CN-008

プログラミング初学者を対象としたオンラインジャッジシステム
Online Judge Systems for Novice Programmers

田中 和季† 村川 猛彦‡
Kazuki Tanaka Takehiko Murakawa

1. はじめに

1.1 研究の背景

多くの大学で，学部学科を問わず，コンピューターを用いた情報処理教育が行われている[1]．コンピューターの基本的な操作，文章作成ソフト，表計算ソフト，プレゼンテーションソフトの使い方だけでなく，プログラミングを学ぶこともある．和歌山大学システム工学部では，後期に「情報処理ⅡA」,「情報処理ⅡB」を開講しており，プログラミングを学修させている．

2020 年度以降，新型コロナウイルスの影響で授業科目が遠隔で実施されることが多く，和歌山大学の情報処理の授業科目も遠隔での実施となった．遠隔授業には，オンデマンド型，ライブ配信型，双方向型の 3 つの形態が存在する．そのうち，和歌山大学の情報処理教育では，オンデマンド型を採用している．オンデマンド型は，学生が好きな時間に授業を受講できる利点がある．また，授業動画は何度でも視聴可能で，理解を深めるために繰り返し学習することができる．しかし，デメリットも存在する．例えば，上記の情報処理科目では，学生との補助的なコミュニケーション手段として Microsoft Teams を採用しているが，Teams を使ったコミュニケーションに慣れていない学生も多く，学生からの意見や質問を受け取りにくいことがある．その他には，インターネット環境が悪い地域に住む学生は，授業動画が快適に閲覧できない場合があり，学習の質に影響を与える可能性がある．

プログラミングの代表的な自習方法の 1 つに「写経型学習」[2]が挙げられる．写経型学習では，学習者がサンプルプログラムの写経を行い，実行し，その結果を確認する一連の流れを学習過程で体験する．著者らは，写経型プログラミングに基づく学習支援システム「LbTyping」を開発してきた[3][4][5]．タイピングゲーム形式のインターフェースを採用しており，学習者はシステムに登録されているソースコードを，打ち込みながら，プログラミングを学んでいく．なお，ミスタイプした文字は，赤く表示され，バックスペースキーで削除しないと次の文字入力に進めない仕組みになっている．しかし，写経型学習には問題点も存在する．理想的には，学習者がタイピングをこなしつつ，プログラミングに必要な知識を身につけることだが，タイピング自体が目的となりがちで，必要な知識の定着が不十分になることが多い．

プログラミングの学習方法について，本学の内外で近年，変革や，新たな試みがなされている．特に，オンライン実行環境の利用が盛んになっている．従来，プログラミングを学ぶ際に，自分のパソコンに開発環境を構築する必要があった．初学者にとっては，この環境構築が難解であり，学修の障壁となることが多かった．しかし，オンライン実行環境の普及により，このような問題が解決され，ブラウザさえあればプログラミングを始めることが可能になった．代表的なオンライン実行環境の例として，paiza.IO や Google Colaboratory が挙げられ，多くの大学で用いられている．

和歌山大学システム工学部における情報処理教育では，期末テストを実施せず，授業各回のレポートのみに基づき，成績評価が行われている．この成績評価方法には 2 つの問題点が存在する．1 つ目は，学生の理解度を総合的に把握するのが難しい点である．レポートは特定の課題に対する理解を示すものであり，コース全体を通じた総合的な理解や応用能力の評価には限界がある．2 つ目は，学生の学習過程における即時的なフィードバックを提供するのが難しい点である．レポートの提出と評価には時間がかかるため，学生が自身の理解度や間違いを早期に認識し，改善する機会が限られる．これにより，学習効果が阻害される可能性がある．

1.2 研究の目的

本研究の目的は，大学におけるプログラミング教育の諸問題への解決であり，その中でも，学生の即時的なフィードバックの提供と，総合的な理解度の評価方法の確立に焦点を当てる．この目的を達成するため，オンラインジャッジ[6][7][8]に着目した．オンラインジャッジは，学生が提出したプログラムを自動で採点し，即座に結果をフィードバックするシステムである．このシステムを採用することで，学生の理解度をリアルタイムで把握し，効果的な学習支援を提供することができると考えられる．

学習支援のためのオンラインジャッジシステムの開発と利用にあたって，対象とするプログラミング言語を明確に定める必要がある．本研究では，和歌山大学の情報処理教育で使われている C 言語と Python を対象言語として選定した．また，学習者が使用しているオンライン実行環境を考慮して，システムを設計することで，より効果的な学習体験が得られるようにした．

1.3 研究の成果

本研究では理解度テストの正誤判定を自動化するオンラインジャッジシステムを Web アプリケーションとして構築した．既発表のシステム[9]に対し改善を図った 2022 年度のシステムでは，C 言語を対象言語とし，オンライン実行環境として paiza.IO を使用した．2023 年度のシステム[10]では，Python を対象言語とし，オンライン実行環境として Google Colaboratory を使用した．その際，新たに開発した Google Chrome ブラウザの拡張機能を活用することで，Google Colaboratory からオンラインジャッジシステムへの提出が容易になり瞬時に結果が得られるようになっている．

† コア Core Corporation
‡ 和歌山大学 Wakayama University

本システムを使用して，2022～2023 年度の和歌山大学システム工学部「情報処理ⅡA」の科目において計 4 回の評価実験を行った．学習者の解答結果を集計し，学習者がどのような間違いをする傾向にあるか分析した．出力する文字列の間違いも見られたが，特にアルゴリズムをプログラムにうまく落とし込めていない誤答が目立った．分析結果から，本システムを利用することによって利用者の苦手な箇所を特定し，指導者が適切なアドバイスをすることを可能にした．

