令和５年度　文部科学省
専修学校における先端技術利活用実証研究

遠隔教育における
プログラミング実習モデルの開発事業

事業成果報告書

２０２４年２月

学校法人河原学園
河原電子ビジネス専門学校

本報告書は、文部科学省の教育推進事業委託費による委託事業として、学校法人河原学園 河原電子ビジネス専門学校が実施した令和5年度「専修学校における先端技術利活用実証研究」の成果をとりまとめたものです。

目次

1. 本事業の概要

1.1. 事業の趣旨と目的

近年、ソフトウェア技術の発展によって、Ｗｅｂ分野ではオンラインショッピング、ＡＩ分野ではＡＩ搭載家電等、新しいサービスの世界的な普及がもたらされている。このような趨勢を背景に、ソフトウェア技術の根幹をなすプログラミング教育には、さらなる質向上が求められてきた。しかし、専門学校のプログラミング実習では、以前より、プログラミングが苦手な学生の授業中フォローが難しい、ソフトウェア品質に関する指導が手薄という大きな課題が存在している。

さらに、昨今の新型コロナウィルス感染拡大に伴う遠隔授業によって、これらの問題は、①個々の学生の学習状況の把握が困難、②学生の個別指導が困難、③教員の負荷増大により、ますますソフトウェア品質に関する指導が困難、④学生の学習意欲の維持が困難という事態にまで拡大している。

本事業では４つのＡＩ機能（①課題に応じて模範となるコードを提示するＡＩ、②学生の個性に応じて動機づけるＡＩ、③不正解の原因・誤り箇所を推定するＡＩ、④採点を完全に自動化するＡＩ）を搭載したプログラミング実習システムとテキスト教材を導入することで、遠隔授業下で課題提示から評価指導（ソフトウェア品質指導含む）までをカバーするプログラミング実習モデルを構築し、上記問題の解決と質向上の対応を図る。

1.2. 事業の背景

1.2.1. プログラミング教育の重要性

近年、IT 技術を基礎とする先端技術が急速な発展を見せている。とりわけソフトウェア技術の高度化はめざましく、とくにWebおよびAI関連分野ではソフトウェア技術による生産性向上とサービス拡大が著しい。情報社会の進展に伴い、これらの発展は産業構造・就業構造にも変化をもたらし今後も拡大傾向が見込まれる。

こうしたソフトウェア技術の成長を根底から支えるのがプログラミング人材である。IT 企業に対して職種別の重要度を尋ねた調査によると、「エンジニア／プログラマー」に関して「いる／非常に重要」＋「いる／ある程度重要」の回答が対象企業の 70%を超過している（図１：独立行政法人情報処理推進機構 社会基盤センター編『IT 人材白書 2020』、2020 年）。プログラミングスキルを持つプログラミング人材は、IT 企業にとって依然、もっとも重要な職種とみなされていることがわかる。そして、専門学校のプログラミング教育が、大学の情報系学科の教育と並んでプログラミング人材育成の一翼を担っている。

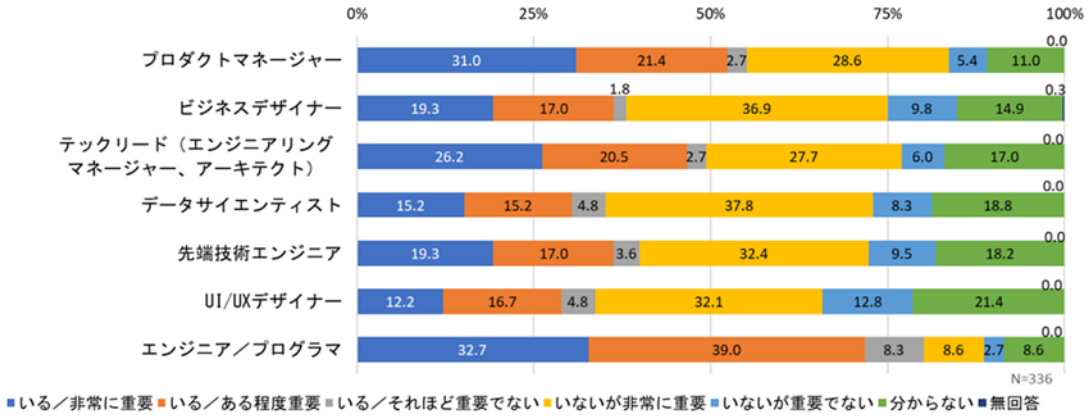


図 1. IT 企業における IT 人材種ごとの重要性の相違

1.2.2. プログラミング実習授業の課題

専門学校のプログラミング教育は、従来、単に言語仕様や文法に関する知識を伝授するだけでなく、その応用であるプログラミング実習を通して実装能力の育成に努めてきた。しかし、そこには二つの問題が存在し、プログラミング教育の長年の課題となっている。

