

コンピュータサイエンス教育における テキストプログラミング支援システムの提案

Proposal of text programming support system in computer science education

小早川 祐一郎[†] 赤澤 紀子[†]

Yuichiro Kobayakawa Noriko Akazawa

1. はじめに

1.1 テキストプログラミング学習の必要性

文部科学省 [1] が 2017 年度に公示した新学習指導要領によると、2020 年から小学校でプログラミング教育が必修化される。また、中学校、高等学校でもプログラミングおよびコンピュータサイエンスに関する内容がより充実したものとなり、2021 年、2022 年と順次新たな学習指導要領 [2][3] に沿った授業が行われることとなる。

中学校の学習指導要領のプログラミングに注目する。現行平成 20 年告示の中学校技術・家庭科の技術分野では、現行のプログラムによる計測制御がある。改訂平成 29 年告示の指導要領では、計測・制御のプログラミングに加えて、ネットワークを利用した双方向性のあるプログラミングが追加されている。この、ネットワークを利用したプログラミングは、現行の高等学校の情報科で学習する内容である。

続いて高等学校に着目すると、問題の発見・解決に向けて情報と情報技術を適切かつ効果的に活用し、情報社会に主体的に参画するための資質・能力を育成することを目標として、必修科目である「情報Ⅰ」が、それらの発展・応用として選択科目である「情報Ⅱ」が新設される。現行の高等学校における情報科目は、プログラミングを教えずともよい「社会と情報」とプログラミングを教える「情報の科学」の 2 科目に分かれており、このうち 1 科目を選ぶ選択必修になっている。多くの高等学校では、前者を選択している。しかし、今回の指導要領の改訂により、必修科目である「情報Ⅰ」にプログラミングが組み込まれ、生徒は必ずプログラミングを行うことになるため、より一層プログラミングの必要性が高まっている。

コンピュータサイエンス教育の一環でプログラミングが行われる場合、Scratch[4] や Viscuit[5]、micro:bit[6] のようなビジュアル型の言語や JavaScript や VBA といったテキスト型の言語が用いられる。コンピュータサイエンス教育において、どちらが適切かどうかについての意見は様々あるが、改訂された高校の次期指導要領の解説(情報編)[7]を見ると、オープンデータを用いた数値

計算を行い、その過程でアルゴリズムやプログラミングの学習および API やライブラリの使用によるプログラムの作成が想定されている。また、久野ら [8] もプログラミング言語を用いたソフトウェア開発を実践するとともに、計算量などといった基本概念とモデル化・シミュレーションとそれらに基づく問題解決を高等学校で扱うことを提案している。このことから、改訂後の高等学校の情報科目では、テキスト型のプログラミング言語が用いられると考えられる。

1.2 情報教育の現状

現段階においても情報教育に力を入れている高等学校では、プログラミング学習を含む「情報の科学」で Scratch や micro:bit といったビジュアル型言語に始まり、JavaScript といったテキスト型言語を用いる授業を行っている。中には、改訂後の「情報Ⅰ」を意識したカリキュラムで授業を行っている学校も実際にあった。

しかし、実情として、文部科学省 [9] の 2015 年度の調査結果によると、教科「情報」における各科目の履修率は 8 割が「社会と情報」であり、「情報の科学」は 2 割にとどまっている。加えて、同年度の文部科学省の調査によると、情報科の担当教員の 3 割が免許外であり、5 割が他教科と兼任していることが分かった。現在よりも高い水準の情報教育が求められているのにも関わらず、情報専任の教員が少ないのである。こういった現状に対し、中山ら [10] も、情報科では免許外教科担任が特例という枠を超えて多用されていること、さらに、それが他教科に比べて情報科が突出していることを懸念している。

そのような状況下で高等学校でテキストプログラミングを実施することは容易ではないと考える。

1.3 テキストプログラミング支援

テキスト型の言語を学習する際、学習者は使用言語の文法や命令といった御作法をある程度把握していなければならず、分からなければ書籍で調べたりネットで検索する動作が必要となる。初学者の場合、そもそも調べことをしない人もいる。実際に、東京都の高等学校の情報科の教員にヒアリングを行ったところ、エラーが出力されたらそのまま手を止め、教員が手助けしてくれるのを待っていた事例があった。もちろん、学習者以上に教える立場にある教員は上記のようなプログラミングの御作

