日本語プログラミング言語の可読性と普及率

新村　裕太（愛知大学大学院経営学研究科）
毛利　元昭（愛知大学経営学部）
岩田　員典（愛知大学経営学部）

要旨

プログラミング言語の中でも特に有名で普及率が高いものには「Python」,「Java」,「JavaScript」,「C/C++」などがある。これらのプログラミング言語は英語がベースで，英語が母国語でない人々にとっては言語の壁がある。一方，我々が使い慣れた母国語で記述できる「日本語プログラミング言語」は，未だ認知度が低く，普及率も低い。構文が自然言語の日本語に近づけて作られた日本語プログラミング言語である「プロデル」は，プログラミング初学者に利用されやすい構文であるが，教育分野では利用されていないなど，「ミスマッチ」を起こしている。そして，そのような「ミスマッチ」を起こさず，日本語プログラミング言語の普及率を上げるには，プログラミング教育が 2020 年度から小学校，2021 年度から中学校，2022 年度から高校で必修化されたことを利用し，初等教育でプログラミング的思考が育まれた状態にし，中等教育以降で日本語プログラミング言語を活用することが重要である。

キーワード：日本語プログラミング言語，なでしこ，プロデル，ドリトル，プログラミング教育

1. はじめに

近年，様々なプログラミングツールが普及しつつあるが，我々が使い慣れた母国語で記述できる「日本語プログラミング言語」は，未だ認知度が低く，普及率も低い状況にある。日本語プログラミング言語の中で有名なものには「なでしこ」,「プロデル」,「ドリトル」というものがある。なでしこは，前身の「ひまわり」という言語の頃から事務の自動化を目標に開発されているという特徴がある。そのため，ファイル処理，画像処理，Word/Excel 連携，ネットワーク，文字列処理など，日々の定型作業を自動化するための便利な命令を 1000 以上備えている。作業の自動化をはじめ，趣味，教育，仕事など幅広く利用することが可能である。プロデルは，特徴として「読めばすぐ分かるプログラム」,「実用的な用途」,「速い」,「プラグイン」の 4 つを掲げている。また，日本語プログラミング言語の中では珍しく，教育分野での使用のためには作られておらず，汎用的なソフト

ウェア開発を目標として作られている。ドリトルは，なでしこやプロデルと比べるとより教育用途に特化しており，教育現場を中心に普及が進んでいるのが特徴である。リファレンスやマニュアルも主には授業用テキストや教師向けの授業資料という形で整備され，配布されている。

本研究では，これらの日本語プログラミング言語の特性を把握し，その普及を妨げる原因について，英語記述のプログラミング言語との可読性の比較，アンケート調査などを踏まえて考察し，普及方法や新たな活用方法を模索することを目的とする。

本稿の構成は次の通りである。まず第1章で，本稿の目的を説明する。第2章では，プログラミング言語の定義，言語処理系，プログラミング言語の種類について述べる。第3章では，日本語プログラミング言語について，なでしこ，プロデル，ドリトルの3つを用いて解説し，日本語プログラミング言語の現状を述べる。第4章では，日本語プログラミング言語の普及率が低い原因を，可読性や必要性の観点から考察し，プログラミング教育での活用についても述べる。第5章では，まとめを述べる。

2. プログラミング言語について

本章では，プログラミング言語について述べる。第2.1節でプログラミング言語について定義し，第2.2節で言語処理系，第2.3節でプログラミング言語の種類という順に述べる。

2.1. プログラミング言語

我々が普段話す日本語のような日常的に使われている言語は，自然言語と呼ばれている。プログラミング言語はそれとは別で，人の手により作られた特殊な規則に基づく言語で，形式言語と呼ばれている。言語とは，一定のルールのもとで文字，音声を組み合わせて意味を表すもの，あるいはその体系のことである。そして，プログラミングとは，コンピュータへの指示書となるプログラムを作成することである。つまり，プログラミング言語とは，コンピュータに指示をするため，文字で意味を示すことやその規則と言える。

プログラミング言語の存在意義は，人間が直接扱うには難しい機械語に代わって，より人間が扱いやすい形を提供することにある。機械語とは，コンピュータのマイクロプロセッサ（CPU/MPU）が直接解釈，実行できる命令コードの体系のことであり，0と1を並べたビット列として表され，人間が直に読み書きしやすい形式ではない。そして，コンピュータが直接理解し実行することのできる言語は，そのコンピュータの種類に固有の機械語だけである。したがって，最終的

には機械語を使ってコンピュータが行うべき作業，計算を指示しなければならない。機械語命令と一対一に対応するアセンブリ言語もプログラミング言語の一つではあるが，一般的には，C 言語や Java などより人間に解り易いプログラミング言語が使われる事が多い。アセンブリ言語は低水準言語（低級言語），C 言語や Java などは高水準言語（高級言語）と呼ばれる。高水準言語で書かれたプログラムを機械語プログラムに変換するのがコンパイラである。機械語プログラムに変換するのではなく，高水準プログラミング言語で書かれたソースコード（プログラム）を逐次解釈，実行するものをインタプリタと呼ぶ。そして，プログラムの実行に主としてインタプリタを用いるプログラミング言語をインタプリタ言語（またはスクリプト言語）と呼び，有名なものには「JavaScript」，「VBScript」，「Perl」などがある。これらの言語処理系の違いについては第 2.2 節で述べる。

AI などを用いて自然言語を機械で処理することを自然言語処理と言うが，開発には未だに難しい点も多く，中でも自然言語には言葉の「あいまいさ」，「意味の重複」が含まれていることから，感情や文脈などの解釈違いにより，勘違いをする場面が多々ある。

例として，図 1 の「いぬがねこをうんだよ」という文章を挙げる。この文章の解釈方法は，①のように「いぬ／が／ねこ／を／うんだよ」すなわち，「犬が猫を産んだよ」という区切り方と，②のように「いぬ／がね／こ／を／うんだよ」すなわち，「犬がね，子を産んだよ」という区切り方ができる。人間は犬が猫を産む訳がないと簡単に理解できるため，②が正解であると導き出せるのに対し，コンピュータの自然言語処理である予測変換などでは①が表示されてしまう。このように，「あいまいさ」や「意味の重複」があってはコンピュータに正確な処理をさせる事が難しい。そのため，プログラミング言語のような形式言語は，その様な要素はすべて取り除き，厳格な文法規則に基づいている。そのため，プログラミング言語は自然言語とは違い，コンピュータでの処理が容易となっている。