ひとつは、プログラミングが苦手な学生のフォローの問題である。プログラミング実習中、教員は個々の学生の作成過程のプログラムを適宜、評価・指導しなければならないが、30 名前後のクラスとなると苦手学生のフォローを十分には行えない。実習授業に複数名の教員を配置するか、少人数クラス制をとれれば自ずと解決できるが、それほど人員潤沢な専門学校は少ない。

もうひとつの問題は、プログラム品質に関する指導の問題である。実習授業では、時間的制約から、学生の作成したプログラムが機能仕様通りに動作しさえすればよいという指導に陥りがちである。ひとりひとりの学生に（機能性品質以外の）ソフトウェア品質まで立ち入った評価と指導を行うことは実習授業の時間内では難しい※。

※実は、この問題には別な側面も存在する。専門学校の情報教育においてソフトウェア品質に関する指導が手薄になりがちな背景には、専門学校ではソフトウェア工学の教育があまり普及していないという事情もある。たとえば、基本情報技術者試験対策が、学生がソフトウェア工学に触れる唯一の機会であったり、教員にソフトウェア工学、ソフトウェア品質に関する知識が乏しかったりすることがある。

1.2.3. 遠隔授業下での問題拡大

どちらの問題も、これまでは放課後や空き時間の指導（場合によっては個別指導）で補うというのが一般的であったが、翌日以降の授業準備も抱える教員にとっては大きな負担を抱えながらの苦肉の策である。しかし、新型コロナウィルス蔓延による遠隔授業への移行を受けて、実習時間中の指導も放課後フォローも同様に困難となったばかりか、以下のように問題が多面的に拡大してしまった。

①遠隔環境下で、実習時間中に机間巡回で個々の学生の取組状況を把握することが困難になってしまった。

②遠隔環境下で、プログラム作成につまずく学生への個別対応がますます困難になってしまった。

③①の問題により、頻回のテストによる学生の理解度確認の必要性が高まったが、プログラミング技術を問うテストは作成だけでなく採点にも手がかかり、教員負荷が大きい。放課後フォローやプログラム品質に関する指導がますます困難になる。

