

スマートフォン向け 日本語プログラミング学習アプリケーションの開発

大久保遥花・宍戸真
東京電機大学大学院情報環境学研究科
{20jkm05}@ms.dendai.ac.jp, {shishido}@mail.dendai.ac.jp

1 はじめに

近年,プログラミング教育が重要視され,日本では,2020 年度から小学校においてもプログラミング教育が導入されている[1].文部科学省の狙いは,「知識及び技能」,「思考力,判断力,表現力等」,「学びに向かう力,人間性等」のような資質・能力を育成することである.手引書では,プログラミング教育において,身近な生活でコンピュータが活用されていることや,問題の解決には必要な手順があることに気付くことで「知識及び技能」を育成し,発達の段階に即して,プログラミング的思考を育成することで「思考力,判断力,表現力等」を育成し,コンピュータの働きを,よりよい人生や社会づくりに生かそうとする態度を涵養することで「学びに向かう力,人間性等」を育成するとされている.

このような現状の中,プログラミング学習を支援するシステムは有益であると言える.本研究の目的は,多くの人にプログラミングの面白さを実感してもらい,人々のプログラミング学習を支援する学習教材を開発することである.

2 先行研究

2.1 ビジュアル言語とテキスト言語の違い



図 1 Scratch の記述イメージ

小学校教育に適した教育用プログラミング言語の提案の論文[2]では,テキスト型言語,フローチャート型言語,ルールベース型言語,ブロック型言語のようなプログラミング言語について,それぞれの利点と欠点を述べている.その中でも,教育用言語として特に使用されることが多いブロック型言語について,問題と考えられる点がいくつか挙げられている.この中で挙げられているブロック型言語のテキスト型言語とのギャップという項目に,共感する部分があった.ブロック型言語は,用意されているブロックを組み立ててプログラムを組むので,コードを一から入力するテキスト型言語に比べて,入力の手間が省け簡単にプログラムを組むことができる.また,文法のエラーを防ぐことができるという利点があるが,一方で,そのプログラムの組み立てる手法は,テキスト型言語の手法と大きく異なる.図 1 は,教育現場でよく使用されているブロック型言語である Scratch の画面の例である.ブロック型言語のようなビジュアル言語は,プログラミング的思考力を培う教材ではあるが,見た目やプログラミングの手法がテキスト型言語と乖離しているため,汎用的なテキスト型のプログラミング言語の学習にスムーズに移行することは難しいだろう.そこで本研究は,汎用的なテキスト型プログラミング言語の学習への移行を円滑にすることを目標とする.

3 提案システム

本節では,提案するシステムの特徴や機能を説明する.

3.1 日本語によるプログラミング

コードの記述・エラーメッセージは基本的に日本語を用いる.その理由は,プログラミング初学者にとって,母国語でない言語による記述は学習を進める上で障壁になる場合が多いからである.日本語によって内容を理解しやすくすることで,初学者が挫折

しないようにすることが狙いである。

3.2 スマホで使えるアプリケーション

現在,日本はスマートフォン,PC,タブレット端末などの電子機器が広く普及しているが,普段コンピュータ機器をあまり使用しない人も含めた個人での所有率が最も高いデバイスはスマートフォンであると考えられる.本稿で提案するシステムは,PCを持っていない人でも気軽にプログラミングを学習して欲しいという狙いがあるため,多くの人が利用しやすいスマートフォンを使って学習ができるようにする。

3.3 汎用的プログラミング言語に近い文法

3.1 で説明したように,日本語を用いた分かりやすい記述を行えるようにすることを目指す,一方で2.1 に示したように,本システムは,汎用的なプログラミング言語の学習への移行を円滑にすることが目標であるため,日本語の文法に片寄りすぎて汎用的なプログラミングの文法と大きく異なってしまうことは避けたい.そこで,一般的に大学の講義に用いられる場合が多いC言語に近い文法で学習できるようにする。