2. プログラミング学習支援

2.1 プログラミング学習とその支援

多くの大学で，プログラミング教育が実施されている．2016 年 11 月から 12 月にかけて，国内の全大学における情報教育状況に関する調査が行われ[1]，約 80%の大学で情報教育が行われていることがわかった．また，情報系科目を必修科目として取り入れている大学は約 60%であることもわかった．

従来，プログラミングを行う人は主に，専門家や職業プログラマのみであったが，近年，義務教育でプログラミング教育が導入されたことにより，児童がプログラミングを行う機会も増えてきた．この場合，プログラミングを行う目的は，ソフトウェア開発ではなく，論理的思考や自己表現などに重きが置かれる．

プログラミング学習をする際，様々な障壁を乗り越えながら進めなければいけない．市村ら[11]によると，大学で初めて習う学生の多くは，プログラミングに対する苦手意識が強いという現状がある．また，必修科目であるプログラミングの受講者は一般的に多く，教員と TA(ティーチングアシスタント)が協力して学生のサポートを行うことがよくあるが，分からないのに手を挙げない，何を質問すればいいか分からない学生には，対応できない．さらに，教員が想定しないポイントで多くの学生が躓いている場合にそのことを発見するのに時間がかかるといった問題が指摘されている．

2.2 和歌山大学システム工学部の情報処理教育

和歌山大学システム工学部では 2019 年度まで，前期に情報処理Ⅰ，後期に情報処理Ⅱが開講されていた．2020 年度からはクォーター制になり，情報処理ⅠA，情報処理ⅠB，情報処理ⅡA，情報処理ⅡB に分かれたが，内容はおおむね同じである．前期の情報処理ⅠA および情報処理ⅠB は，コンピューターを利用した効率的な情報の収集・整理・活用能力を磨く科目である．具体的には，ワープロソフトを用いてレポート作成を行うことや，プレゼンテーションソフトを用いてスライドを作成することなどを学ぶ．後期の情報処理ⅡA および情報処理ⅡB では，プログラミングを学ぶことになっている．2022 年度には，学生は C コース，Python コースのいずれかを選ぶことができた．どちらのコースでも，学生はまず，各プログラミング言語に特有の語（printf，%など）やその意味を学ぶことから始まる．並行して，プログラム作成にあたり必要な事項の考え方を講義や演習課題を通して学んでいく．ただし C コースは 2022 年度を限りに廃止され，2023 年度は Python のみとなった．

2.3 プログラミングの代表的な学習方法

2.3.1 写経型学習

写経型学習はプログラミングの学習方法として，岡本ら[2]により提唱されている方法であり，ワークブックも出版されている[12]．この学習方法では，まず学習者がサンプルプログラムを記載通りに打ち込んで（写経），ファイルに保存して実行し，結果を確認するという一連の流れを体験する．しかし，[2]で報告されているとおり，学習者がプログラムの実行する段階まで容易に到達することは少ない．タイピングミスによる文法エラーを発生してしまうからである．そのため，学習者はエラーの修正に追われるが，エラーメッセージは多くの場合，専門用語を含む英語で表示されるため，すぐには修正できない．また，文法エラーを解消できたとしても，実行できたが結果が違う，いわゆる論理エラーということもある．この場合，エラーメッセージは表示されないため，自力で間違い箇所を発見し，修正しなければならない．

また，写経型学習は単調な作業になりやすく，学習内容の定着が不十分になる可能性があるが，完全な知識の定着を目指すものではなく，あくまで初学者がプログラミングに慣れるために行う学習方法である．和歌山大学システム工学部の情報処理教育でも，情報処理ⅡA の C コースでは，この手法を採用しており，受講選択前にそのことを明示している．また，情報処理ⅡA の Python コースでは，写経型学習を用いるとは明言していないが，学生が課題に取り組む過程で，PDF 資料や講義動画に掲載されているサンプルコードを写経することもある．

2.3.2 PBL

PBL は「Problem Based Learning」の略であり，日本語に訳すと「問題解決型学習」となる[13]．PBL はアメリカの教育学者であるジョン・デューイによって提唱され，プログラミング以外の様々な分野で使われている学習方法である．PBL では自ら見つけた問題を解決していく能力を養うことを中心としている．IT 企業等における実際の開発手段と PBL の学習方法は似ているので，プログラミング学習に使われることが多いのも特徴である．授業で PBL を実施する際，教員が問題を提示し，その解決策を学生たちは自力で模索することになる．しかし，プログラミングを初めて習う初学者には難しすぎるという問題があり，PBL が使われることは少ない．

2.3.3 課題提出型学習

課題提出型学習は，特にオンライン授業でよく採用される方法で，プログラミングに限らず，多くの演習科目でこの方法が採用されることが多い．この方法では，学習者はまず講義動画や書籍などを通じてプログラミングを学ぶ．その後，理解を深めるために与えられた課題に取り組む．しかし，この学習方法にはいくつか問題点がある．まず，課題に対する直接的なフィードバックが得られないことが多く，学習者は自分で書いたプログラムが正しいかどうかを自分で検証する必要がある．次に，課題を提出すること自体が目的になりやすい．これは，学習者が単に課題を提出することのみに焦点を当て，深い理解や長期的な知識の定着をおろそかにする傾向があるからである．