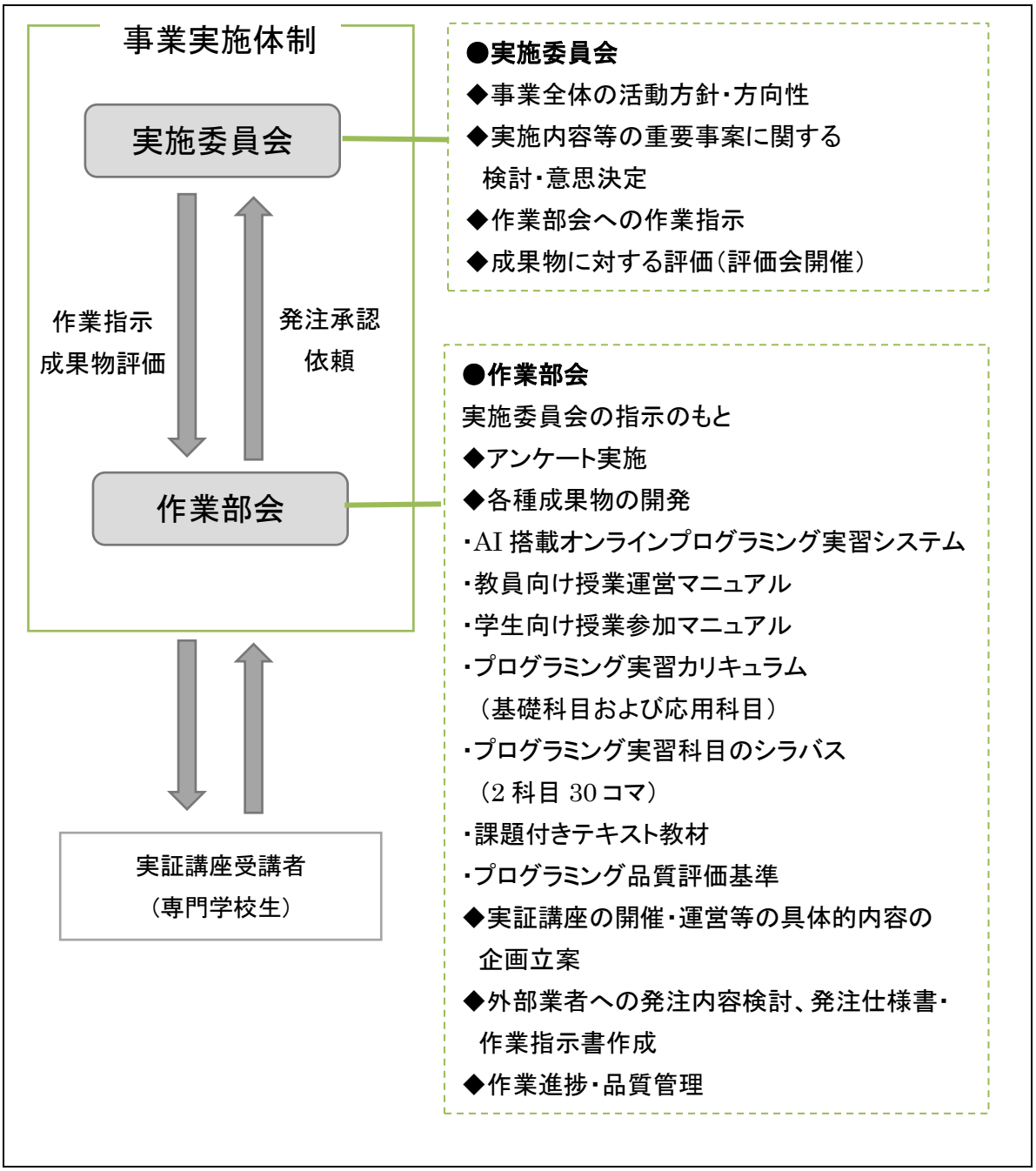
④教室と比較して開放的な学習環境（学生の自宅等）のため、学生の集中力の維持が困難になってしまった。

遠隔授業におけるプログラミング実習の問題	
課題①	課題への取組状況等、**個々の学生の学習状況の把握が困難**
課題②	プログラム作成につまずく**学生の個別指導が困難**
課題③	遠隔環境下では、プログラミング課題とその達成度評価の重要性が増し、**教員の学生指導負荷が増大**
課題④	教室と比較して開放的な学習環境（学生の自宅等）であるため、**学生の学習意欲の維持が困難**

遠隔授業下でも、対面授業と同等もしくは、より少ない教員負担で、学生の理解度を把握し、適切なフィードバックを行うことが可能なプログラミング実習モデルの構築が求められる。

1.3. 事業の運営・実施体制

1.3.1. 委員会・作業部会からなる組織体制



1.3.2. 委員会・作業部会の人員構成

◆実施委員会の構成員（委員）

氏名		所属・職名	役割等	都道府県名
1	河原成紀	学校法人河原学園　理事長	委員長・ 統括・管理	愛媛県
2	芦澤昌彦	学校法人河原学園　教務部長	運営・管理・ 開発・評価	愛媛県
3	中村亮	学校法人河原学園 河原電子ビジネス専門学校 教頭	開発・実証	東京都
4	井坂昭司	学校法人小山学園 東京テクニカルカレッジ 副校長	開発・実証	東京都
5	伊藤泰宏	学校法人岩崎学園 情報科学専門学校 教務部 次長	開発・実証	神奈川県
6	柿本圭介	学校法人岩崎学園 情報セキュリティ大学院大学 客員研究員	開発・実証	神奈川県
7	鈴木塁	学校法人静岡理工科大学　静岡産業技術専門学校 教諭	開発・実証	静岡県
8	本田澄	大阪工業大学 情報科学部 ソフトウェア工学研究者	開発・実証	大阪府
9	齋藤大輔	高千穂大学 経営学部　准教授	開発・実証	東京都
10	松浦達也	株式会社いよぎんコンピュータサービス 次長	開発・評価	愛媛県
11	影浦義丈	株式会社エイチビーソフトスタジオ 代表取締役	開発・評価	愛媛県
12	國廣尚良	株式会社アイ・エヌ・エス　経営企画室	開発・評価	愛媛県
13	赤松正教	一般社団法人 愛媛ニュービジネス協議会 副会長	開発・評価	愛媛県
14	赤松民康	アカマツ株式会社　代表取締役社長	評価	愛媛県
15	廣井久典	愛媛県経済労働部産業支援局　産業人材課長	評価	愛媛県

