**２０２１　年度　　卒業論文**

卒業論文題目

|  |
| --- |
| ブロックプログラミングを用いた論理的思考と コーディングを身に着けるための学習環境 |

　指導教員　　　　　　鷹野　孝典　教授

神奈川工科大学　情報工学科

　　　　　学籍番号　　　　1821121

　　　　　学生氏名　　　　島岡慎也

　提出日　２０２１年　１２月２３日　指導教員　　印

　受理日　２０２１年　１２月２３日　情報工学科長　　印

# **論文要旨**

# 目次

[**論文要旨** 2](#_Toc89051929)

[目次 1](#_Toc89051930)

[**1.** **まえがき** 1](#_Toc89051931)

[1.1 はじめに 1](#_Toc89051932)

[1.2 本論文の構成 1](#_Toc89051933)

[2. 関連研究 2](#_Toc89051934)

[2.1 2](#_Toc89051936)

[**3.** **提案システム** 3](#_Toc89051937)

[3.1 提案方式の概要 3](#_Toc89051939)

[3.2 提案方式の構成 3](#_Toc89051940)

[3.2.1. 問題を自動生成する 3](#_Toc89051950)

[3.2.2. 問題の正誤を評価 3](#_Toc89051951)

[3.2.3. 整合性の評価 3](#_Toc89051952)

[3.3 提案方式を構成する機能 3](#_Toc89051953)

[**4.** **実装システム** 4](#_Toc89051954)

[4.1 実装システムの概要 4](#_Toc89051956)

[4.2 実装システムの実行環境 4](#_Toc89051957)

[4.3 実装システムの機能説明 4](#_Toc89051958)

[4.3.1. Visual Program Language 4](#_Toc89051963)

[4.3.2. Node.jsについて 5](#_Toc89051964)

[4.3.3. コードジェネレート機能 5](#_Toc89051965)

[4.3.4. コードの実行 5](#_Toc89051966)

[4.4 実装システムの穴あき問題の実装 5](#_Toc89051967)

[4.4.1. 穴埋め問題の生成 5](#_Toc89051969)

[4.5 実装システムの穴あき問題の評価の実装 5](#_Toc89051970)

[4.5.1. 穴埋め問題の解答の評価 5](#_Toc89051972)

[4.6 Blocklyの変更点 7](#_Toc89051973)

[**5.** **実験** 8](#_Toc89051974)

[5.1 実験目的 8](#_Toc89051976)

[5.2 実験環境 8](#_Toc89051977)

[5.3 実験 8](#_Toc89051978)

[5.4 実験の考察 8](#_Toc89051979)

[4.5.2. 実験の感想一覧 8](#_Toc89051980)

[**6.** **むすび** 9](#_Toc89051981)

[**謝辞** 10](#_Toc89051982)

[**参考文献** 11](#_Toc89051983)

[その他 11](#_Toc89051984)

# **まえがき**

## はじめに

2020年以降小学校，中学校，高等学校にてプログラミング教育の必修が全面実施される．

文部科学省の発表によると，プログラミング教育を必修とする目的は，現代社会で普遍的に求められる力としての論理的思考などを育むことである．論理的思考とは，「思考や論証の組み立て，思考の妥当性が保証される法則や形式」に則って思考を組み立てることとする．学習者が教科書を読み進めるだけではなく，プログラミングに実際に触れることで，論理的思考を身に着けられる環境を作成したいと考えた．ブロックプログラミングを用いることで，学習の初期段階でプログラム言語の構文への理解が浅い状態でもコードの実行が可能である．Blocklyのコードジェネレート機能を用いることで，プログラムの構文への理解につながる．論理的思考がすでに培われていることを前提として問題が提示される教育環境となっている．学習者が論理的な思考をもって問題文を読み取ることができる必要がある．プログラミング言語における構文を理解するには，コーディング経験も重要となる．Blocklyによって生成されたコードを見るだけではなく，実際に学習者がコーディングする環境でないと，プログラミング言語の構文への理解の定着にはつながらない．