最近では，課題を提出する際，メールや Teams などのコミュニケーションソフトウェアではなく，Learning Management System (LMS)が使われることが多い．LMS は，教育活動を管理するためのオンラインプラットフォームである．このシステムでは，授業資料の配布，課題の提出と評価，そして成績評価などが行える機能を搭載している．和歌山大学では，LMS として Moodle を採用しており，プログラミング科目のみならず，様々な科目で使用されている．

2.4 オンラインジャッジシステム

オンラインジャッジシステム[6]とは，多くのプログラミングに関する演習問題が掲載されており，オンラインでソースコードを自動採点してくれるシステムのことである．各問題にはテストケースが用意されており，提出されたソースコードの正誤と実効速度の計算をすばやく行ってくれるのが特徴である．有名なオンラインジャッジシステムの例として，AtCoder や Codeforces などが挙げられる．

オンラインジャッジシステム自体は学習支援を行うものではないが，本研究では，以下に述べる 2 つの理由で着目した．まず，提出したプログラムの正誤が，解答者に即座にフィードバックされる．このことは，課題提出型学習と比べて大きな利点である．次に，解答者が要件を満たすプログラムを提出して最終的に正解に至るまでに試行錯誤の体験が得られる．規模の大小を問わずソフトウェア開発においいても試行錯誤は不可欠であり，それを体験できるという点でオンラインジャッジシステムを用いた学習は効果があるのではないかと考えられる．

本研究では，情報処理ⅡA を受講した学生がどの程度プログラミングが身についているか確認するために，オンラインジャッジシステムを用いた理解度テストを行った．テスト実施後は学生の解答を分析し，正答率やエラーの傾向を明らかにし，講義内で適切な指導を行った．情報処理ⅡA で 2 回の理解度テストを実施しており，教育評価[14]においては総括的評価に位置付けられる．

2.5 関連研究およびサービス

2.5.1 paiza.IO

初学者がつまずきがちなポイントとして，プログラミング環境の構築が挙げられる．従来，コンパイルおよび実行のための環境構築にあたり，ソフトウェア（コンパイラ）のインストールや，コマンドラインインタフェースの利用などが必要であった．これらの問題はオンライン実行環境上でプログラミングをすることによって解決できる．

無料で利用可能なオンライン実行環境の 1 つに paiza.IO[15]がある．C 言語や Python（バージョン 2 および 3）を含む主要 24 言語に対応しており，ソースコード公開機能があるほか，他のサイトでのオンライン実行環境の埋め込みを認めている．画面例を図 1 に示す．

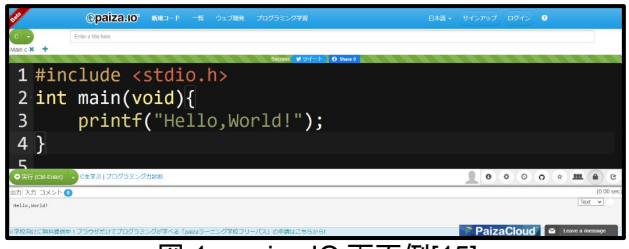

図 1　paiza.IO 画面例[15]

3 節で述べる，C 言語を対象としたオンラインジャッジシステムでは，問題出題画面に paiza.IO を埋め込んであり，学習者が環境構築を終えていなくても，ブラウザさえあればプログラミング学習に取り組むことができる．

2.5.2 Google Colaboratory

Google Colaboratory(以下，Colab と表記する)[16]は，Google 社が開発したブラウザで動作する Python 専用のオンライン実行環境である．paiza.IO と大きく違う点は，Jupyter ノートブック形式を採用しており，文章とコードを同時に含むドキュメントを作成できる点である．2023 年度の情報処理ⅡA および情報処理ⅡB では，Jupyter ノートブック形式のファイルを学生に配布し，課題に取り組ませている．画面例を図 2 に示す．

図 2　Google Colaboratory 画面例[16]

2.5.3 プログラミング初学者向けのオンラインジャッジシステム

狩野ら[17]は，プログラミング入門者向けのオンラインジャッジシステムによるプログラミング教育を提案した．このシステムでは，Web 上でコンパイル・実行する機能や，エラーメッセージの表示，プログラムを一行ごとに実行するデバッグ機能を備えている．このシステムでは初学者向けの易しい問題が採用されているのも特徴である．

2.5.4 意図的に埋め込まれたバグによる学習支援システム BugTutor

岩渕ら[18]は，意図的に埋め込まれたバグによるプログラミング学習支援システム BugTutor を開発し，学生に使用させた．具体的には，学習内容の説明文には正しい情報を与えるが，サンプルプログラムにバグを混ぜ込んだものを提示して学習を行わせている．誤情報だけでは学習者が間違ったプログラムを覚えてしまう可能性があるため，バグを埋め込んだサンプルプログラムを提示してから，15 分後に正しいプログラムが掲示されるようになっている．評価実験では，Python のプログラミング経験がない学生を対象に，BugTutor を利用する被験者 6 名と，BugTutor を利用しない被験者 6 名に分けて，対照実験を行い，t 検定により学習時間の有意性を示している．

3. 学習支援の Web アプリケーション

3.1 開発方針

3.1.1 先行システムの概要および問題点

既発表のオンラインジャッジシステム[9]について，概要を示したのち，機能拡張などの方針を述べる．当該システムでは，オンライン実行環境 paiza.IO を埋め込むことにより，1 画面でプログラム作成，動作確認，答案提出ができるようになっている．本システムを用いた評価実験を 2021 年度開講情報処理ⅡA 最終回に行った．「wakayama を 10 回表示するプログラムを作成せよ」という問題を出題し，正答・誤答ともに多様な解答が得られた．