◆作業部会の構成員（委員）

氏名		所属・職名	役割等	都道府県名
1	芦澤昌彦	学校法人河原学園　教務部長	運営・管理・開発・評価	愛媛県
2	二宮竜也	学校法人河原学園　システム部職員	開発・実証	愛媛県
3	槇裕美	学校法人河原学園	運営・管理	愛媛県
4	井坂昭司	学校法人小山学園 東京テクニカルカレッジ 副校長	開発・実証	東京都
5	伊藤泰宏	学校法人岩崎学園 情報科学専門学校 教務部 次長	開発・実証	神奈川県
6	柿本圭介	学校法人岩崎学園 情報セキュリティ大学院大学 客員研究員	開発・実証	神奈川県
7	鈴木塁	学校法人静岡理工科大学 静岡産業技術専門学校 教諭	開発・実証	静岡県
8	本田澄	大阪工業大学 情報科学部 ソフトウェア工学研究者	開発・実証	大阪府
9	齋藤大輔	高千穂大学 経営学部　准教授	開発・実証	東京都

1.3.3. 各機関の役割・協力事項

○教育機関

・遠隔授業環境下でのプログラミング教育に関する課題の共有
・アンケート調査（専門学校向け、IT 企業向け）項目作成
・AI 搭載オンラインプログラミング実習システムの仕様策定
・プログラミング実習カリキュラム案の作成
・プログラミング実習科目のシラバス案の作成
・教員向け、学生向けマニュアル案の作成
・課題付きテキスト教材案の作成
・実証方法の検討
・実証講座への協力（参加する学生の募集、機材設置・会場提供）

○研究者

・ソフトウェア工学研究者の立場から AI 搭載オンラインプログラミング実習システムの仕様とプログラミング品質評価基準に対する助言
・プログラミング教育研究者の立場から、マニュアル案、テキスト、教材案に対する助言

○企業（IT 企業、AI 関連企業）

・アンケート調査（IT 企業向け）項目作成
・プログラミング人材受け入れの立場からの提案

・プログラミング実習カリキュラム案に対する評価と助言
・プログラミング実習科目のシラバス案に対する評価と助言
・テキスト教材に掲載する課題内容の検討
・プログラミング品質評価基準案に対する評価と助言
・AI 搭載オンラインプログラミング実習システムの仕様の検討

〇業界団体
・地域のプログラミング人材の需要と動向に関する情報提供
・実証方法の検討
・本事業成果物の地域への普及協力

〇行政機関
・地域プログラミング人材育成の立場から成果物の方向性の検討
・地域プログラミング人材育成の立場から実証方法の検討

1.3.4. 実施委員会の開催

◆第 1 回実施委員会

日　時	2023 年 8 月 7 日（月）16：00～17：30
場　所	遠隔：Microsoft Teams（リモートワークのためのコラボレーションツール）
出席者	芦澤昌彦（学校法人河原学園）、廣井久典（愛媛県経済労働部産業支援局）、赤松民康（アカマツ株式会社）、赤松正教（一般社団法人愛媛ニュービジネス協議会）、影浦義丈（株式会社エイチビーソフトスタジオ）、松浦達也（株式会社いよぎんコンピュータサービス）、國廣尚良（株式会社アイ・エヌ・エス）、齋藤大輔（高千穂大学経営学部）、本田澄（大阪工業大学情報科学部）、鈴木塁（学校法人静岡理工科大学静岡産業技術専門学校）、井坂昭司（学校法人小山学園東京テクニカルカレッジ）、伊藤泰宏（学校法人岩崎学園情報科学専門学校教務部）、二宮竜也（学校法人河原学園）、槇裕美（学校法人河原学園）
議　題	１．挨拶 ２．進捗報告 ３．意見交換 ４．今後のスケジュール ５．事後連絡
配布資料	資料１　課題点リスト 資料２　各種発注仕様書 資料３　合同遠隔授業計画書