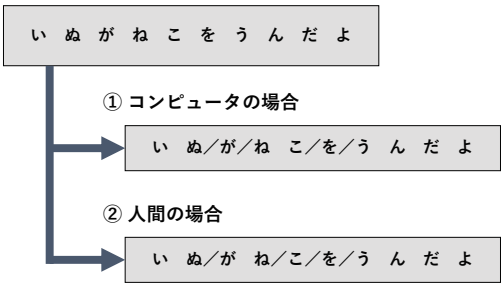

図 1　いぬがねこをうんだよ[8)]

2.2. 言語処理系

言語処理系とは，プログラミング言語で記述されたプログラムを計算機上で実行するためのソフトウェアである。その構成として，インタプリタとコンパイラ

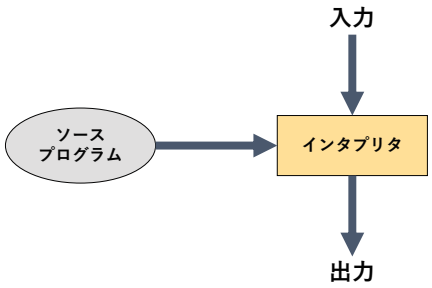

図２　インタプリタ[11)]

という２つがある。

インタプリタとは，言語を逐次意味解析しながら，その意味する動作を実行するものである（図 2)。

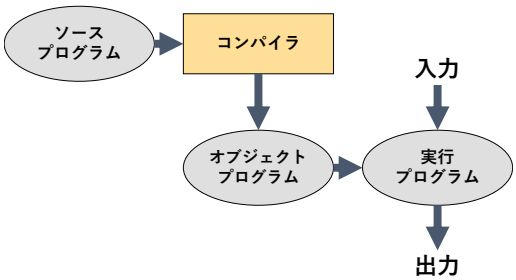

図３　コンパイラ[11)]

コンパイラとは，言語を他の言語（機械語や中間言語）に変換し，その言語のプログラムを計算機上で実行させるものである（図 3)。元のプログラムをソースプログラム，翻訳の結果と得られるプログラムをオブジェクトプログラムと呼ぶ。機械語で直接，計算機上で実行できるプログラムを実行プログラムと呼ぶ。オブジェクトプログラムがアセンブリプログラムの場合には，アセンブラにより機械語に翻訳されて，実行プログラムを得る。他の言語の場合には，オブジェクトプログラムの言語のコンパイラでコンパイルすることにより，実行プログラムが得られる。仮想マシンコードの場合には，オブジェクトコードはその仮想マシンにより，インタプリトされて実行される。

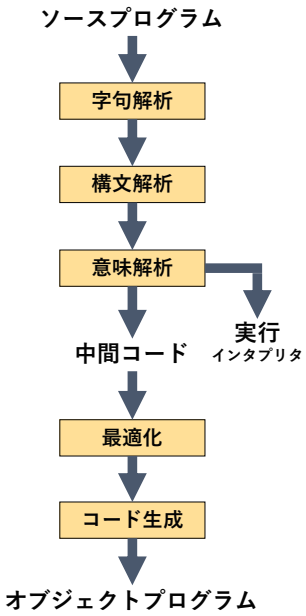

図４　言語処理系の基本構成[11)]

言語処理系の基本構成（インタプリタとコンパイラの基本構成）は主に５つに分かれる（図 4)。

1. 字句解析:文字列を言語の要素（トークン）の列に分解する。
2. 構文解析：トークン列を意味反映した構造（木構造）に変換する。
3. 意味解析：木構文の意味を解析する。インタプリタでは，ここで意味を解析し，それに対応した動作を行う。コンパイラでは，この段階で中間コード（内部的なコード）に変換する。

4. 最適化：中間コードを変形して，効率の良いプログラムに変換する。
5. コード生成：内部的なコードをオブジェクトプログラムの言語に変換し，出力する。例として，中間コードからターゲットの計算機のアセンブリ言語に変換する。

コンパイラの性能とは，如何に効率の良いオブジェクトコードを出力できるかであり，最適化でどのような変換ができるかによる。インタプリタでは，プログラムを実行するたびに字句解析，構文解析を行うため，実行速度はコンパイラの方が高速である。機械語に翻訳するコンパイラの場合は，直接機械語で実行されるために高速である。コンパイラでは中間コードでやるべき操作の全体を解析することができるため，高速化が可能である。

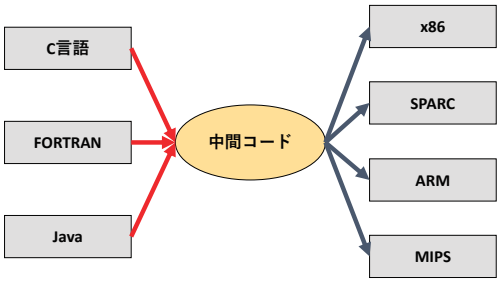

図５　中間コード[11)]

中間言語として都合の良い中間コードを用いると，様々な言語から中間言語への変換プログラムを作ることによって，それぞれの言語に対応したコンパイラを作ることができる（図５）。

2.3. プログラミング言語の種類

現在，プログラミング言語は Python や Java, C++, JavaScript などのメジャーなものからマイナーなものまで合わせると 200 種類以上存在していると言われている。その中には，企業が作成したものから個人が作成したものまで数多く存在し，文字を書くものだけではなく，視覚的に表現されたブロックを組み合わせるものや画像データを用いる言語など様々な形態が採られている。このように多種多様なプログラミング言語が存在する背景には，大きく分けて３つの要因が関わっている。

第一に，プログラミング言語ごとに得意分野が異なるという点が挙げられる。Web 開発であれば，PHP や JavaScript, Ruby，Python などのスクリプト言語がまず候補に挙がる。しかしながら，その中の１つの言語だけ覚えておけば，Web 開発以外でもすべてのシーンで使えるかというとそうではない。例えば PHP は，Web 開発に特化しており，iPhone や iPad で動作するアプリを作ることにはあまり適していない。iPhone や iPad 用のアプリを制作するには，Objective-C か Swift が主要となっている。Android 用のアプリを作るならば，Java や Java との互換性が 100% である Kotlin を用いる必要がある。また，スマートフォンアプリを開発するにも，ゲーム開発で

