履歴が消えている学習者等が時折見受けられる。それらの学生には、その理由を進捗確認で確認して指導する等、きめ細かい指導が可能になったと考える。

進捗確認で音声でのやり取りをしながら、プログラミング履歴を確認しての指導を行うことで、対面に準じた質の個別指導が可能であると考えられる。オンライン授業ではなり替わりや他人の課題のコピペなどの不正行為が問題になる場合もあるが、本稿で実践しているようなウェブ会議システムを用いた進捗確認を繰り返しながら、プログラミング環境上の実行ログを突き合わせ確認することで、そのような危険性を低減化できる可能性があると考えられる。実行ログと進捗確認により、学生のスキルの状況と取り組み時間の把握を引きつづき行っていきたい。

初学者に焦点を当てた場合、可視化ログにおける連続エラーに着目し、早めに対応・支援する必要があると考える。連続エラーを起こす学生には、まずは支援が必要であるかを確認した上で、適切な足場架けを行うことが必要である。この足場架けとして、3章3.3にて追加資料の作成・提示等を学習者の質問に応じて進めていることを示した。

現状で、学習者の質問はメールや質問対応のウェブ会議システムが中心である。プログラミング環境からの「質問」の投稿は、その詳細方法を示していないこともあり、2020年10月16日現在では使われてはいない。一方で、2019年度の授業では、後半の応用課題にて、個別にテーマを決めプログラミングする際には、コードが付与された状況で質問を確認することにより、効果的な回答が安直に可能となった。このような過年度の状況を踏まえ、授業後半に取り組む予定の、より難易度の高い課題における効果的な質問方法として学習者に周知をしていきたいと考えている。特に授業時には、プログラミング環境からの質問を行うことで、問題となるコードを担当者が共有でき、その状態で、ウェブ会議システムにて質問する流れで指導していくことを予定している。

さらには2020年度のオンライン授業の工夫を踏まえ、効果的なプログラミング授業について、

理解を深めていきたいと考える。

参考文献

- [1] 布施泉, 中原敬広, 岡部成玄, ”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 Vol35, No.2, pp.221-pp.226, 2018.
- [2] 布施泉, 中原敬広, 岡部成玄, ”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.