ブロック型教材とオンラインジャッジの結合による プログラミング演習環境の設計

田 中　賴 人[1]

１．はじめに

サイバー大学（以降「本学」とする）は教室を持たず、全ての講義・演習・試験等を遠隔で行うフルオンライン大学である。全ての学習者は自己所有のパーソナル・コンピュータを持ち、講義の視聴や演習は各々の都合にあわせ遠隔・非同期の環境下で進められる。このような特性を持つ本学では、他大学と異なる教育上の困難さとして「学習者の顔や動作が見えない」「演習の際、学習者の画面を見られない」「質問に即座に答えられない」等が挙げられる。

一方、初学者を対象とするプログラミング導入教育の分野では近年 Blockly[1)]、Scratch[2)]、Viscuit[3)]等の視覚的な操作体系を持つ演習環境が普及しつつある。これらは文字でプログラムを記述する従来の環境と異なり、キーボード入力の煩雑さや間違いを回避できる、初学者の特性に適した演習の方法である。しかし、外観の操作性を初学者に合わせるだけでは「教員から学習者の様子がわからない」「質問への応答に時間を要する」という、遠隔・非同期における問題点を解決することはできない。教員が直接学習者の様子を見られないオンラインの環境下では、学習者が自らの作業の正否を確認できる、自学自習を主体としたプログラミング演習環境を用意することが望ましい。本稿では、初学者に適した視覚的な教材と遠隔・非同期に適したプログラム自動採点システムを結合し、演習課題とともに学習者に提供する方法について述べる。

２．導入科目「プログラミング入門」の概要

本学には 7 つのプログラミング科目があり、本稿の演習環境を用いる「プログラミング入門」はそれらの導入部として位置づけられる[4)]。本学の 2 つの学期である春学期、秋学期のどちらにも開講され、1 学期あたりの履修者数は約 200～360 名、プログラミングの経験を持たない初学者を主な対象とし、講師とティーチング・アシスタント（TA）で運営を行っている。同科目の目的はプログラムの作成を通した情報技術への理解を図ることで、

[1] サイバー大学 IT 総合学部・講師

学習者が履修を終えると「C プログラミング演習」「UNIX サーバ構築 I 」「ソフトウェア工学」等の、より専門性の高い、目的志向の強い後続科目と進んでいく。

同科目では対象範囲を制御構造の「逐次実行、繰り返し、条件分岐」とし、初学者を単位修得のレベルにまで導くため、全 15 回の構成を以下のように分割している。

- 第 1 段階（第 1 回～第 5 回）…ゲーム型教材による制御構造の概念獲得
- 第 2 段階（第 6 回～第 10 回）…ブロック型教材による制御構造の応用
- 第 3 段階（第 11 回～第 15 回）…JavaScript による実用的プログラムの記述

次章以降で述べるプログラミング演習環境は、上記の第 2 段階「ブロック型教材による制御構造の応用」で演習に用いられるプログラム作成・実行環境を指す。

３．システム構成

3.1. ブロック型教材 Blockly

「プログラミング入門」の演習では Google による教材 Blockly を採用した（図 1 ）。その理由は全ての操作を Web ブラウザ上で行えインストールの必要がないこと、全てのソースコードを入手できカスタマイズが可能なこと、無料で導入できること、の 3 点である。

Blockly は GUI を持つビジュアルエディタの一つである。学習者は講師からの指示によって画面上にブロックを配置し、演習の要件を満たすようにプログラムを記述する。プログラムの作成と実行、動作確認は Web ブラウザ上で完結するため、学習者による演習環境のインストールや初期設定の作業を必要としない。また本来の Blockly では学習者のプログラムが学習者の端末内の記憶領域（HTML5 の LocalStorage）に格納されるが、本稿の演習環境ではプログラムを演習用サーバに送信し、全ての学習者のプログラムを講師が一括管理するよう、Blockly への機能拡張を行った。この拡張により

- 「自宅と勤務先」のような複数の端末を使い分ける学習者が、プログラムの格納場所を意識せずに演習を続けられる
- 講師が演習用サーバにアクセスして学習者のプログラムを取り出し、演習の進捗確認や助言、質問応答に用いることができる

という 2 つの利点を Blockly に付加している。

図 1　Blockly の画面

3.2. 自動採点システム Sharif-Judge

学習者のプログラムが演習の要件を満たせたか否かはオンラインジャッジで判定され、採点される。プログラミング教育におけるオンラインジャッジの活用は松永[5]や長尾ら[6]の試みにあるように、その有用性が確認されている。

「プログラミング入門」の演習ではオンラインジャッジとして Naderi による Sharif-Judge を採用した[7]。その理由は全てのソースコードを入手できカスタマイズが可能なこと、Blockly から出力される Python のプログラムを自動採点できること、GUI の画面インタフェースと CUI での自動採点処理が分離されていること、の 3 点である。またオンラインジャッジはいつでも呼び出せるので、学習者のペースに合わせた自学自習に適している。