はC# を用いてUnity，またはC++ を用いてUnreal EngineやCocos2d-xで開発するのが現在の主流となっている。さらに，一般的なプログラミング言語とは少し異なるが，Webサイトを表示するには，HTMLという専用のマークアップ言語を使用してページ記述をし，データベースを操作する場合には，SQLというデータベースを操作する専用の言語を使用する。コンピュータの種類によっても使用する言語が異なってくる場合もある。このように，プログラミング言語ごとに専門分野や得意分野が異なり，何をどのような環境で作りたいかに応じて，プログラミング言語を変える必要がある。上記であげたプログラミング言語を含め，多くのプログラミング言語では，複数の用途に使用できるようになってはいるが，得意分野と苦手分野が明確に存在する。そのため，その時々に応じて適したプログラミング言語を使い分けることにより，効率良く開発を進める事ができるため，複数のプログラミング言語が生まれた要因となっている。

第二に，プログラミング言語の進化という点が挙げられる。ITの世界は常に進化しているイメージが強いが，それはプログラミング言語においてもそうである。プログラミング言語が使われていく中で，その言語では解決できない，もしくは解決しにくい課題が見つかっていき，それに合わせて様々な新しいプログラミング言語が開発され，その言語が主流となっていく場合もある。例えば，かつてWebサーバではPerlが標準的なプログラミング言語であったが，同じくWebサーバに特化しており，性能や書きやすさで優れたPHPが人気になり，一気にWebサーバ分野での使用率を抜かしていった。また，最近では，JavaとKotlinの間でも同様のことが起ころうとしている。KotlinがJavaと比較して優れているといわれる点は，コンパクトなコードによる安定性とセキュリティの向上，安全性を重視したコンパイラ，豊富なIDE（統合開発環境）の選択肢，生産性向上の可能性などがある。しかし，それ以上に重大な点は，KotlinとJavaの互換性が100％であることにある。つまり，JavaからKotlinに移行するのに余分な作業が必要なく，Javaのフレームワークも利用できるため，移行リスクを最小限にして，より利便性の高い言語を使用できるのである。また，2017年には，KotlinがAndroid公式言語となり，2019年には，Android上でKotlinファーストの推進を発表し，Java以上にAndroid上で優先されるプログラミング言語となっている。このような事が要因となり，Kotlinは近年急速にユーザーを伸ばし，Javaに取って代わろうとしている。このように新しいプログラミング言語を作ることによって，これまで時間のかかっていた処理を簡単に実現でき

るようになるなど，効率性，利便性の向上を見込めることもあるため，プログラミング言語の種類が増加していく要因となっている。

第三に，プログラミング言語の多様性という点が挙げられる。プログラミング言語は誰でも自由に作れ，その新しく作られる言語は，第二でも述べたように何かしらの課題解決のために生まれている事が多い。そのため，同様の用途のプログラミング言語同士であっても，完全に近い進化であるかどうかに関わらず，劣っている点もあれば，優れている点もあるといった状態で並立して生き残っている。それぞれ独自の事情やセールスポイントがあるからこそ，並立していても何かしらの使い分けが存在し，共存が可能になっているという事である。さらに，IT の分野では一般的に独占が起きづらいと言われており，常に多様な選択肢が存在する。例えば，スマートフォン一つとっても，Apple の iOS や，Google の Android など複数の OS が存在し，どの OS にも良い点があり，一概にどちらが優れているとは言えない。そして，それぞれの OS に合わせた言語も発表され，プログラミング言語の多様性も生まれているのである。その他にも，用途的に並立した言語同士であっても個人の書き方やプラットフォームなどに関する好みによって分かれてくる。例えば，Java と C# はどちらもデスクトップアプリを開発したり，Web アプリを開発したりすることが可能で，静的型付けを行うオブジェクト指向のプログラミング言語という点や，一度は中間形式に変換される点など，細かい点でも似ている点がある。しかし，その他の書き方やプラットフォームなどでは異なっている点も多く，Java と C# のそれぞれに数多くの支持者がおり，現在も開発継続されている。このように，用途的に並立しても消えずに存在することが多いという，プログラミング言語の多様性も，複数のプログラミング言語が存在する要因となっている。

上記で挙げたプログラミング言語以外のものとして，本研究の題材でもある，「日本語プログラミング言語」という種類がある。第 1 章でも述べたように，日本語プログラミング言語は，未だ認知度が低く，普及率も低い状況にある。日本語プログラミング言語の中で有名なものには「なでしこ」，「プロデル」，「ドリトル」というものがある。この 3 つのような日本語プログラミング言語同士でも，様々な用途や特徴があるが，その中でもほとんどの日本語プログラミング言語に共通していて，最も使用率の高い用途が，プログラミング初学者や学生向けのプログラミング教育である。日本人にとって日本語で読み書きできるという点は最大の利点であるため，プログラミング入門用の言語として期待できる。ま

た，日本語プログラミング言語の先駆けとなったMindや，熱心に開発が続くなでしこ，プロデル，ドリトルはいずれもトイ（知育玩具用）言語ではなく，十分に本格的な言語であり，教育分野以外でも多くの業務に使用可能となっている。

3. 日本語プログラミング言語について

本章では，第3.1節で日本語プログラミング言語について，第3.1.1項でなでしこ，第3.1.2項でプロデル，第3.1.3項でドリトル，第3.2節で日本語プログラミング言語の現状という順に述べる。

3.1. 日本語プログラミング言語

日本語プログラミング言語とは，日本語風のプログラミング言語のことである。ひらがなやカタカナ，漢字などを用いて日本語でソースコードを書き，アプリケーション開発を行うことができる。

日本語プログラミング言語の歴史は意外と古く，1982年に発表されたぴゅう太には，日本語BASIC（G-BASIC）が搭載されていた。1983年には，ワープロ感覚で日本語をもちいてプログラミングができればという考えで開発された和漢が開発された。1985年には，Forthの語順が日本語に近いことに注目したMindが開発された。そして，2000年には，TTSneo，プロデル，ひまわり，ドリトルなど，インタプリタ型の日本語プログラミング言語が次々とフリーソフトとして発表された。

日本語プログラミング言語の最大の利点は，我々が使い慣れた母国語である日本語で記述できるという点である。しかしながら，日本語プログラミング言語は，未だ認知度が低く，普及率も低い状況にある。そのため，「日本語プログラミング言語の可読性と普及率」について考察する必要がある。まず初めに，日本語プログラミング言語の中でも有名である，「なでしこ」，「プロデル」，「ドリトル」について特徴や使用用途を述べる。