◆第 2 回実施委員会

日　時	2023 年 10 月 24 日（火）16：00～17：30
場　所	遠隔：Microsoft Teams（リモートワークのためのコラボレーションツール）
出席者	河原成紀（学校法人河原学園）、芦澤昌彦（学校法人河原学園）、廣井久典（愛媛県経済労働部産業支援局）、影浦義丈（株式会社エイチビーソフトスタジオ）、松浦達也（株式会社いよぎんコンピュータサービス）、國廣尚良（株式会社アイ・エヌ・エス）、齋藤大輔（高千穂大学経営学部）、鈴木塁（学校法人静岡理工科大学静岡産業技術専門学校）、井坂昭司（学校法人小山学園東京テクニカルカレッジ）、二宮竜也（学校法人河原学園）、槇裕美（学校法人河原学園）
議　題	１．委員長挨拶 ２．進捗報告 ３．意見交換 ４．今後のスケジュール ５．事務連絡
配布資料	資料１　課題・意見リスト 資料２　実証講座授業運営案

◆第 3 回実施委員会

日　時	2024 年 2 月 2 日（金）16：00～17：30
場　所	対面：学校法人河原学園本部 遠隔：Microsoft Teams（リモートワークのためのコラボレーションツール）
出席者	河原成紀（学校法人河原学園）、芦澤昌彦（学校法人河原学園）、影浦義丈（株式会社エイチビーソフトスタジオ）、松浦達也（株式会社いよぎんコンピュータサービス）、國廣尚良（株式会社アイ・エヌ・エス）、齋藤大輔（高千穂大学経営学部）、本田澄（大阪工業大学情報科学部）、鈴木塁（学校法人静岡理工科大学静岡産業技術専門学校）、井坂昭司（学校法人小山学園東京テクニカルカレッジ）、伊藤泰宏（学校法人岩崎学園情報科学専門学校）、二宮竜也（学校法人河原学園）、槇裕美（学校法人河原学園）
議　題	１．委員長挨拶 ２．成果報告 ３．意見交換 ４．今後のスケジュール ５．事務連絡
配布資料	資料１　実証講座報告書 資料２　課題付きテキスト教材 資料３　教員向け授業運営マニュアル 資料４　学生向け授業参加マニュアル

1.3.5. 作業部会の開催

◆第 1 回作業部会

日　時	2023 年 7 月 28 日（金）16：00～17：30
場　所	対面：学校法人小山学園　専門学校東京テクニカルカレッジ 遠隔：Microsoft Teams（リモートワークのためのコラボレーションツール）
出席者	芦澤昌彦（学校法人河原学園）、齋藤大輔（高千穂大学経営学部）、本田澄（大阪工業大学情報科学部）、鈴木塁（学校法人静岡理工科大学静岡産業技術専門学校）、井坂昭司（学校法人小山学園東京テクニカルカレッジ）、伊藤泰宏（学校法人岩崎学園情報科学専門学校）、二宮竜也（学校法人河原学園）、槇裕美（学校法人河原学園）
議　題	１．挨拶 ２．意見交換 ３．今後のスケジュール ４．事務連絡
配布資料	資料１　前年度課題点 資料２　各種発注仕様書 資料３　マニュアル構成案 資料４　合同遠隔授業計画書

◆第 2 回作業部会

日　時	2023 年 10 月 16 日（月）16：00～17：30
場　所	対面：学校法人小山学園　専門学校東京テクニカルカレッジ 遠隔：Microsoft Teams（リモートワークのためのコラボレーションツール）
出席者	芦澤昌彦（学校法人河原学園）、齋藤大輔（高千穂大学経営学部）、本田澄（大阪工業大学情報科学部）、鈴木塁（学校法人静岡理工科大学静岡産業技術専門学校）、井坂昭司（学校法人小山学園東京テクニカルカレッジ）、伊藤泰宏（学校法人岩崎学園情報科学専門学校教務部）、柿本圭介（学校法人岩崎学園情報セキュリティ大学院大学）、二宮竜也（学校法人河原学園）、槇裕美（学校法人河原学園）
議　題	１．挨拶 ２．意見交換 ３．今後のスケジュール ４．事務連絡
配布資料	資料１　課題・意見リスト 資料２　実証講座授業運営検討資料