このシステムには問題点が 2 つある．1 つ目が，入力に応じて出力が変化するプログラムの正誤判定ができない点である．例えば，「整数 N を入力に取り，N 回 wakayama を表示せよ」のような課題に対し適切に判定できない．2 つ目が，2023 年度情報処理ⅡA では，Google Colaboratory が推奨開発環境として採用されたため，paiza.IO をオンライン実行環境として用いるのでは，利用にあたり混乱が予想される点である．

3.1.2 C 言語のオンラインジャッジシステム

2022 年度に構築した C 言語のオンラインジャッジシステムは，入力に応じて出力が変化するプログラムの正誤判定を可能にした．また，問題解答画面に paiza.IO が埋め込まれており，ローカル開発環境がなくてもプログラミングに取り組むことができる．学生は paiza.IO でプログラムを完成させたあと，所定の欄に貼り付けを行い，提出ボタンを押す．本システムのサーバーが，学生番号およびコードを受け取り，サーバー内でコンパイルおよびプログラム実行を行う．いくつかのテストケースでプログラムが正しいかどうかを検証し正誤判定を行う．

3.1.3 Python のオンラインジャッジシステム

3.1.1 で述べた問題に関して，paiza.IO の代わりに，Google Colaboratory を iframe タグで埋め込むことや，Google Colaboratory に遷移するリンクを設置することで対処を試みた．しかし，前者に関しては，Google Colaboratory は，iframe タグによる埋め込みができないため，技術的に不可能なことがわかった．後者に関しては，技術的には簡単ではあるが，1 ページで完結するという本システムのメリットを捨ててしまうことになるため，リンクの設置は取りやめた．そこで，Google Colaboratory 上から簡単に提出できるように，Chrome 拡張機能の開発を行った．

2023 年度に構築した Python のオンラインジャッジシステムは，2022 年度に構築した C 言語のオンラインジャッジシステムのバックエンド部分を Python のプログラム判定ができるように改良して実現されている．

3.2 システム構成

本システムのサーバー部の構成は，[9]のものと同じである．HTML のレンダリング，およびデータベースとの通信を担うバックエンド部分に，Python の Web フレームワークである Flask を，データベースに PostgreSQL を採用した．また，開発環境を構築するために Docker を使用し，Flask コンテナ，PostgreSQL コンテナを作成した．C プログラムの動作確認は，Flask コンテナのプロセスから GCC を呼び出してコンパイルを行い，実行ファイルが生成されたなら，実行して入力を与え，出力を獲得する．

3.3 データ管理

本節では，使用するデータの管理をするために設計したデータベースについて説明する．用意したテーブルは問題(question)テーブルと提出(submit)テーブルである．問題テーブルは 1 問につき 1 レコードで，主キー(question_id)のほか，出題に関する様々な情報を格納する．提出テーブルは 1 回の解答につき 1 レコードで，解答の通し番号(response_id)のほか，提出者の学生番号(student_id)やソースコード(source)，を含む解答結果を格納し，4 節で説明する評価実験の際に分析した．なお，student_id は学生番号を表す文字列であり，主キー・外部キーのいずれでもないこととし，1 問に同一の学生が複数回提出したとき，提出テーブルの別々のレコードとして格納する．また正誤判定を行うため，問題テーブルは input と output，input2 と output2，input3 と output3 のカラムを持ち，それぞれ，正誤判定時の入力と出力の組となる．PostgreSQL の配列型や，JSON などのデータフォーマットにより，任意個の入力と出力の組を保存することも検討したが，組が多くなるとその都度実行するため判定に時間を要することから，それらを採用せず，1 回の提出に対するテスト実施は最大 3 回までとした．

3.4 システムの使用方法

3.4.1 C 言語のオンラインジャッジシステム

最初に学生に Moodle を通して，問題解答画面に遷移する URL を知らせておく．学生は，図 3 のようなページを閲覧して，問題文と出力例を読み，右のオンライン実行環境 paiza.IO で仕様を満たすプログラムを作成する．プログラムを実行し，出力例と同じになることが確認できたら，画面下にあるテキストボックスにプログラムをコピーし，学生番号を入力する．

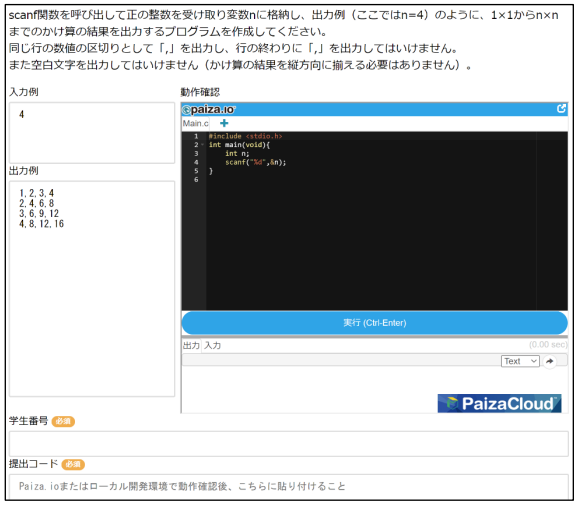

図 3　C 言語のオンラインジャッジシステム初期画面例

提出ボタンを押すと，図 4 のような小さなウィンドウが表示される．これはブラウザのアラート機能を利用した，ソースコードの正誤判定結果である．

図 4　正誤結果判定例

図 4 のうち「Accepted」は提出内容に応じてサーバーからブラウザに送られる正誤判定結果である．ここに表示される内容とその意味を表 1 に示す．また，図 4 のうち「(1)」は，データベースで管理される response_id の値に対応し，提出ごとに増える通し番号となっている．利用者は Moodle の課題の答案として，この番号を提出し，本システムを利用したことが証明できる．