課題

穴あき問題を自動で生成する際に，正答だと想定した選択肢以外にも実行できる選択肢が存在する場合などに，それを正解だと判断せず，既定の選択肢以外が不正解だと認識してしまう。

提案

　問題としての正解以外でも，既定の選択肢ではないが，実行結果から問題の解答として正しい，問題の解答としては正しくないがコードとしては実行できるなど，問題の正誤だけでなくコードとしての整合性も考慮した採点をする．

## 本論文の構成

　本論文は全６章で構成されている。本章では研究の背景や動機，目的について述べる．第２章では関連研究について述べる．第３章では本研究の提案方式とその特徴述べる。第４章では提案方式を構築した本システムの設計及び実験システムの実装について述べる．第５章では実験を行う。実験の概要と，比較実験後の結果を述べる．第６章では結論と今後の展望について述べる．

# 関連研究



## 

# **提案システム**

　本研究では，穴あき問題の解答の規定の正答だけでなく、正答以外の選択の際にも「構文上の正しさ」と「問題文との整合性」の２点の評価をすることにより，不正解となった場合の学習効率の向上に焦点をあてた学習支援を提案する．なお，提案方式の概要を3.1節，提案方式の構成を3.2節，提案方式を構成する機能を3.3節に示す．



## 提案方式の概要

提案方式は，ブロックプログラミングとコードの比較をする部分，穴埋め問題を利用し複数のコードを見比べる部分，問題と解答から，構文上正しいかどうか，問題文と比べ正しいかどうかを評価する部分に分かれている．

## 提案方式の構成



### 問題を自動生成する

事前に準備されたコードを全て単語毎に区切り，別に用意した規定のワードと比較し，マッチングした箇所を穴あき問題へと変換する．この時，穴あき問題の問題数を変更することで難易度の調整をする．

### 問題の正誤を評価

　3.2.1章にて作成した穴あき問題にて，選択肢中に正答かどうかを判別するオプションを設定しておき，それにより問題の正誤を判定する．

### 整合性の評価

問題の規定の正解以外でも、実行結果から問題の解答として正しい選択肢、あるいは問題の解答としては正しくないが，コードとしては実行できる状態となる選択肢など、問題の正誤だけでなくコードとしての整合性も考慮した評価をする。前者の評価は、事前に用意した結果と、解答から作成される実行結果との比較によって評価をすることが可能である。後者の評価は、実際にコードを実行した結果、事前に用意した正答の結果とは異なるが、エラーが発生しないまま終了した場合などにどのような間違いだったのかを評価する。

## 提案方式を構成する機能

・問題自動生成

・正誤評価

・整合性評価

# **実装システム**

　本研究では，穴あき問題の解答の規定の正誤だけでなく、規定の正答以外の選択の際にも「構文上の正しさを評価する機能」と「問題文との整合性を評価する機能」の２点により，不正解となった場合の学習効率の向上に焦点をあてた学習支援のための機能を実装した．なお，実装システムの概要を4.1節，実装システムの実行環境を4.2節，に示す．



## 実装システムの概要

　実装システムの構築環境を表4.1に示す．

表 4.1 実装システムの構築環境

|  |  |  |
| --- | --- | --- |
| 利用用途 | 名称 | バージョン・型番 |
| OS | Windows 10 |  |
| 開発言語 | Javascript |  |
| パッケージ管理 | npm |  |
| VPL | Blockly |  |

## 実装システムの実行環境

・本研究のプログラムのダウンロード

以下のgithubのリンクよりprojectの内容をすべてダウンロード．

|  |
| --- |
| https://github.com/kait-takanolab/1821121-shimaoka-thesis/tree/main/project |

・node.jsのインストール（Windowsの場合？）

以下のリンクよりNode.jsをインストール．

　Node.jsをインストールすることで、Node.jsというサーバ上でJavascriptを実行ができる．また，npmも同時に利用が可能になる．

|  |
| --- |
| https://nodejs.org/ja/ |

・サーバの実行コマンド

　上記でダウンロードした本研究のprojectの位置で下記のコマンドを実行することでサーバが起動される．その後loaclhost:3000にアクセス．

|  |
| --- |
| npm start |

## 実装システムの機能説明



### Visual Program Language

ビジュアルプログラミング言語（VPL）とは，プログラムを視覚的なオブジェクトの組み合わせによって作成するプログラミング言語である．ブロックプログラミング，グラフィカルプログラミング言語とも言う．VPLとは総称であり，どのように視覚的に示すかにはいろんな種類があるが，本研究ではBlocklyを利用する．