◆第 3 回作業部会

日　時	2024 年 1 月 19 日（金）16：00～17：30
場　所	対面：学校法人小山学園　専門学校東京テクニカルカレッジ 遠隔：Microsoft Teams（リモートワークのためのコラボレーションツール）
出席者	芦澤昌彦（学校法人河原学園）、齋藤大輔（高千穂大学経営学部）、本田澄（大阪工業大学情報科学部)、鈴木塁（学校法人静岡理工科大学静岡産業技術専門学校）、井坂昭司（学校法人小山学園東京テクニカルカレッジ）、伊藤泰宏（学校法人岩崎学園情報科学専門学校教務部）、二宮竜也（学校法人河原学園）、槇裕美（学校法人河原学園）
議　題	１．挨拶 ２．意見交換 ３．今後のスケジュール ４．事務連絡
配布資料	資料１　実証講座報告書

2. 事業活動の内容

2.1. 遠隔教育の導入方策とそのモデル化の概要

2.1.1. 課題解決のための４つの AI

遠隔環境下で拡大されたプログラミング実習授業の問題を解決するため、学生がオンラインで提出するプログラムを自動採点・評価するシステム（オンラインジャッジシステム）に、以下で述べる４つの AI を組み込んだ新しいシステム（AI 搭載オンラインプログラミング実習システム）を導入した。

オンラインジャッジ自体は、近年、民間企業によるプログラマーの採用活動の一環として導入が進んでおり、国内では AtCoder 社（https://atcoder.jp/）が開発・運営するオンラインジャッジシステムが有名である。しかし、既存のオンラインジャッジシステムは、教育目的よりも競技や採否判断を目的として開発されており、会津大学などいくつかの大学での採用事例はあるものの、初学者がプログラミングを学ぶ上で適した環境とは言い難い。そこで、本事業では、初学者がプログラミングを学ぶ用途で、かつ、遠隔授業での利用に適するという二つの要件を踏まえ、課題に応じて模範となるコードを提示する AI、不正解の原因・誤り箇所を推定する AI、学生の個性に応じて動機づける AI、採点を完全に自動化する AI の 4 つの AI 技術を新たに組み込んだオンライン実習システムを開発した。

2.1.2. 本事業の AI 搭載オンラインプログラミング実習システムの特長

①AI 技術 1：課題に応じて模範となるコードを提示する AI

既存のオンラインジャッジでは、問題文と模範解答ないしは他の学習者が提出した解答コードが与えられるだけであり、問題の解答を見ずに解答を考える際に手がかりとなる情報として、学習者または教員が外部の参考文献を探してくる必要があった。そこで、文書の推薦アルゴリズムを応用して、問題の模範解答の類似コードを OSS のソースコード等から推薦する AI 技術を組み込む。推薦された類似コードを読むことで解答に必要な構文などをその場で学べるようにして、学習者自身で効率的に学習を進められるような仕組みを組み込んだ。

なお、当初は本 AI 技術では類題を推薦することで、理解度に応じて学習に適した課題を提示する想定であった。しかし、実際に類題を推薦する AI を試作した結果、類題候補である既存の問題はいずれも本事業が対象とする問題よりも粒度が大きいことや、既存の問題では解答に求められるスキルがアルゴリズムの実装に偏っており、本事業が対象とする問題で必要となる Python の文法やライブラリの用法などのスキルを問えないことから、既存の問題は類題として使用できないことが分かった。そのため、本 AI 技術は類似コードを推薦することで解答の手がかりを提示し、学習者自身で効率的に学習を進められる仕組みを実現することとした。それに伴い、AI 技術の名称も「学生の理解状況を推定する AI」から「課題に応じて模範となるコードを提示する AI」に変更している。

②AI 技術 2：不正解の原因・誤り箇所を推定する AI

既存のオンラインジャッジでは、学生がバグを含んだプログラムを提出した際に、不正解である旨

が表示されるだけで、何が原因で不正解となっていて、プログラム中のどの部分に誤りがあるか分からない。熟達したプログラマーであれば、デバッグ作業を通してプログラム中の誤りを見つけることができるが、プログラミング初学者が自身でデバッグすることは難しい。そこで、ソフトウェア工学研究において、研究が進んでいる Fault Localization 技術を応用して、誤りの原因および誤った箇所を推定する AI 技術を組み込んだ。既に学術界においては、Fault Localization 技術をオンラインジャッジに組み込んだ研究事例も報告されている。