表 1　正誤判定結果

表示	意味	正解かどうか
Accepted	出力が条件にあっている	正解
Wrong Answer	出力が条件にあっていない	不正解
Time Limit Exceeded	プログラムが 2 秒以内に終了しなかった	不正解
Runtime Error	実行時エラー	不正解
Compile Error	コンパイルに失敗した	不正解

主要な 4 つの正誤判定結果について説明する．「Hello」を 10 行出力するプログラムを C 言語で作成する問題とし，前処理指令や main 関数は省略している．

Accepted と判定されるプログラムの例を図 5(a)示す．このプログラムに対し，サーバーでコンパイルと実行が行われ，「Hello」を改行区切りで 10 回出力するのを獲得する．データベースの出力文字列と照合して同一であることを確認し，正答として扱う．

```
int i;
for (i = 0; i < 10; i++) {
  printf("Hello\n");
}
```
(a) Accepted

```
int i;
for (i = 0; i < 9; i++) {
  printf("Hello\n");
}
```
(b) Wrong Answer

```
while(1) {
  printf("Hello\n");
}
```
(c) Time Limit Exceeded

```
int i
for (i = 0; i < 10; i++) {
  printf("Hello\n");
}
```
(d) Compile Error

図 5　正誤判定のプログラム例（抜粋）

Wrong Answer と判定されるプログラムの例を図 5(b)に示す．このプログラムに対し，サーバーでコンパイルと実行が行われ，「Hello」を改行区切りで 9 回出力するのを獲得する．データベースの出力文字列と照合して同一でないことを確認し，誤答として扱う．

Time Limit Exceeded と判定されるプログラムの例を図 5(c)に示す．このプログラムに対し，サーバーでコンパイルと実行が行われ，「Hello」を改行区切りで無限に出力する．本システムでは，実行時間が 2 秒を超える実行については強制終了したのち，出力の正誤は問わずに，誤答として扱う．正答を出力したあとに何も出力せず，実行時間が 2 秒を超える場合にも，Time Limit Exceeded となる．

Compile Error と判定されるプログラムの例を図 5(d)に示す．このプログラムに対しては，コンパイル時にエラーが発生し，実行ファイルが作られず，誤答として扱う．

3.4.2 Python のオンラインジャッジシステム

あらかじめ，Google Colaboratory で理解度テスト用の問題ページを作成しておき，Moodle でそのページに遷移する URL を学生に知らせておく．学生は図 6 のように，課題文やヒントなどを見ながら，コード入力欄でプログラミングを行う．

図 6　Python のオンラインジャッジシステム初期画面例

プログラムが完成したら，ブラウザ右上にあるアクションメニューから，本システムの Chrome 拡張機能のボタンをクリックし，ポップアップを表示させる(図 7)．ポップアップ内のソースコードを取得するボタンを押すと，Google Colaboratory に入力されたソースコードが貼り付けられる．その状態で，学生番号を入力して，提出ボタンを押すと，図 4 と同様のアラートが表示される．

図 7　提出のためのポップアップ表示例

正誤判定については，言語が Python に異なる点を除いて，おおむね 3.4.1 で述べたものと同じである．実行に先立ち，python コマンドに-m py_compile オプションを指定して，提出されたソースコードに対してコンパイルを行っており，ここで例外が発生したら，入力を与えて実行することなく，Compile Error と判定している．

2023 年度の実施にあたっては，Chrome 拡張機能のインストールおよびこの機能を用いた提出を推奨したが，拡張機能を利用しない学生向けに授業科目の Moodle からも提出できるようにした．ただし，Moodle 提出の場合は，著

者らがその都度回収してシステムに登録し，正誤判定結果を答案提出に対する評価結果として登録したため，即時のフィードバックはできていない

3.5 教員解答情報閲覧機能

学生の解答状況を確認するためのページを作成した．表形式になっており，question_id，response_id，student_id，正誤結果，整形されたソースコードが，提出順で表になっており，教員は，このページを一覧し，ページ内検索などを行うことで，特徴的なソースコードを認識しやすくなっている．

4. 評価実験

4.1 評価の目的および方法

和歌山大学システム工学部の 2022，2023 年度「情報処理ⅡA」を受講しているシステム工学部学生を対象として，計 4 回の評価実験を行った．評価実験の実施概要を表 2 に示す．本評価実験を行う目的は，主に 2 つある．(1)授業内容の定着度を確認する．(2) 1 回の提出で正解しなかった学生が，どのような誤答を提出し，最終的に正解に至った（または至らなかった）のかを確認する．3.2 節で述べた構成のサーバーを研究室内に配置し，学内外からの HTTPS 通信により，システムが正常に動作していることが確認できた．

表 2　理解度テスト概要

実施回	実施日	使用言語	出題内容
2022 年度第 1 回	2022/11/24	C 言語	標準出力
2022 年度第 2 回	2022/12/1	C 言語	九九の表
2023 年度第 1 回	2023/11/16	Python	偶奇判定
2023 年度第 2 回	2023/11/30	Python	回文判定