3.1.1. なでしこ

ホームページでの紹介には，「『なでしこ3』は，日本語をベースに開発されたプログラミング言語です。主にWebブラウザで動かすことを目的に開発されています。ブログやWebサイトに組み込んで使えます。また，OSや環境を問わず，PC/スマホ/タブレットとさまざまな環境で動きます。作業の自動化をはじめ，趣味，教育，仕事などに利用できます。」とある[14)]。

このようになでしこは，日本語で記述をするため，日本人のプログラミング初学者や，学生，英語の苦手な人などに向けて作られている。しかし，元々，前身のひまわりという言語の頃から事務の自動化を目標に開発されている。そのた

め，ファイル処理，画像処理，ネットワーク，Word/Excel 連携，文字列処理など，日々の定型作業を自動化するための便利な命令を 1000 以上備えている。そして，なでしこは 2004 年に公開されて以降，2017 年には Web ブラウザでも動かせる v3（CoffeeScript や TypeScript のように，JavaScript に変換されて実行される広義の altJS）が公開されるなど，その後も活発に開発が続けられている。

3.1.2. プロデル

ホームページでの紹介には，「プロデルは，日本語でプログラムが書けるプログラミング言語です。次のような特徴を持っています。プロデルの文法は日本語文のように書けるように設計されています。プログラム例文を見ればすぐに何をしているか理解できます。バッチ処理や画像描画，GUI 部品の機能が用意されています。Excel，CSV 形式操作，ODBC 接続もできます。ちょっとしたツールやゲーム作りも可能です。TTSneo よりも高速に動くようになりました。大量のデータ処理も短時間で実行できます。また MSIL コンパイラも搭載していますので，コンパイルで更なる高速化も可能です。プラグインで FeliCa や Twitter との連携機能を利用できます。また C# を使って独自のプラグインを作ることもできます。」とある[15]。

このようにプロデルは，「読めばすぐ分かるプログラム」，「実用的な用途」，「速い」，「プラグイン」の 4 つを特徴として挙げられる。そのため，日本語プログラミング言語の中では珍しく，教育分野での使用のためには作られておらず，汎用的なソフトウェア開発を目標としている。また，プロデルにはプロデル Web サーバという，プロデルで開発された Web アプリケーションを動作させるための専用 Web サーバが用意されている。この Web サーバを利用することにより，プロデルで Web アプリケーション開発の実地やサービスの公開が可能となる。プロデルは同じく日本語プログラミング言語の TTSneo を前身とし，仕様を一新，アップグレードする形で公開された言語である。TTSneo の公開は 2002 年，さらに，TTSneo の前身である TTS が公開されたのは 2000 年であるため，TTS の公開から数えると，プロデルは 20 年以上もの歴史を持つ。

3.1.3. ドリトル

ホームページでの紹介には，「ドリトルは教育用に設計されたプログラミング言語です。中学校・高校の教科書や副教材などに採用されています。小学校（総合・算数・理科・音楽など），中学校（技術科の計測制御・双方向コンテンツ），高等学校（情報の科学・社会と情報，情報 I・情報 II），大学（プログラミング入門，IoT）などでご利用ください。」とある[16]。

このようにドリトルは，なでしこやプロデルと比べ，より教育用途に特化し，教育現場を中心に普及が進んでいるのが特徴である。リファレンスやマニュアルも主には授業用テキストや教師向けの授業資料という形で整備され，配布されている。

3.2. 日本語プログラミング言語の現状

日本語プログラミング言語の現状について，「日本語で記述できるプログラミング言語の存在を知っていますか？」という内容の認知度アンケート調査を行った（図 6）。アンケート調査を行った対象は，いずれかのプログラミング言語を授業もしくは，職業関連などで一度は触れたことがある方のみ，100 名である。アンケート対象の年齢には特に制限は設けず，年代ごとに認知度が大きく変わるという事もなかったため，年齢は表記しない。

アンケート結果は，「はい」と回答した方が 12 名，「いいえ」と回答した方が 88 名となった。ただし，アンケート対象 100 名の中では，いずれかのプログラミング言語に職業関連などで触れたことがある方が 78 名，授業で触れたことがあるだけの方が 22 名。職業関連の方が比べて多いため，「はい」が多少多い可能性も考えられる。いずれかのプログラミング言語に一度は触れたことがある方のみに絞っても認知度がかなり低く，どのプログラミング言語にも一度も触れたことがない方を含めれば，更に認知度が低くなる結果が出ると考える。

このように，日本語プログラミング言語は我々の母国語である日本語で記述できるプログラミング言語であるにも関わらず，未だ認知度が低く，普及率も低い状況にある。そのため，「日本語プログラミング言語の可読性と普及率」につい

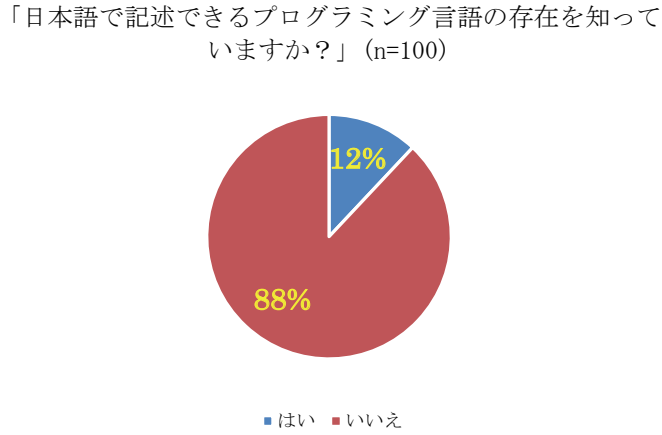

図 6　プログラミング言語の認知度調査

て考察する必要がある。そして，普及率が低い原因について考察するにあたり，日本語プログラミング言語と英語記述のプログラミング言語との可読性の比較や，日本語プログラミング言語同士での可読性の比較を行う必要がある。第 4 章では，これらの調査を行った上で，普及率の高い言語との活用目的，環境，構文などの違いを見つけ出し，普及率にどの様な影響を及ぼしているか考察し，日本語プログラミング言語の普及率を上げる方法について述べる。