4 設計・開発

4.1 開発環境

本研究では,Androidアプリを開発する.開発ツールは Android Studio で,言語は Java を使用する.レイアウト画面設計には XML が用いられる。

4.2 構成

本アプリケーションの構成は,全 5 章から成る.それぞれの章の内容は,表 1 にまとめる。

表 1 章ごとの学習内容

章番号	学習内容
1	基本(文字列を表示・変数への代入)
2	演算子
3	制御文
4	配列
5	関数

それぞれの章で,学習内容に関する説明を読み,それに関する演習問題を解くという流れに沿って学習を進める。

4.3 演習問題の例

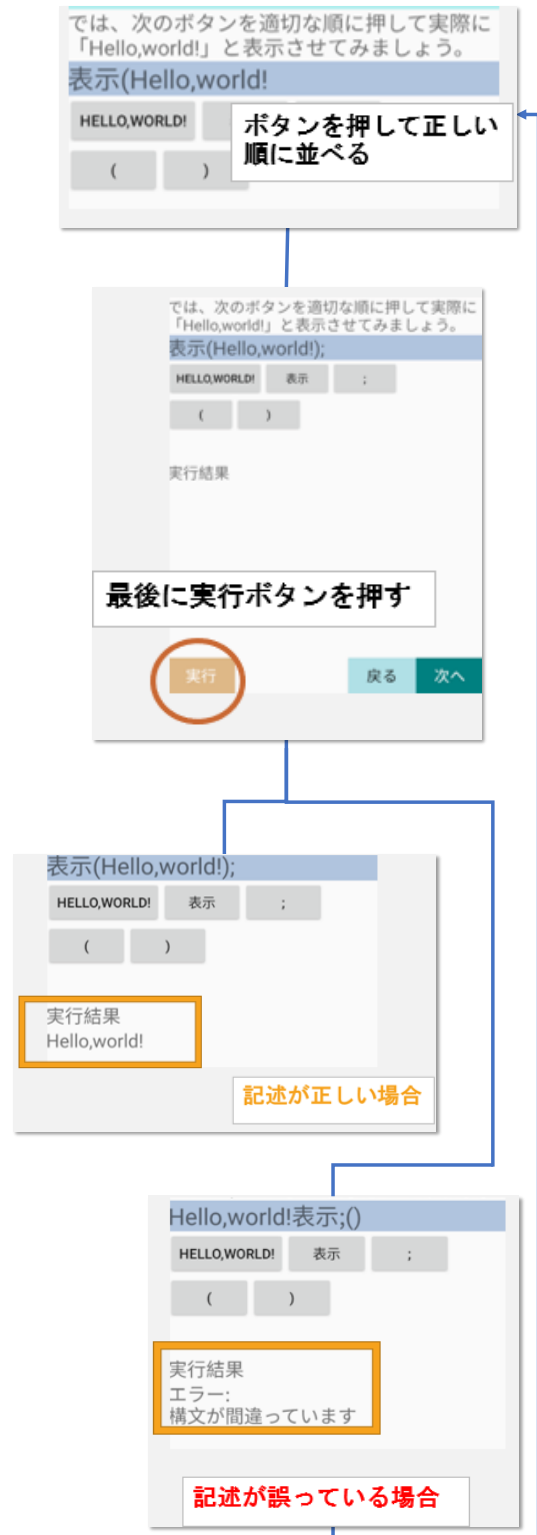


図 2 プログラム実行のフローチャート

図 2 は,全体の動作の流れを示した図である.まず,入力画面で,パーツのボタンを正しい順番で押して記述をする.ボタンを押し終えたら,実行ボタンを押す.もし記述内容が正しければ,実行結果が正しく表示される.記述が正しくない場合は,エラーメッセージが表示され,また最初から入力をすることになる.