2023 年度第 1 回理解度テストの実施にあたり，解説ページを作成し，学生に提供した．オンラインジャッジシステムの特色として，より厳しい基準で判定を行い、速やかに返答する点を挙げ，「より厳しい基準」については，授業のプログラミング課題と対比しながら詳しく説明した．課題文に入力・出力の例が書かれているとき，その通りに動作すれば完成と見なせるのに対し，オンラインジャッジシステムでは，問題文に書かれていない入力・出力の組を使用して検証し，それによって不正解(Wrong Answer)と判定することもある．正解(Accepted)でなければ，解答者に示して修正・再提出を促す．

また学生間のプログラミングスキルの違いや，オンラインジャッジは多くの学生に未知である点に配慮して，解説ページには 2 種類の想定ユーザーおよびシステム利用形態を記載した．プログラミングが得意な人には，腕試しの場として，一発合格を狙うことを，得意でないという人は，試行錯誤の場として，授業資料を見直しながら亀の歩みで Accepted を目指すことを，それぞれ示して動機付けを試みた．

4.2 評価結果

4.2.1 2022 年度第 1 回理解度テスト

2022 年度第 1 回理解度テストは，2022 年度情報処理ⅡA 第 7 回に実施した．出題した問題は，「Hello」「C」「Programming」を 3 行に分けて出力するプログラムを作成せよという問題である．出題意図は，printf 関数を用いたプログラミングが適切にできるかどうかである．実施結果を表 3 に示す．また，1 回の提出で正解しなかった人が，正解するまでに間違う回数の平均は 2.29 回であった．

表 3　2022 年度第 1 回理解度テスト集計結果

	人数(人)	割合(%)
解答者数	64	100.0
1 回で正解	45	70.3
最終的に正解	62	96.9
最終的に正解できず	2	3.1

4.2.2 2022 年度第 2 回理解度テスト

2022 年度第 2 回理解度テストは，2022 年度情報処理ⅡA 第 8 回に実施した．出題した問題は，整数 N を入力にとり，1×1 から N×N までの積（乗法九九の表）をコンマ区切りで出力するプログラムを作成せよという問題である．出題の目的は，二重ループを用いたプログラムが適切に書けるかどうかを確認することである．実施結果を，表 4 に示す．また，1 回の提出で正解しなかった人が，正解するまでに間違う回数の平均は 2.23 回であり，最大で，9 回の不正解を提出する学生がいた．

表 4　2022 年度第 2 回理解度テスト集計結果

	人数(人)	割合(%)
解答者数	83	100.0
1 回で正解	35	42.2
最終的に正解	69	83.1
最終的に正解できず	14	16.9

4.2.3 2023 年度第 1 回理解度テスト

2023 年度第 1 回理解度テストは，2023 年度情報処理ⅡA 第 7 回に実施した．出題した問題は，整数 N を入力にとり，N が偶数なら，「Even」，そうでないなら，「Odd」を出力するプログラムを作成せよという問題である．出題の目的は，偶奇判定は，if 文を用いて解答可能であることを問うためである．実施結果を，表 5 に示す．また，1 回の提出で正解しなかった人が，正解するまでに間違う回数の平均は 1.4 回であった．

表 5　2023 年度第 1 回理解度テスト集計結果

	人数(人)	割合(%)
解答者数	82	100.0
1 回で正解	50	61.0
最終的に正解	60	73.2
最終的に正解できず	22	26.8

4.2.4 2023 年度第 2 回理解度テスト

2023 年度第 2 回理解度テストは，2023 年度情報処理ⅡA 第 8 回に実施した．出題した問題は，入力された文字列が回文（前から読んでも後ろから読んでも同じ文字列）なら「Yes」を，そうでないなら「No」を出力するプログラムを作成せよという問題である．出題目的は，for 文や if 文を適切に使用できるか，および回文判定のアルゴリズムを適切に構成できるかを評価することである．実施結果を，表 6 に示す．また，1 回の提出で正解しなかった人が，正解するまでに間違う回数の平均は 2.1 回であり，最大で 15 回の不正解を提出する学生がいた．

表 6　2023 年度第 2 回理解度テスト集計結果

	人数(人)	割合(%)
解答者数	40	100.0
1 回で正解	24	60.0
最終的に正解	30	75.0
最終的に正解できず	10	25.0

4.3 考察および課題

4.3.1 2022 年度第 1 回理解度テスト

2022 年度第 1 回理解度テストの正答例（#include で始まる行や main 関数は省略している．以下同じ）について説明する．本問題の解答方針として，printf 関数を 3 回使う，図 8 のような解答のほか，printf 関数を 1 回だけ使用し，文字列の中で複数回改行をするような解答が考えられる．後者の解答は，変わった書き方ではあるが，写経型学習支援システム LbTyping のタイピング課題で，改行を表す制御文字「\n」を文字列中で複数回使用するものが存在したため，この方針の解答も登録されたと考えられる．

```
printf("Hello\n");
printf("C\n");
printf("Programming\n");
```

図 8　2022 年度第 1 回理解度テスト正答例(抜粋)

誤答の傾向としては，出力する語の先頭が大文字になっていない答案や，Programming のスペルミスをしている答案があった．本システムでは，文字列の完全一致が求められ，注意書きにも大文字・小文字の区別をする旨を書いた．一度は誤答を提出してしまっても，その後の提出で出力ミスを修正して正解に至った学生が何人か確認できた．