4. 日本語プログラミング言語の可読性と普及率

本章では，第 4.1 節で英語記述のプログラミング言語との可読性の比較，第 4.2 節で日本語プログラミング言語同士での可読性の比較，第 4.3 節で日本語プログラミング言語の必要性，第 4.4 節で普及率が低い原因，第 4.5 節でプログラミング教育での活用，第 4.6 節で初等教育からのプログラミング教育の必要性，第 4.7 節でプログラミング教育の実態という順に述べる。

4.1. 英語記述のプログラミング言語との可読性の比較

日本語プログラミング言語と英語記述のプログラミング言語との可読性の比較をするにあたって，① Java と②なでしこについてアンケート調査を行った。両者ともアンケート内容は「FizzBuzz 問題」について記述された簡易的なソースコードである。アンケート調査を行った対象は，プログラミング言語に触れた経験がない方も含めた 100 名である。アンケート対象の年齢には特に制限は設けず，年

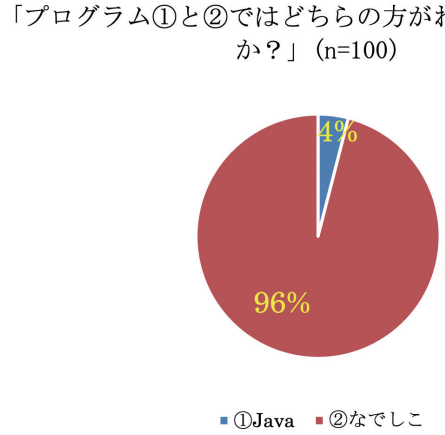

図 7　英語記述のプログラミング言語との可読性の比較

代ごとに認知度が大きく変わるという事もなかったため，年齢は表記しない。以上を前提として，「プログラム①と②ではどちらの方がわかりやすいですか？」という内容でアンケート調査を行った(図7)。また，そのように回答した理由についても同時に集計を行った。

アンケート結果は，「①」と回答した方が4名，「②」と回答した方が96名となった。ほぼ全員がJavaよりなでしこの方がわかりやすい，すなわち英語記述のプログラミング言語より日本語プログラミング言語の方が，可読性が高いといえる結果となった。「①」と答えた理由としては，「Javaのような英語記述のプログラミング言語は使い慣れているため」，「普段からJavaを使うことが多いため」，「日本語プログラミング言語は初めて見たので少し違和感があるため」，「知らないプログラミング言語よりも知っているプログラミング言語の方がわかりやすいため」という4つが挙がった。この事から，「①」と回答した4名ともに共通する要因は，Javaとなでしこの認知度や普及率の差であると考える。日本語プログラミングは英語記述のプログラミング言語と比べ，遥かに認知度や普及率で劣っているため，当然といえる意見である。そのため，これらの意見は，日本語プログラミング言語の普及率がより高くなった際には解決へ向かうと考える。

4.2. 日本語プログラミング言語同士での可読性の比較

日本語プログラミング言語の中でも有名な「なでしこ」，「プロデル」，「ドリトル」の3つの中では，なでしことプロデルは自然言語の日本語に近く，ドリトルはより英語記述のプログラミング言語に近い性質を持っている。第4.1節，日本語プログラミング言語と英語記述のプログラミングとの可読性の比較では，Javaとなでしこを用いてアンケート調査を行ったが，母国語である日本語で記述されているからという理由で，日本語プログラミング言語の方が英語記述のプログラミング言語よりわかりやすいと回答した方がほとんどであった。しかしながら，母国語である日本語プログラミング言語の中でも，構文が自然言語の日本語に近づけて作られた言語と，英語記述のプログラミング言語に近づけて作られた言語で分かれている。

そこで本節では，日本語プログラミング言語同士での可読性の比較をするため，①なでしこと②ドリトルについてアンケート調査を行った。アンケート内容は「FizzBuzz問題」について記述された簡易的なソースコードである。アンケート調査を行った対象は，プログラミング言語に触れた経験がない方も含めた100名である。アンケート対象の年齢には特に制限は設けず，年代ごとに認知

度が大きく変わるという事もなかったため，年齢は表記しない。以上を前提として，「プログラム①と②ではどちらの方がわかりやすいですか？」という内容でアンケート調査を行った（図 8）。

アンケート結果は，「①」と回答した方が 45 名，「②」と回答した方が 55 名となった。一見，ほぼ同じようにも見えるが，「①」と回答した方は，ほとんどがプログラミング言語に触れた経験がない，もしくはプログラミング言語を授業で触れたことがある程度であった。それに対し，「②」と回答した方は，プログラミング言語に職業関連などで触れたことがある，かなり英語記述のプログラミング言語を使い慣れている方がほとんどであった。すなわち，なでしこのように構文が自然言語の日本語に近づけて作られている言語は，プログラミング初学者や，プログラミング言語の学習途中の学生などに受け入れられやすい傾向があると考える。そして，ドリトルのように，構文が英語記述のプログラミング言語に近づけて作られている言語は，いずれかのプログラミング言語を習得している，もしくはプログラミング言語の学習過程にあるが，既にある程度十分な知識を持っている方に受け入れられやすい傾向があると考える。そのため，教育分野での普及では，構文を自然言語の日本語に近づけて作られた，学習用の日本語プログラミング言語がカギとなり，仕事で実際に使用するなどの本格的な業務分野での普及では，構文を英語記述のプログラミング言語に近づけて作られた，ソフトウェア開発環境の整った日本語プログラミング言語がカギとなる。

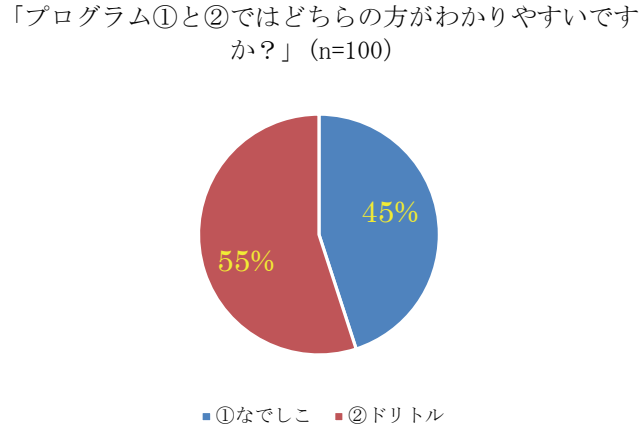

図 8　日本語プログラミング言語同士での可読性の比較

4.3. 日本語プログラミング言語の必要性

日本語プログラミング言語の必要性として，日本人のプログラミング初学者や，学生，英語の苦手な人などにとって学習コストが低いという事ももちろん挙げられるが，必要になってくる要因の中でも最も大きなものとして，言語間距離がある。