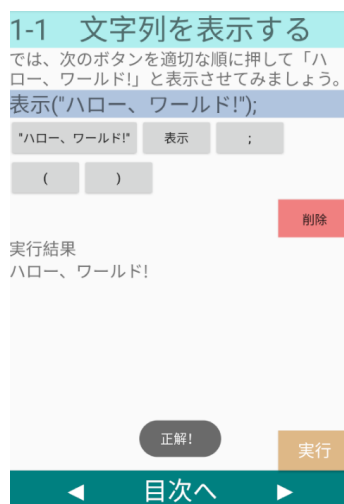


図 3 文字列を表示(正解)

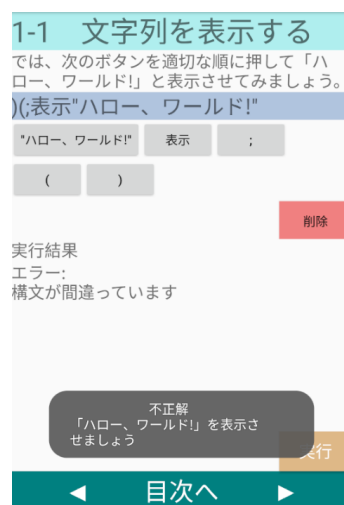


図 4 文字列を表示(不正解)

図 3,4 は文字列を表示させる演習問題の例である.「実行すると」ハロー,ワールド!」と表示される」という画面の例で, 図 3 は成功した場合, 図 4 はエラーが発生する場合の画面である.それぞれ,画面下部に正解または不正解のトーストが表示される.

次に,変数の代入に関する問題について説明する.この問題は,変数 a に「10」,変数 b に「や」,変数 c に「ありがとう」をそれぞれ代入し,変数の中身を出力するという問題である.3 つの変数を出力することが出来ると正解を示すトーストが表示される.

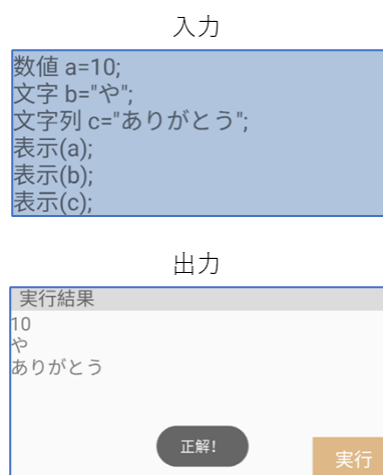


図 5 変数の問題(正解)

図 5 は,変数 a,b,c にそれぞれ数値,文字,文字列を代入し,3 つの変数を表示させるプログラムの記述と実行結果である.ボタンを押して上図のように記述し,実行ボタンを押すと下図のような結果が表示され,画面下部に正解のトーストが表示される.

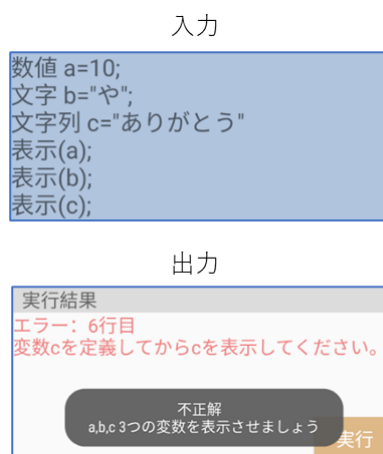


図 6 変数の問題(エラー)

図 6 は図 5 のプログラムの 3 行目「文字列 c=「ありがとう」;」の文の最後のセミコロンを記述しなかった場合の入力と実行結果である.下図のような結果が表示され,画面下部に不正解のトーストが表示される.

次に,算術演算子に関する問題について説明する.この問題は,「3×8」,「48÷3」,「56を11で割った余り」を出力するという問題である.3 つの演算結果を出力することが出来ると正解を示すトーストが表示される.