Sharif-Judge の基本的な使用法は、プログラムに対する入力と出力の組を設定することである。一例として、ある自然数の約数を求めるプログラムでは入力が 12 の場合、出力は 6 が正しい。同様に入力が 140 の場合は出力が 12、入力が 360 の場合は出力が 24 となるのが正しい。これらの入出力の組を Sharif-Judge に登録しておくと、学習者が提出したプログラムが自動実行され

・12 を入力した際、6 が出力されたか
・140 を入力した際、12 が出力されたか
・360 を入力した際、24 が出力されたか

を全て満たすか否かで合否が判定される。

本来の Sharif-Judge では図 2 のような Web インタフェースを介して実行されるが、本稿の演習環境では学習者が Blockly を経由してプログラムを提出するため、Sharif-Judge の Web インタフェースを省略するように設置している。

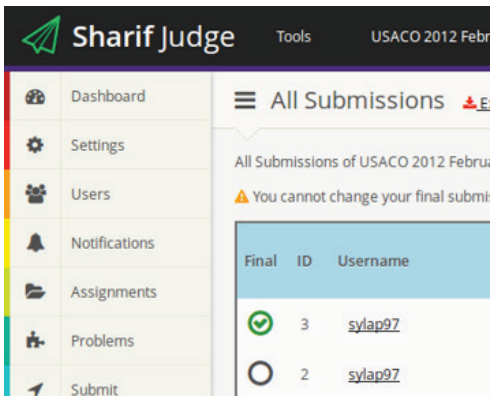

図 2　Sharif-Judge の Web インタフェース

3.3. Blockly と Sharif-Judge の結合

ブロック型教材 Blockly と自動採点システム Sharif-Judge はどちらもプログラミング導

入教育において有用なものであるが、両者は目的が異なる別個のシステムである。そのため、本稿の演習環境では学習者がプログラムを記述してから自動で採点を受けるまでの流れが以下となるように、演習用サーバ内での結合を行った。

1）学習者が Blockly の画面上でブロックを配置し、演習の題意を満たすようにプログラムを作成する。
2）プログラムが完成すると、学習者は Blockly 画面上の「提出」ボタンを押す。
3）学習者の Blockly プログラムが Web ブラウザ内で Python に変換される。
4）変換後の Python プログラムが演習用サーバに送信される。
5）変換後の Python プログラムを演習用サーバ上の Sharif-Judge が実行し、プログラムに対する入力と出力の組が正しいか否かを判定する。
6）判定結果を学習者の Blockly 画面上に表示する。

学習者によるプログラムの作成から判定結果の表示までの流れの模式図を図 3 に示す。

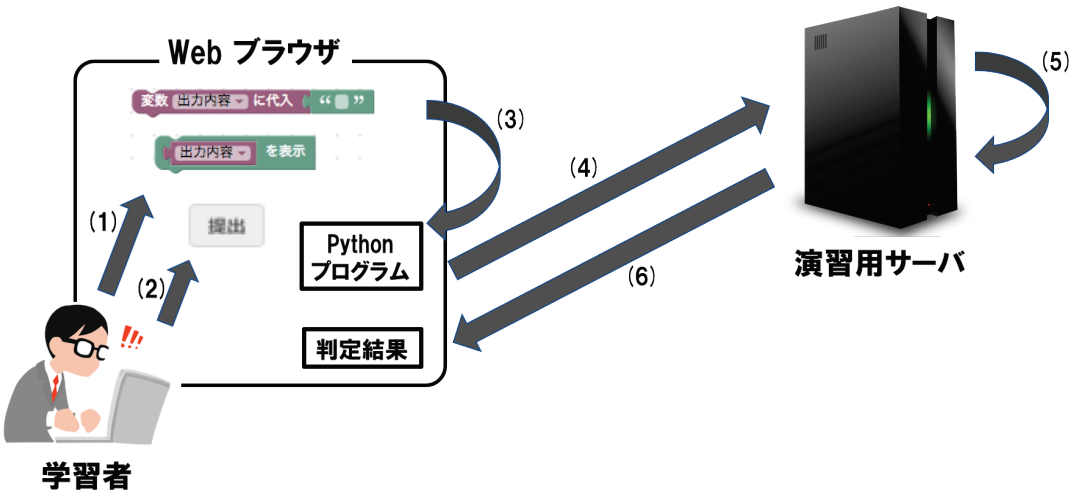

図 3　採点を受けるまでの流れ

4．演習の例

図 4 は「プログラミング入門」で実際に学習者に課されている演習課題の実行例である。この課題で学習者はプログラムに入力する文字列が回文であるか否か、すなわち逆順にした文字列が元の文字列と同一であるかを判定するための記述が求められる。「たけやぶやけた」を入力すると「回文です」と出力されるのが正しく、「たけやぶ」を入力すると「普通の文」と出力されるのが正しい。この題意を満たすように、学習者は画面上のブロックを組み合わせてプログラムを作成する。