[†] 国立大学法人電気通信大学, The University of Electro-Communications

法やエラー対処方法を理解しておく必要があるが、1.2節で述べた問題がある以上、情報学に精通している教員でなければ、教員も学生と一緒にその場で調べるといった動作が生じてしまう。

そこで本研究では、以上のようなテキストプログラミングを行ううえで障壁となりうる問題を緩和する機能を提案する。提案システムでは、キーワードをエディタ上で入力することで、そのキーワードに関連した命令を表示し、補完可能とする。エラーに関しては、吐かれたエラーが理解できない、そもそもエラーメッセージを読まないという問題があると考え、間違いであろう箇所を指摘し、なおかつエラーメッセージを分かりやすい表現で提供する。

これら機能により、次に挙げるものが達成されることを期待する。

- プログラミングの御作法やエラー解説の時間を少なくすることで、コンピュータサイエンス教育の本質の学習に割く時間を増やす
- 適宜調べることによって積み重なるストレスを減らし、やる気を持続させる
- 副次的な効果として、指導者の負担を減らす

本システムでは Python 言語をサポート対象とする。文部科学省 [11] の掲示した教員研修用教材 (2019 年 7 月時点では閲覧不可) では、Python を用いており、ヒアリングを行った高校でも試験的に Python を使った授業を展開していた。また、Guo[12] の調査によると、アメリカの主要大学 39 校のうち 69% の大学がコンピュータサイエンス入門コースで Python を採用していた。このことから、プログラミング教育の入門におけるテキスト型のプログラミング言語として Python は適当であり、今現在日本でも Python を使う流れになっていると言えるため、これを採用する。

2. 関連研究

2.1 キーワード入力からの補完

2.1.1 コメントイン: コメントを先に書くことによる新形態の API リファレンス

薄羽ら [13] は、使うべき API の仕様が分からないユーザーを対象とし、先に「後にコメントとなる自然言語」を入力すると、それに関連した「そのプログラミング言語でできること」の一覧を推薦し、選択することで命令が記述されるシステムを提案した。ほぼ全員がプログラミング初心者である高校生に実験を行ったところ、開発したシステムを用いることで、全員が課題をクリアし、システムの有効性を示した。

2.1.2 Example-Centric Programming: Integrating Web Search into the Development Environment

Brandt ら [14] は、ユーザーがサンプルコードを見つけるのに役立つ Web 検索インターフェース Blueprint を提案した。Blueprint は、入力されたキーワードから Web 上のサンプルコードと説明文を検索・取得し、それをユーザーのプログラムに追記できるというシステムである。これにより、被験者は標準の Web ブラウザを使用するよりも早くサンプルコードを見つけ、高速にコードを記述することができたと著者は結論づけている。

2.2 エラーメッセージの改善

2.2.1 Identifying and Correcting Java Programming Errors for Introductory Computer Science Students

Hristova ら [15] は、Java プログラミングはコンパイルエラーのメッセージが複雑なこともあり、初心者の学生にとっては大変困難であると述べた。著者らはそれら問題を解決すべく、Expresso という Java プログラミング用の教育ツールの作成を行った。

事前準備として大学教授や教育助手およびコンピュータサイエンスの学生を対象に Java プログラミングにおけるエラーの調査を行い、リスト化した。

それらリストをもとに著者らは、コンパイラメッセージを入門生が理解しやすいコメントに変換するよう設計されたシステムをプリプロセッサとして実装した。

3. 提案する支援機能

本研究では、以下 2 つの機能を提案する。

1. キーワードからの推測および補完
2. エラーメッセージの簡素化

これら機能は、Python 言語を対象とし、Visual Studio Code[16](図 1) と言われるコードエディタに拡張機能として組み込む形で提供する。Visual Studio Code とは Microsoft が開発した、Windows、macOS、Linux 上で動作する無償のコードエディタである。日本語を含む多くの言語に対応しており、シンタックスハイライト、コード補完、デバッグなどの機能を持つ。さらに、さまざまな拡張機能が公開されており、プラグインとして簡単にインストールすることが可能なため、ユーザー各々は自分に合ったカスタマイズをすることができる。