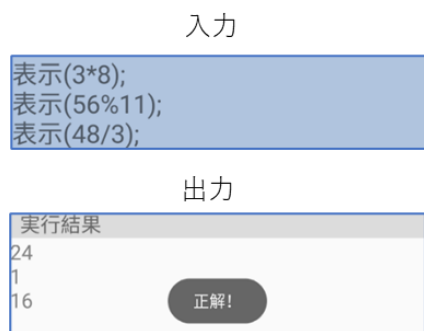


図 7 算術演算子の問題(正解)

図 7 は, " 3×8 ", " 56 を 11 で割った余り", " $48 \div 3$ " の演算結果を表示させるプログラムの記述と実行結果である。画面上のボタンを押して上図のように記述し, 実行ボタンを押すと下図のような結果が表示され, 画面下部に正解のトーストが表示される。

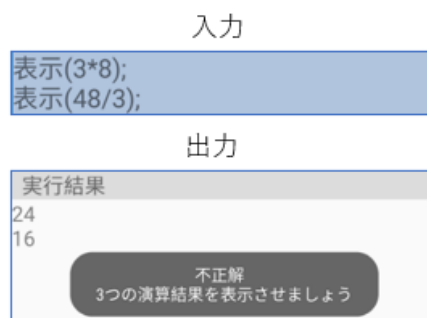


図 8 算術演算子の問題(不正解)

図 8 は, " 3×8 ", " $48 \div 3$ " の計算結果を表示させるプログラムの記述と実行結果である。この場合, 3 つの演算結果を表示させることが出来ていないため, 画面下部に不正解のトーストが表示される。

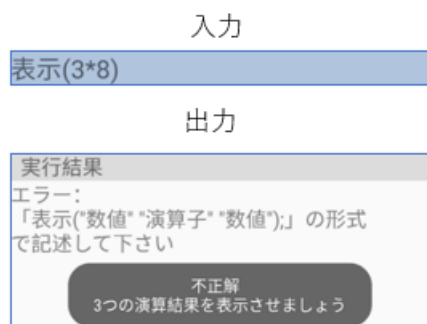


図 9 算術演算子の問題(エラー)

図 9 は, " 3×8 " の計算結果を表示させる記述が, 末

尾の; が抜けているためエラーが引き起こされる例である。この場合も図 8 と同じく画面下部に不正解のトーストが表示される。

4.4 演習問題の実装

本節では, 4.3 で説明した演習問題について, どのように実装しているか説明する。

まず, 図 3.4 の文字列を表示させる問題は, 正解の文字列を用意し, 入力欄に記述された内容が正解の文字列と等しければハローワールドと表示させ, 等しくなければエラーが発生するようになっている。

次に, 図 5.6 の変数に関する問題について説明する。最初に入力欄に記述された文字列を改行ごとに `split` メソッドを用いて分割し, 配列にする。そして `for` ループで「表示(変数名);」文が記述されている箇所を探す。「表示(変数名);」文を見つけたら, それよりも前に該当する変数を定義する文が記述されている場合はその変数の値を出力させ, 定義する文が記述されていない場合はエラーメッセージを表示させる。

最後に, 図 7.8, 9 の算術演算子の問題について説明する。入力が「表示(計算式);」の形式に当てはまっているかどうかを正規表現で判断する。マッチしていれば括弧内の計算式を取り出し, 数値と演算子に分け, 演算結果を返すようになっている。

5 まとめ

本稿では, 日本語を用いたプログラミング初学者向けのスマートフォン向けアプリケーションを提案した。今後は, 表 1 に示した内容を学べるように教材を豊富にし, 様々なタイプの演習問題を取り入れる。また, UI/UX デザインを改良し, 使いやすいレイアウトにする。教材を作成した後は, 教材を実際に使用してもらい評価を受ける実験を行っていく。

参考文献

1. 文部科学省. 小学校プログラミング教育の手引 (第二版), 2018.
2. 久保文乃, 久野靖. 小学校教育に適した教育用プログラミング言語の提案, 2019.