### Node.jsについて

　Node.jsとはサーバ上で動作するJavascript環境．ほとんどのJavascriptと異なり，ウェブブラウザ上でなく，サーバサイドでの動作となる．本研究ではサーバ上での実行と，npmの利用のためにNode.jsを利用する．

npmとはパッケージ管理システムの一種で，Node package managerの意である．Node.jsを用いたシステムの管理のために作成され，Node.jsのツールやパッケージのインストールや管理はもちろん，それらパッケージを利用することも可能．

　また，Node.jsのWebアプリケーションフレームワークであるexpress.jsを用いてjson環境を構築した．

### コードジェネレート機能

generateJS(),indent(),

コードジェネレート機能とは，Blocklyによって提供されるモジュールの一つで，組み合わせたブロックからコードを生成する機能を提供している．Blockly公式より提供されるコードジェネレート機能はJavaScript，Python，PHP，Lua，Dartの５つのプログラミング言語に対応したものが存在し，関連研究にはC言語など他の言語に対応させたものも存在する．本研究ではJavascriptに関連した穴あき問題の生成・評価をしている．

### コードの実行

execution(),kaitouran(),jsruncode(),jsrunque(),jsrunout(),jsruntst()

## 実装システムの穴あき問題の実装



### 穴埋め問題の生成

match(), easy(),normal(),hard(),getRandomInt(),

・本研究では問題文となるコードから穴埋め問題を自動生成する．

・既定のワードとマッチした箇所を問題が生成される個所としてリストアップ

・リストされた場所を選択肢のためのセレクトボックスに置換

・セレクトボックスの内容を問題とし、オプションとして正答のものには正解としてans属性を付与

## 実装システムの穴あき問題の評価の実装



### 穴埋め問題の解答の評価

answer(),check(),check2(),

・既定の正答の個数をカウントし、何問正解したかを評価

・全問題を規定通りに正解した場合は、満点と評価

・既定の正答以外の場合に、実際に選択されたコードでの実行結果からエラーが発生しない場合などに、構文上は正しいことを評価

・既定の正答以外の場合に、問題の想定とは違うが動作が一致する場合には評価をする

・実行して結果を目で確認する

## Blocklyの変更点

・text\_printを変更

・for文を変更

# **実験**



## 実験目的

## 実験環境

## 実験

## 実験の考察

### 実験の感想一覧

# **むすび**

# **謝辞**

　本研究を行うにあたり，多くの助言をくださった鷹野孝典教授に心から感謝いたします．

研究を一緒に進めながら様々な意見を出し合い，本研究の方向性の確立にご協力くださった鷹野研究室の４年生のメンバーに深く感謝いたします．

# **参考文献**

# その他

* ソースコード等を記述するときは，表内に挿入し，行間を「固定値：12pt」に設定する．行間を制する者はwordを制する．
* 研究でやったことはすべて，基本的に「本研究では」と称する
* ファイル名は，「1821~-name-thasis.docx」で統一する．
* 目次では、謝辞や参考文献のページ番号は載せない
* 要旨には，背景，課題，提案，実験で確認したことを書く
* 図表番号とラベルの間は半角スペースを入れる
* 節と節の間は改行する．章と章は改ページする
* 本文は日本語：MS明朝　　英語：Cenutty
* ただし数式は「挿入→数式」から入力し，本文中で数式を用いる場合はフォントの形が似ているTimes New Romanを使う．
* 付録にあるプログラムの説明がしたいとき本文の方に，例えば，“（付録３参照）”，“……を付録３に示す．”等と参 照先を記しておく．
* 章の終わりに改ページを行う．
* 改ページはCtrl+Enter
* 今後の課題ではなく展望．修正点とか書かない