機能 1: キーワードからの推測および補完

通常のコード補完の場合、入力と部分一致した関数名やメソッド名が推測され候補として表示される。ただし、

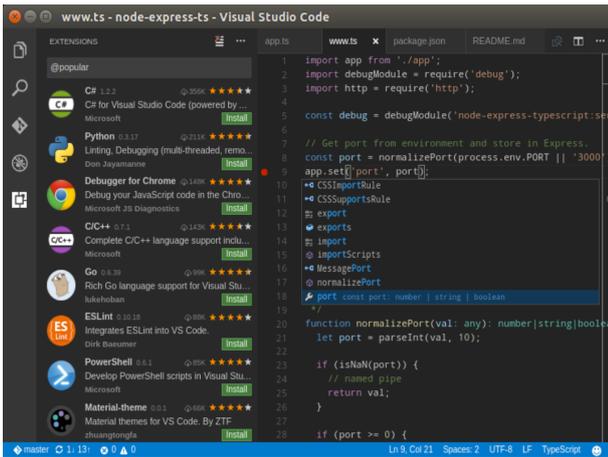


図 1: Visual Studio Code

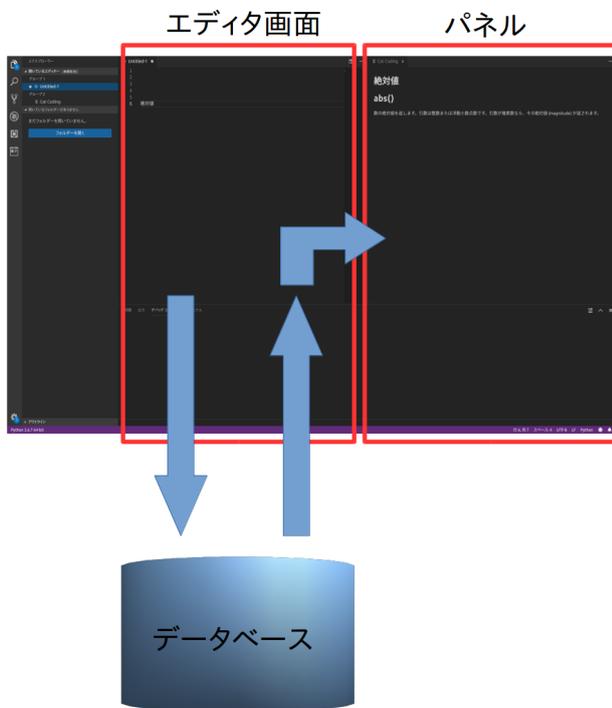


図 2: システムの構成



図 3: キーワード推測の例

これはある程度その関数名やメソッド名が思い浮かんでいる時にしか効果を発揮しない。提案するシステムでは、キーワードを入力するとそれに関連するような命令や関数名をデータベースを参照し、パネル上に候補として表示する(図2)。

一例として、「絶対値」というキーワードを入力すると絶対値を返す関数 `abs(x)` とその説明が表示されるといふ具合である(図3)。それから関数 `abs(x)` が補完されるという機能を目指す。

エディタの画面で命令および関数の確認ができるため、エディタからブラウザに画面を切り替えるといった画面移動が不要となる。これにより、分からない関数があっても書籍で調べたりネットで検索かける手間が省け、作業効率が高るとともに学習者のフラストレーションの蓄積を抑えることができると考える。また、学習者の学習意欲が低下するのを防ぐことも期待できる。

機能 2: エラーメッセージの簡素化

1章の1.3節で実例として挙げた通り、初学者はエラーが吐かれると手が止まってしまう傾向にある。原因として次のようなことが考えられる。

- エラーメッセージが英語であり、抵抗を感じる
- 専門用語が出てくる
- エラーメッセージが色々な情報と一緒に出力されるため、着目すべきポイントが分からない

そこで、エラーが発生している箇所を指摘し、かつ分かりやすい表現で提供してくれる機能を追加する。具体的な機能は次の通りである。

- エラーメッセージを日本語化する
- 専門用語を用いず、一般的な言葉で表現する
- 余計な情報を削り、端的に表す

これにより、学習者のエラーに対する抵抗を小さくすることができる。結果的に機能1同様、学習者のストレスを緩和し、学習意欲の低下を防ぐことができると著者らは考えている。

4. 実装方針

Visual Studio Code の公式によると、拡張機能は TypeScript での開発が推奨されているため、それを用いる。

機能1で参照する、キーワードおよびそれらに対応する関数名やその説明などの情報を持つデータベースの実装には SQLite[17] を使用する。SQLite とは、パブリッ