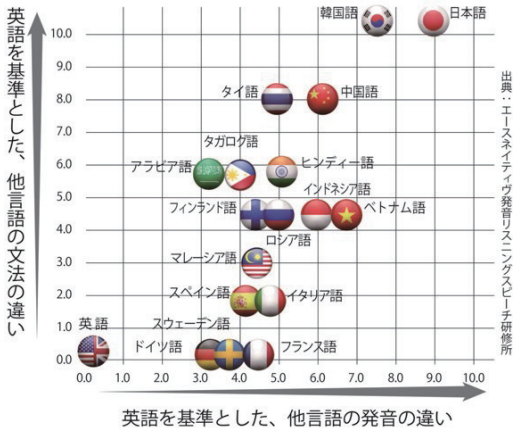

図 9　言語間距離[18)]

インド・ヨーロッパ語族の言語として代表的なものには英語がある。しかしながら，図 9 からわかるように，日本語は英語からかけ離れた言語である。そのため，プログラミング言語においては，他の国に比べても，英語記述のプログラミング言語ではなく，母国語である日本語の言語の必要性が高くなってくる。また，言語間距離の原因として，日本語との単語や文法，発音，文化，風習などの違いが考えられる。その中でも，プログラミング言語について考察するため，文法に着目すると，日本語は SOV 型であるが，インド･ヨーロッパ語族の言語（ドイツ語，オランダ語を除く）は SVO 型である。文法の違いは可読性の高さについて考えていく際にも重要な要素になり得る。

4.4. 普及率が低い原因

ここまでに述べたことを踏まえた上で，日本語プログラミング言語の普及率が低い原因として，日本語プログラミング言語を取り巻く環境で起きている「ミスマッチ」が挙げられる。第 2.3 節，プログラミング言語の種類でも述べたように，プログラミング言語ごとに専門分野や得意分野が異なる。さらに，コンピュータの種類によっても使用する言語が異なってくる場合もあるため，何をどのような環境で開発するかに応じて，プログラミング言語を変える必要がある。また，第 4.2 節，日本語プログラミング言語同士での可読性の比較で挙げたアンケート結果より，教育分野での普及では，構文を自然言語の日本語に近づけて作られた，学習用の日本語プログラミング言語がカギとなり，仕事で実際に使用するなど本格的な業務分野での普及では，構文を英語記述のプログラミング言語に近づけて作られた，ソフトウェア開発環境の整った日本語プログラミング言語がカギとなる。

まず初めに，教育分野での普及について述べる。日本語プログラミング言語の中では，ドリトルが，なでしこやプロデルと比べて最も教育分野に特化しており，教育現場を中心に普及が進んでいるのが特徴である。また，リファレンスやマニュアルも主には授業用テキストや教師向けの授業資料という形で整備され，配布されている。このような事から，ドリトルを教育用途で利用するのに適した環境は整っているように見える。しかしながら，ドリトルは構文を英語記述のプログラミング言語に近づけて作られた言語であり，それを利用すると考えられる層は，アンケート結果より，いずれかのプログラミング言語を習得している，もしくはプログラミング言語の学習過程にあるが，すでにある程度十分な知識を持っている層である。

次に，業務分野での普及について述べる。日本語プログラミング言語の中では，プロデルが，なでしこやドリトルと比べて最も業務分野に特化している。プロデルは，日本語プログラミング言語の中では珍しく，教育分野での使用のためには作られておらず，汎用的なソフトウェア開発を目標としているのが特徴である。バッチ処理や画像描画，GUI 部品の機能が用意されている，Excel，CSV 形式操作，ODBC 接続が可能，ツールやゲーム作りが可能，大量のデータ処理を短時間で実行可能，MSIL コンパイラを搭載しているので，コンパイルで更なる高速化が可能，プラグインで FeliCa や Twitter との連携機能を利用可能，C# を使って独自のプラグイン制作が可能など実用的である。このような事から，プロデルを業務用途で利用するのに適した環境は整っているように見える。しかしながら，プロデルは構文を自然言語の日本語に近づけて作られた言語であり，それを利用すると考えられる層は，アンケート結果より，プログラミング未経験の初学者や，プログラミング言語の学習途中の学生などの層である。

このように，教育分野での普及，業務分野での普及ともに，利用するのに適した環境は整っているように見える。しかしながら，構文が自然言語の日本語に近づけて作られた日本語プログラミング言語である「プロデル」は，構文の性質が日本語に近い分，プログラミング初学者に利用されやすいが，業務分野で利用するのに適した環境になっている。すなわち，プログラミング初学者に利用されやすい構文であるが，教育分野では利用されていないなど，日本語プログラミング言語の構文の性質と，利用されている環境が「ミスマッチ」を起こしている。そして，そのような「ミスマッチ」を起こさず，日本語プログラミング言語の普及率をより上げるには，プログラミング教育が 2020 年度から小学校，2021 年度から中学校，2022 年度から高校で必修化

されたことを利用し，初等教育でのプログラミング教育でプログラミング的思考が育まれた状態にし，中等教育以降でのプログラミング教育で日本語プログラミング言語を活用することが重要である。

4.5. プログラミング教育での活用

はじめに，プログラミング教育の目的はプログラミング的思考を養うことにあり，プログラミング的思考とは，「自分が意図する一連の活動を実現するために，どのような動きの組合せが必要であり，一つ一つの動きに対応した記号を，どのように組み合わせたらいいのか，記号の組合せをどのように改善していけば，より意図した活動に近づくのか，といったことを論理的に考えていく力」と文部科学省によって定義されている。また，「児童がおのずとプログラミング言語を覚えたり，プログラミングの技能を習得したりするといったことは考えられますが，それ自体をねらいとしているのではない」とも述べられている。つまり，プログラミング教育の目的は，技能の習得ではなくプログラミング的（論理的）思考力を身に着けることである。

そして，そのような目的であるプログラミング教育において日本語プログラミング言語を活用していくには，「動きに対応した記号」が日本語で表現されている日本語プログラミング言語で書かれたコードは，日本語を理解できれば手順書のように読めるが，曖昧さを含まないように一定のルールに従って記述されている必要がある。この特性を活かして，読解力やわかりやすく記述する力を養うといった活用ができるのではないかと考える。