4.3.2 2022 年度第 2 回理解度テスト

2022 年度第 1 回理解度テストの正答例について説明する．本問題は，九九の表を出力するだけでなく，コンマの出力や改行を行うタイミングを考慮して解答する必要がある．正答例を図 9(a)に示す．

```
for (i = 1; i <= n; i++) {
  for (j = 1; j <= n; j++) {
    if (j == n) {
      printf("%d\n", i * j);
    } else {
      printf("%d, ", i * j);
    }
  }
}
```

(a) 正答例

```
for (i = 1; i <= n; i++) {
  for (j = 1; j <= n; j++) {
    printf("%d,", i * j);
  }
  printf("\n");
}
```

(b) 誤答例

図 9　2022 年度第 2 回理解度テスト解答例(抜粋)

この問題の難しさは，二重ループの使用と出力フォーマットを厳格に守らなければならない点にある．前者に関しては，ループ変数を 2 つ使用しなければならないが，同じループ変数を使用してしまい，無限ループに至ることもある．後者に関しては，各数値の後にコンマを配置する必要があるが，行の最後にはコンマをつけてはならない．そのために，ループ内で現在の列の位置をチェックし，最後の列ならコンマを打たずに改行することをプログラムとして表現する必要がある．特に，初学者は後者の実装がうまくいかず，図 9(b)のような誤答例が登録された．このソースコードを実行すると，各行の末尾にコンマがつくため，不正解である．

4.3.3 2023 年度第 1 回理解度テスト

2023 年度第 1 回理解度テストの正答例について説明する．本問題は，入力された整数の偶奇判定を行う際，2 で割った余りについて着目し，「if N % 2 == 0:」のように条件式を書くことが重要である．

しかし，剰余演算子「%」の代わりに，商を表す演算子「/」や「//」を用いた答案が登録された．「if N / 2 == 0:」のように書いたらは，「N を 2 で割った結果が 0 であるか」を意味し，N が 0 でない限り False と評価されてしまう．

4.3.4 2023 年度第 2 回理解度テスト

2023 年度第 2 回理解度テストの正答例について説明する．本問題は，文字列が回文なら，Yes，そうでないなら，No を出力せよという問題である．正答例を図 10 に示す．

```
def palindrome(word):
    length = len(word)
    for i in range(length // 2):
        if word[i] != word[length - 1 - i]:
            return "No"
    return "Yes"
```

図 10　2023 年度第 1 回理解度テスト正答例(抜粋)

本問題を理解度テストとして実施する際，ページ内に 5 項目からなるヒントを掲載した．このヒントは，図 10 のようなプログラムを作成する上で参考になる．

この課題で正答を得る際の難しさは，主に 3 つある．1 つ目が，文字列 word の後ろから i 番目は，word の長さを length としたとき，「word[length - 1 - i]」と表現できる点．2 つ目が，ループの範囲は，「length // 2」または「length」である点．3 つ目が，if 文を使用して文字が一致するかチェックし，一致しない場合は即座に「No」を返し，ループを抜けたら「Yes」を返す流れがわかりにくい点である．

誤答例としては，図 10 のうち「word[length – 1 - i]」が「word[length - i]」となっているものが登録された．この場合には i = 0 のとき，word[length]を評価することになり，IndexError を引き起こしてしまう．

4.3.5 開発したシステムの教育的効果

本研究で開発したオンラインジャッジシステムの教育的効果は主に 2 つある．1 つ目は，自己学習の促進である．評価実験の結果から，学生は 1 回の提出で正解に至らなくても，複数回の試行を通じて正解にたどり着く経験をしていることが明らかになった．規模の大小を問わずソフトウェア開発において，このような試行錯誤のプロセスは非常に重要である．2 つ目は，応用力の向上である．評価実験では，九九の表や回文判定といったプログラミング課題を通じて，授業課題よりも難易度が高い問題を出題した．基礎的なプログラミング事項を把握するだけでなく，問題を分析し，適切なアルゴリズムを設計し，それをソースコードに落とし込む応用力が求められる．1 回の提出で正解できた学生は少なく，授業課題だけでは満足できない学生にとって，骨のある問題であったと考えられる．

4.3.6 今後の課題

理解度テストで出題した問題の妥当性を検証する際に，年度ごとに実施された第 1 回と第 2 回のテスト間での難易度の差に関して注目する．2022 年度，2023 年度共に，第 1 回の理解度テストは，標準出力や if 文といったプログラミングにおける基礎的な項目を扱っていた．そのため，1 回で正解する学生は多く，何回か間違えても最終的に正解に至る学生がほとんどであった．

第 2 回理解度テストは，for 文を用いて乗法九九の表を出力する問題や回文判定を行う応用力を問う問題を出題した(九九の表を出力するプログラム自体は授業で学習済みで，そのプログラムでは 1×1 から 9×9 までに固定されており，数値の区切りは空白文字で，改行前の空白の有無は問わなかった)．そのため，1 回で正解する学生は少なく，試行錯誤を行っても正解に至らない学生が多くなった．

今回の評価実験では，1 つの授業科目につき 2 回，計 4 回の実施のみであったが，毎回の授業や自習・復習用として，オンラインジャッジを使ってもらう場合に気をつけるべきことが 2 点ある．1 つ目は，授業内容とリンクした問題を採用することである．授業内容に密接に関係した問題を解くことで，学生は自分の学んだ内容が定着しているかを簡単に確認することができる．2 つ目は，難易度を徐々に上げていくことである．評価実験では，年に 2 回の実施であったため，第 1 回と第 2 回で大きな難易度の差が生まれてしまった．授業ごとの実施であれば，徐々に難易度を上げることが可能で，オンラインジャッジを利用する学生も取り組みやすいのではないかと考えられる．