クドメインの軽量なりレーショナルデータベースマネジメントシステムである。関数名や説明といったフィールドから構成される各レコードは、複数のキーワードと対応させる。

例えば、「sum()」という関数には、「和」、「総和」、「合計」といった複数のキーワードと結びつける。これにより、学習者各々の表現の違いによるシステムの動作の影響を少なくし、柔軟なキーワード推測・補完が可能となる。

データベースは関数名・説明のテーブルとキーワードのテーブルを2つ用意する。しかし、この場合1つの関数名に対し複数のキーワードを持ち、1つのキーワードに対し複数の関数名を持つという多対多の関係になる恐れが出てくる(図4)。通常リレーショナルデータベースシステムでは、2つのテーブル間に多対多のリレーションシップを直接設定することはできないため、中間テーブルを用意する(図5)。中間テーブルを用意することで1対多のリレーションシップにする。

参照先のデータベースに格納するデータは、Pythonの公式ドキュメント[18](図6)から取得する。

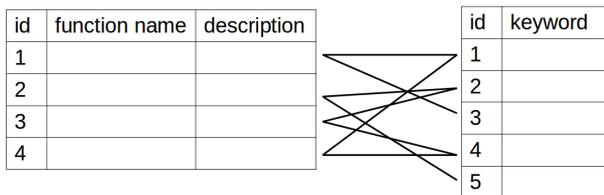


図 4: 関数名テーブルとキーワードテーブル

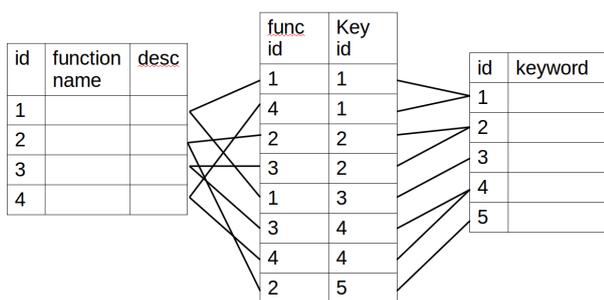


図 5: 中間テーブルを用意した場合

5. 評価について

評価実験として、Pythonによる簡単なプログラミング演習を行ってもらう。対象者は次の通りである。



図 6: Pythonの公式ドキュメント

- 高校生
- コンピュータサイエンス専攻以外もしくは入門したばかりの大学生

その際、提案するシステムを使用した場合と使用しなかった場合でのコーディングの進み具合をはかる。同時に提案システムの使用回数も計測する。

最後に被験者にアンケートを実施し、システムの使用感や必要性を評価してもらう。

6. おわりに

6.1 まとめ

本稿では、はじめに指導要領改訂から見えるテキストプログラミングの必要性を述べた。また、情報教育の現状と東京都の高等学校でのヒアリング結果を交え、テキストプログラミングを行ううえで起こりうる問題・課題を挙げるとともに、それらを緩和する支援システムを提案した。

具体的には、将来的に全員が学ぶことになる高校生もしくは彼らと同等の情報学の知識を持っているであろうコンピュータサイエンスが専攻でない大学生を対象とし、「キーワードからの推測・補完」と「エラーメッセージの簡素化」の2つの機能を持たせる。これら機能により、円滑なコーディングが可能となり、コンピュータサイエンスに対する学習意欲を保ち続けられることを期待する。

6.2 今後の展望

本稿で提案したシステムがコンピュータサイエンス教育において有用であるかを確かめるべく、まずはシステムの実装に取り掛かる。それと並行して実際にテキスト型プログラミングを授業内で扱っている教育現場を引き続き視察する。調査の中で新たな問題が発見された場合は、それらを解消する機能も反映させたいと考えている。

また、現段階ではデータベースに格納する関数等のデータは、Python の公式ドキュメントに記載されている組み込み関数を想定している。しかし、Python でのプログラミングとなると組み込み関数だけではできることが限られるため、モジュールのインポートが必要となってくるであろう。文部科学省が一時期公開した教員研修用教材 (2019 年 7 月時点では閲覧不可) でも requests モジュールや json モジュールをプログラム例の中で使用していた。そのため、教員研修用教材を含めたあらゆる教材に記載されているプログラム例を調査して、どこまでを網羅するかを決定する必要がある。

最後に、被験者を募り、提案システムを使って簡単な Python によるプログラミング演習を行ってもらい、システムの評価を行うことを考えている。具体的な実験内容や評価方法に関しても今後検討する必要がある。