4.6. 初等教育からのプログラミング教育の必要性

プログラミング教育の必要性として最も重要な要素に，論理的思考力を鍛えることがある。そして，その論理的思考力を早期段階である初等教育の頃から鍛えることが重要であると考える。また，文部科学省の文書には，「今後の社会の在り方について，とりわけ最近では，『第4次産業革命』ともいわれる，進化した人工知能が様々な判断を行ったり，身近な物の働きがインターネット経由で最適化されたりする時代の到来が，社会の在り方を大きく変えていくとの予測がなされているところである。教育界には，そのような社会的変化の中でも，子供たちが自信を持って自分の人生を切り拓き，よりよい社会を創り出していくことができるそうした資質・能力として，読解力，論理的思考力，創造性，問題解決能力などは，時代を超えて常にその重要性が指摘されてきており，これからの時代においてもその重要性が変わることはない。

これらに加えて，情報や情報技術を問題の発見・解決に活用していく力（情報活用能力）の重要性も高まっている。学校教育は，こうした資質・能力の育成に向けて充実を図らなければならない。」というような記述もあり，これからの時代，より一層重要性が増すと考える。

また，第 4.4 節，普及率が低い原因でも述べたように，日本語プログラミング言語の普及率をより上げるには，プログラミング教育が 2020 年度から小学校，2021 年度から中学校，2022 年度から高校で必修化されたことを利用し，初等教育でのプログラミング教育でプログラミング的思考が育まれた状態にし，中等教育以降でのプログラミング教育で日本語プログラミング言語を活用することが重要である。また，この方法が日本人にとっては最も学習コストが低く，効率の良い学び方だと考える。そのため，中等教育以降での学習成果をより上げるためにも，初等教育でのプログラミング教育による下積みが必要性として挙げられる。

4.7. プログラミング教育の実態

コロナウイルスの影響による大きな遅延はあったものの，2020 年度から初等教育でのプログラミング教育が実際に開始された。そこで，プログラミング教育の実態調査を行い，現場の教師はどのような実感や考えであるか，小学校教師の方に「児童の読解力・記述力は 5 年ほど前と比較してどう変わったか」，「児童の算数の回答力は 5 年ほど前と比較してどう変わったか」，「小中学校でのプログラミング教育を必要と感じるか否か」，「現在のプログラミング教育について困っていること」，「日本語で記述できるプログラミング言語を初等教育に利用してみたいか否か」の 5 項目についてご回答頂いた。

まず初めに，プログラミング教育で向上するといわれている読解力や記述力の現状について把握するため，「児童の読解力・記述力は 5 年ほど前と比較してどう変わったか」について述べる。児童の読解力・記述力については，低下の傾向にあるそうである。それは，日頃から自分の頭で考えることが減ってきていることと関係しているように感じ，答えがわからなかったらすぐに大人に聞いたり，調べたりする習慣がつき，考える力を養う機会を逃しているように思えるとのことである。その結果，文章から様々なことを読み取ったり，想像したりすることが苦手な子たちが増え，また，記述力に関しても，「何を書いたらいいかわからない」と考えることをやめてしまう子や，頭の中で自分の思考を整理して文章に表すことが苦手な子が増えてきている現状であるそうだ。低下の一途をたどっているからこそ，このタイミングで初等教育でのプログラミング教育が開始され

たのは正解であったと言える。

次に，プログラミング教育で能力を育んでいく際に重要な部分となっていく算数の力について把握するため，「児童の算数の回答力は5年ほど前と比較してどう変わったか」について考察する。児童の算数の回答力基本的な計算の力は大きな変化はないように感じるそうである。しかし，子どもが自分の考えを発表する場面で，友達にわかりやすく，順序立てて説明することを苦手とする子は増えてきているようである。プログラミング的思考とは，「自分が意図する一連の活動を実現するために，どのような動きの組合せが必要であり，一つ一つの動きに対応した記号を，どのように組み合わせたらいいのか，記号の組合せをどのように改善していけば，より意図した活動に近づくのか，といったことを論理的に考えていく力」と文部科学省によって定義されているが，まさにこの力が低下してきているということがわかる。

次に，「小中学校でのプログラミング教育を必要と感じるか否か」について考える。論理的思考力を鍛えることは今の子たちにとって大切なことだと思い，従来の国語や算数などの科目の中でも，教材の扱い方や教師側の指導の仕方によって論理的思考力を育てることができるのではないかとのことである。これは，2020年度から開始された初等教育でのプログラミング教育の手法である，既存の科目にプログラミングの要素を取り込むという方法でやっていけるのではないかということである。

次に，初等教育でのプログラミング教育は2020年度より始まったばかりのため，なにかしらトラブルや解決しなければならない事柄はあるのか把握するため，「現在のプログラミング教育について困っていること」についてまとめる。初等教育でのプログラミング教育の開始1年目時点では，プログラミング教育が算数や理科などの教科書の中で一部扱われるようになってきたものの，国や県，市の研修などが進んでおらず，何をどのように教えていけばよいのか個人に任されている部分が多いため，はっきりとわかっていないのが現状のようである。これに関しては，開始1年目だからという事も考えられるが，最も大きな要因としては，コロナウイルスの流行により，2020年度の授業や研修が大幅に遅延したことが挙げられると考える。

最後に，「日本語で記述できるプログラミング言語を初等教育に利用してみたいか否か」について述べる。日本語で記述できるプログラミング教材は小学校の高学年，中学生くらいであれば活用できるのかも知れないが，まず，初等教育では国語や算数などの教科学習の中で，論理的思考力を育てていくことが大切であると考えているとのことである。やはり初等教育の間は，現状のプログラミング

教育のやり方でプログラミング的（論理的）思考力を育んでいくことが大切であり，日本語プログラミング言語などを導入するならば，中等教育以降で行うことが適切であると考える。

5. まとめ

本稿では，日本語プログラミング言語の普及を妨げる原因について，英語記述のプログラミング言語との可読性の比較，アンケート調査などを踏まえて考察し，普及方法や新たな活用方法を模索した。

日本語プログラミング言語の中では，ドリトルが，なでしこやプロデルと比べて最も教育分野に特化しており，教育現場を中心に普及が進んでいるのが特徴である。リファレンスやマニュアルも主には授業用テキストや教師向けの授業資料という形で整備され，配布されている。このような事から，ドリトルを教育分野で利用するのに適した環境は整っているように見える。しかしながら，ドリトルは構文を英語記述のプログラミング言語に近づけて作られた言語であり，それを利用すると考えられる層は，アンケート結果より，いずれかのプログラミング言語を習得している，もしくはプログラミング言語の学習過程にあるが，すでにある程度十分な知識を持っている層である。