5. おわりに

本研究では，理解度テストの採点を自動化するオンラインジャッジシステムの構築を行った．提出されたソースコードを分析することによって，初学者が犯しやすいエラーやプログラミングのつまずきどころを明確にした．2022 年度には，既発表のオンラインジャッジシステムを改良し，C 言語用の理解度テストの内容見直しも同時に行った．2023 年度には，学習言語が Python に変更されたので，サーバー側で Python プログラムの正誤判定ができるように改良した．また，オンライン実行環境である Google Colaboratory から，オンラインジャッジシステムに解答提出が容易にできる Chrome 拡張機能を開発した．2022 年度および 2023 年度の和歌山大学システム工学部 1 年次向け授業で，本システムを利用して評価実験を行ったところ，正答・誤答ともに多様な答案が得られた．

今後の課題としては，繰り返し間違った答案を提出してしまい，なかなか正解に至らない学生に対する支援を挙げることができる．評価実験では，講義で配布した動画や PDF 資料を見ながら解いても良いと周知したが，それだけではプログラミングが苦手な学生に対しては不十分でああった．この問題に対処するために，オンラインジャッジシステムに学習支援機能の追加が必要である．例えば，提出されたソースコードの内容に応じて，正解になるためのヒントや，関連する授業の動画や資料へのリンクを提示する機能である．これらの機能をオンラインジャッジシステムに組み込むことで，プログラミングが苦手な学生に対しても，より効果的な学習支援を提供することができると考えられる．

謝辞

本研究は JSPS 科研費 19K12267，22K12291 の助成を受けたものです．

参考文献

[1] 高橋尚子：“国内 750 大学の調査から見えてきた情報学教育の現状：-（3）一般情報教育編-”, 情報処理, Vol.58, No.6, pp.526-530 (2017).
[2] 岡本雅子,喜多一: “プログラミングの「写経型学習」における初学者のつまずきの類型化とその考察”, パイデイア：滋賀大学教育学部附属教育実践総合センター紀要, 第 22 巻, pp. 49-53 (2014).
[3] takehiko/LbTyping: 写経型学習のための Web アプリケーション. https://github.com/takehiko/LbTyping（2024 年 6 月 13 日参照）.
[4] 村川猛彦, 田中和季: “タイピングによるプログラミング学習のコンテンツ追加および授業実践”, 情報処理学会第 85 回全国大会, 1H-02, 第 4 分冊, pp.405-406 (2023).
[5] Murakawa, T.: “LbTyping: A web application for programming learning by typing”, The Fourteenth International Conference on Information, Intelligence, Systems and Applications (IISA 2023), Volos, Greece (2023).
[6] 渡部有隆: “オンラインジャッジの開発と運用 -Aizu Online Judge-“, 情報処理, Vol.56, No.10, pp.998-1005 (2015).
[7] Wasik, S., Antczak, M., Badura, J., Laskowski, A., and Sternal, T.: “A Survey on Online Judge Systems and Their Applications”, ACM Computing Surveys, Vol.51, Issue 1, Article No.3, pp.1-34 (2018).
[8] Watanobe, Y., Rahman, M. M., Matsumoto, T., Rage, U. K, and Ravikumar, P.: “Online Judge System: Requirements, Architecture, and Experiences”, International Journal of Software Engineering and Knowledge Engineering, Vol.32 No.06, pp.917-946 (2022).
[9] 田中和季, 村川猛彦: “プログラミング学習支援のためのオンラインジャッジシステムの構築”, 情報知識学会誌, Vol.32, No.2, pp.307-312 (2022).
[10] 田中和季, 村川猛彦: “Python 学習支援のためのオンラインジャッジシステムの構築”, 2024 年電子情報通信学会総合大会, D-15-05 (2024).
[11] 市村哲, 梶並知記, 平野洋行: “プログラミング演習授業における学習状況把握支援の試み”, 情報処理学会論文誌, Vol.54, No.12, pp.2518-2527 (2013).
[12] 喜多一, 岡本雅子, 藤岡健史, 吉川直人: “写経型学習による C 言語プログラミングワークブック”, 共立出版 (2015).
[13] 内田奈津子, 久野靖, 中山泰一: “PBL によるプログラミング入門科目の提案: 一般情報教育における入門カリキュラムの構築”, 情報処理学会論文誌, Vol.62, No.7, pp.1393-1414 (2021).
[14] 梶田叡一: “教育評価”, 第 2 版補訂 2 版, 有斐閣 (2010).
[15] ブラウザでプログラミング・実行ができる「オンライン実行環境」| paiza.IO. https://paiza.io/ja (2024 年 6 月 13 日参照).
[16] Colaboratory へようこそ - Colab https://colab.research.google.com/?hl=ja (2024 年 6 月 13 日参照).
[17] 猪狩知也，速水治夫：“プログラミング入門者向けのオンラインジャッジシステムによるプログラミング学習支援”, マルチメディア、分散協調とモバイルシンポジウム 2014 論文集 (2014).
[18] 岩渕悠太, 高島健太郎, 西本一志: “意図的に埋め込まれたバグによるプログラミング学習支援”, 情報処理学会研究報告ヒューマンコンピュータインタラクション（HCI）, Vol.2020-HCI-187, No.12, pp.1-6 (2020).