参考文献

- [1] 文部科学省: 小学校学習指導要領 (平成 29 年告示)(オンライン), 入手先 <http://www.mext.go.jp/component/a_menu/education/micro_detail/_icsFiles/afieldfile/2019/03/18/1413522_001.pdf> (参照 2019-07-25).
- [2] 文部科学省: 中学校学習指導要領 (平成 29 年告示)(オンライン), 入手先 <http://www.mext.go.jp/component/a_menu/education/micro_detail/_icsFiles/afieldfile/2019/03/18/1413522_002.pdf> (参照 2019-07-25).
- [3] 文部科学省: 高等学校学習指導要領 (平成 30 年告示)(オンライン), 入手先 <http://www.mext.go.jp/component/a_menu/education/micro_detail/_icsFiles/afieldfile/2018/07/11/1384661_6_1_2.pdf> (参照 2019-07-25).
- [4] MIT Media Lab: Scratch(online), available from <<https://scratch.mit.edu/>> (accessed 2019-07-25).
- [5] 合同会社デジタルポケット: Viscuit(オンライン), 入手先 <<https://www.viscuit.com/>> (参照 2019-07-25).
- [6] Microsoft: micro:bit(オンライン), 入手先 <<https://microbit.org/ja/>> (参照 2019-07-25).
- [7] 文部科学省: [情報編] 高等学校学習指導要領解説 (平成 30 年告示)(オンライン), 入手先 <http://www.mext.go.jp/component/a_menu/education/micro_detail/_icsFiles/afieldfile/2019/03/28/1407073_11_1_1.pdf> (参照 2019-07-25).
- [8] 久野 靖, 和田 勉, 中山泰一: 初等中等段階を通じた情報教育の必要性和カリキュラム体系の提案, 情報処理学会論文誌「教育とコンピュータ」, Vol.1, No.3, pp.48-61 (2015).
- [9] 文部科学省: 情報教育に関連する資料 (平成 27 年告示)(オンライン), 入手先 <http://www.mext.go.jp/b_menu/shingi/chukyo/chukyo3/059/siryo/_icsFiles/afieldfile/2015/11/11/1363276_08_1.pdf> (参照 2019-07-25).
- [10] 中山泰一, 中野由章, 角田博保, 久野 靖, 鈴木 貢, 和田 勉, 萩谷昌己, 笈 捷彦: 高等学校情報科における教科担任の現状, 情報処理学会論文誌「教育とコンピュータ」, Vol.3, No.2, pp.41-51 (2017).
- [11] 文部科学省: 高等学校情報科「情報 I」教員研修用教材 (2019 年 7 月時点で閲覧不可), 入手先 <http://www.mext.go.jp/a_menu/shotou/zyouhou/detail/1416746.htm> (参照 2019-07-25).
- [12] Philip Guo: Python Is Now the Most Popular Introductory Teaching Language at Top .S. niversities(online), available from <<https://cacm.acm.org/blogs/blog-cacm/176450-python-is-now-the-most-popular-introductory-teaching-language-at-top-s-niversities/fulltext>> (accessed 2019-07-25).
- [13] 薄羽 大樹, 宮下 芳明: コメントイン:コメントを先に書くことによる新形態の API リファレンス, 第 23 回インタラクティブシステムとソフトウェアに関するワークショップ (WISS2015) 論文集, pp.67-72 (2015).
- [14] Joel Brandt, Mira Dontcheva, Marcos Weskamp, Scott R. Klemmer: Example-Centric Programming: Integrating Web Search into the Development Environment, *ACM Conference on Human Factors in Computing Systems (CHI'10)*, pp.513-522, ACM (2010).
- [15] Maria Hristova, Ananya Misra, Megan Rutter, Rebecca Mercuri: Identifying and correcting Java programming errors for introductory computer science students, *SIGCSE '03 Proceedings of the 34th SIGCSE technical symposium on Computer science education*, pp.153-156, SIGCSE (2003).

- [16] Microsoft: Visual Studio Code(online), available from <<https://code.visualstudio.com/>> (accessed 2019-07-25).
- [17] SQLite: SQLite(online), available from <<https://sqlite.org/index.html/>> (accessed 2019-07-25).
- [18] Python Software Foundation: Python 3.7.4 ドキュメント (オンライン), 入手先 <<https://docs.python.org/ja/3/>> (参照 2019-07-25).