日本語プログラミング言語の中では，プロデルが，なでしこやドリトルと比べて最も業務分野に特化している。プロデルは，日本語プログラミング言語の中では珍しく，教育分野での使用のためには作られておらず，汎用的なソフトウェア開発を目標としているのが特徴である。バッチ処理や画像描画，GUI 部品の機能が用意されている，Excel, CSV 形式操作，ODBC 接続が可能，ツールやゲーム作りが可能，大量のデータ処理を短時間で実行可能，MSIL コンパイラを搭載しているので，コンパイルで更なる高速化が可能，プラグインで FeliCa や Twitter との連携機能を利用可能，C# を使って独自のプラグイン制作が可能など実用的である。このような事から，プロデルを業務用途で利用するのに適した環境は整っているように見える。しかしながら，プロデルは構文を自然言語の日本語に近づけて作られた言語であり，それを利用すると考えられる層は，アンケート結果より，プログラミング未経験の初学者や，プログラミング言語の学習途中の学生などの層である。

このように，教育分野での普及，業務分野での普及ともに，利用するのに適した環境は整っているように見える。しかしながら，構文が自然言語の日本語に近づけて作られた日本語プログラミング言語である「プロデル」は，構文の性質が日本語に近い分，プログラミング初学者

に利用されやすいが，業務分野で利用するのに適した環境になっている。すなわち，プログラミング初学者に利用されやすい構文であるが，教育分野では利用されていないなど，日本語プログラミング言語の構文の性質と，利用されている環境が「ミスマッチ」を起こしている。そして，そのような「ミスマッチ」を起こさず，日本語プログラミング言語の普及率をより上げるには，プログラミング教育が 2020 年度から小学校，2021 年度から中学校，2022 年度から高校で必修化されたことを利用し，初等教育でのプログラミング教育でプログラミング的思考が育まれた状態にし，中等教育以降でのプログラミング教育で日本語プログラミング言語を活用することが重要である。

参考文献

1）クジラ飛行机,“プログラミング言語大全”，株式会社技術評論社，(2020).

2）文部科学省,“小学校段階における論理的思考力や創造性，問題解決能力等の育成とプログラミング教育に関する有識者会議”，2016 年 4 月 19 日更新. https://www.mext.go.jp/b_menu/shingi/chousa/shotou/122/index.htm［アクセス日：2022 年 11 月 18 日].

3）文部科学省,“小学校段階における論理的思考力や創造性，問題解決能力等の育成とプログラミング教育に関する有識者会議について”,2016 年 4 月 19 日更新. https://www.mext.go.jp/b_menu/shingi/chousa/shotou/122/attach/1370404.htm［アクセス日：2022 年 11 月 18 日].

4）文部科学省,“小学校段階におけるプログラミング教育の在り方について（議論の取りまとめ)”，2016 年 6 月 16 日更新. https://www.mext.go.jp/b_menu/shingi/chousa/shotou/122/attach/1372525.htm［アクセス日：2022 年 11 月 18 日].

5）文部科学省,“小学校段階における論理的思考力や創造性，問題解決能力等の育成とプログラミング教育に関する有識者会議　議事要旨・議事録・配付資料”，2016 年 6 月 3 日更新. https://www.mext.go.jp/b_menu/shingi/chousa/shotou/122/giji_list/index.htm［アクセス日：2022 年 11 月 18 日].

6）文部科学省,“小学校段階における論理的思考力や創造性，問題解決能力等の育成とプログラミング教育に関する有識者会議（第 3 回）議事録”，2016 年 6 月 3 日更新. https://www.mext.go.jp/b_menu/shingi/chousa/shotou/122/gijiroku/1382219.htm［アクセス日：2022 年 11 月 18 日].

7）文部科学省,“小学校段階における論理的思考力や創造性，問題解決能力等の

育成とプログラミング教育に関する有識者会議（第3回）配付資料”，2016年6月3日更新.
https://www.mext.go.jp/b_menu/shingi/chousa/shotou/122/shiryo/1371637.htm［アクセス日：2022年11月18日].

8）AI（人工知能）関連メディア,“【入門編】自然言語処理（NLP）とは｜意味・仕組み・活用例・課題”，2019年11月20日更新.
https://ledge.ai/nlp/［アクセス日：2022年11月18日].

9）Hatada’sHomePage（旧),“プログラミング言語の基本概念”，2016年更新.
http://shopping2.gmobb.jp/htdmnr/www08/lp2016/chap01/language.html［アクセス日：2022年11月18日].

10）IT用語辞典,“機械語”，2018年10月24日更新.
https://e-words.jp/w/%E6%A9%9F%E6%A2%B0%E8%AA%9E.html［アクセス日：2022年11月18日].

11）HPCS Lab.,“言語処理系とは”，2014年更新.
http://www.hpcs.cs.tsukuba.ac.jp/~msato/lecture-note/comp-lecture/note1.html［アクセス日：2022年11月18日].

12）TECHのススメ,“プログラミング言語とは？”，2020年9月16日更新.
https://haa.athuman.com/media/it/programming/2120/［アクセス日：2022年11月18日].

13）TechTarget,“Javaから「Kotlin」に乗り換えたくなる5つの理由”，2020年2月17日更新.
https://techtarget.itmedia.co.jp/tt/news/2002/17/news06.html［アクセス日：2022年11月18日].

14）なでしこ,“なでしこ3–日本語プログラミング言語”，2022年10月9日更新.
https://nadesi.com/doc3/［アクセス日：2022年11月18日].

15）日本語プログラミング言語「プロデル」,“プロデル”，2022年11月12日更新.
https://rdr.utopiat.net/［アクセス日：2022年11月18日].

16）プログラミング言語「ドリトル」,“教育用プログラミング言語「ドリトル」情報ページ”，2021年11月13日更新.
https://dolittle.eplang.jp/［アクセス日：2022年11月18日].

17）TECH CAMP,“「なでしこ」「プロデル」「ドリトル」三大日本語プログラミング言語を解説”，2021年2月1日更新.
https://tech-camp.in/note/technology/41396/#i-9［アクセス日：2022年11月18日].

18）ACE PRO,“衝撃の事実”，2022年2月22日更新.
https://www.ace-schools.co.jp［アクセス日：2022年11月18日].