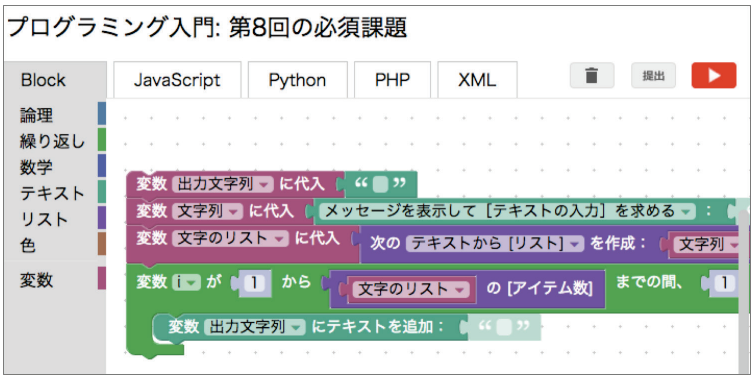

図４　演習課題の例

学習者がプログラムの記述を終えて「提出」ボタンを押すと、オンラインジャッジによる判定結果が図５のように表示される。図５では「さかさま」「まさかさかさま」「ひるめしたのしめる」「ひるめしたのしめるひ」等の文字列がプログラム実行時に入力され、それらに対する出力結果が正しかったかどうかが学習者に示されている。合格の場合はここで演習を終え、不合格の場合は併記されるヒントをもとにプログラムを修正して再度提出すればよい。

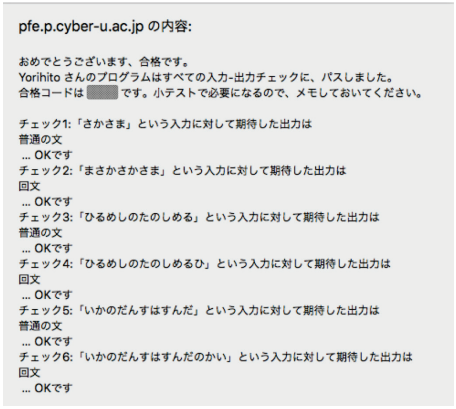

図５　演習合格時の画面

学習者のプログラムは演習サーバ内のデータベースに格納されるため、学習者が自力で合格にたどり着けず講師に質問した際には、講師は学習者のプログラムを見て画面上のブロックの状態を再現することができる。

５．おわりに

本稿ではフルオンライン大学に適したプログラミング導入教育の演習環境として Blockly および Sharif-Judge による結合システムを提案した。演習環境のもとで学習者は

Blockly による簡便な操作性とオンラインジャッジによる自動採点の利便性を享受でき、いずれも時間・場所の制約を受けることがない。また Blockly と Sharif-Judge は Web ブラウザを介して透過的に提供されるため、学習者はブロック型教材とオンラインジャッジという 2 つのシステムの操作方法を別々に習得する必要がない。

本学では演習環境を「プログラミング入門」の全ての履修者に提供し、その総数は 2016 年度春学期から 2017 年度秋学期まで、約 1000 人に達している。また演習環境では学習者が作成したプログラムの履歴を合否、タイムスタンプと共に記録している。履歴情報に基づいた学習状況の分類や個別指導の効率的な支援が、演習環境における今後の課題である。

注および参考文献

1) N. Fraser: "Google blockly - a visual programming editor", URL: http://code.google.com/p/blockly, accessed Dec. 2017.

2) M. Resnick, et al.: "Scratch: programming for all", Commun. ACM 52, 11, pp.60-67 (2009)

3) 原田 康徳: “子供向けビジュアル言語 Viscuit とそのインタフェース”, 情報処理学会研究報告, 114(HI-116), pp. 41-48 (2005)

4) 田中 頼人, 川原 洋: “フルオンライン大学におけるプログラミング導入教育の実践”, 第 42 回教育システム情報学会全国大会, pp. 41-42 (2017)

5) 松永 賢次: “導入プログラミング教育におけるオンラインジャッジシステムの活用の試み”, 情報科学研究(31), pp. 25-41 (2010)

6) 長尾 和彦, 古谷 勇樹: “JUnit に対応したオンラインジャッジシステムの開発”, 教育システム情報学会全国大会, pp. 255-256 (2015)

7) M. Naderi: "Sharif-Judge: A free and open source online judge system for programming courses", URL: https://github.com/mjnaderi/Sharif-Judge/, accessed Dec. 2017.