③AI 技術 3：学生の個性に応じて動機づける AI

従来のオンラインジャッジでは、学生が問題を解いたか否かに応じて、点数を獲得できるといった仕組みを備えており、システムから学生にフィードバックを与えることができる。しかし、フィードバックの内容は工夫されておらず、単に成否に応じた得点情報が与えられるだけである。心理学研究においては、学生の個性に適したフィードバックを与えることで、全員一律で同じフィードバックを与えるよりも優れた改善効果をもたらすという研究結果が報告されている。そこで、学生の個性に合わせてフィードバックを与えることで、学習意欲をより効果的に引き出す AI 技術を組み込んだ。また、ゲーミフィケーション技術により、プログラミング学習を促すことに成功した研究事例もあり、AI 技術によりフィードバックを個別最適化するとともに、そもそものフィードバックの選択肢を魅力的なものとした。

④AI 技術 4：採点を完全に自動化する AI

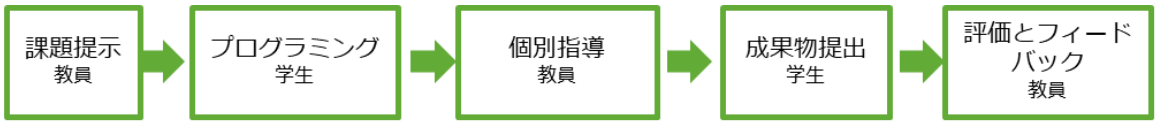
従来のオンラインジャッジは、プログラムを自動採点するために必要なテストケースを教員が用意する必要があり、テストケースの作成は容易ではないという問題があった。採点回数が十分に多ければ、採点コストがテストケースの作成コストを上回り、自動化によるコスト削減効果が出るものの、採点回数が少ないと、かえって採点の手間が増える問題がある。また、自動採点では正解・不正解の 2 値でしか評価できず、プログラムの品質を測定することはできない。そこで、ソフトウェア工学研究におけるテストケース生成技術を応用することで、与えられた模範解答からテストケースを自動生成する AI 技術を組み込み、さらに、ソフトウェアメトリクスの測定技術を組み込むことで、教員の負荷を下げながら、より詳細なフィードバックを学生に与えられるような仕組みを組み込んだ。

以上の 4 つの AI 技術を組み込んだオンラインジャッジ（AI 搭載オンラインプログラミング実習システム） を開発することにより、遠隔環境下で生じる前述の 4 つの問題（①学生の取組状況を把握が困難、②プログラム作成につまずく学生への個別対応が困難、③教員負荷が大きく、放課後フォローやプログラム品質に関する指導がますます困難、④学生の意欲や集中力の維持が困難）の解決をはかった。なお、本事業では、AI 搭載オンラインプログラミング実習システムの開発の他、運用マニュアルやこのシステムの利用を前提としたカリキュラム・シラバス、テキスト教材を製作する。また、情報系専門学校にソフトウェア品質に関する教育を普及させるため、テキスト教材の一部にプログラム品質の評価基準に関する解説を盛り込む。

2.1.3. 遠隔教育の導入方策とそのモデル化の概要①

●遠隔授業における従来型プログラミング実習の問題点

プログラミング実習は、プログラム言語の文法知識の暗記にとどまらない応用力を育成する活動で、プログラミング教育の中核をなす。対面授業では安定的に運営できたプログラミング実習授業のプロセスだが、遠隔授業下では 4 つの問題が生じている。



①学習状況の把握が困難
適切な課題を提示できない

②学生の意欲維持が困難
学生が課題に取り組まない

③巡回による個別指導が困難
学生が課題を解けない

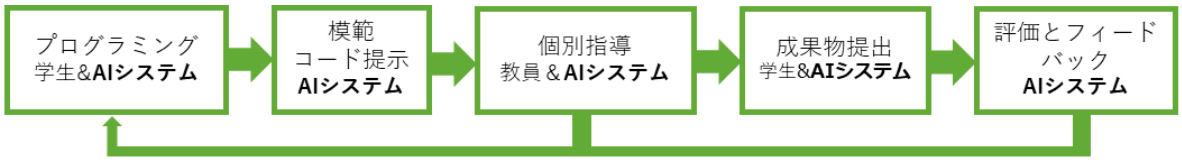
④対面より課題数が増え教員の負担増加
フィードバックの質が下がる

AI 搭載オンラインプログラミング実習システムの導入

・実習プロセス全体をオンライン上で実施
・需要の高い Python に対応
・実習授業のコマ数にかかわらず導入可能

●AI 搭載オンラインプログラミング実習システムを導入したプログラミング実習モデル

4 つの AI を搭載した実習システム・教材・課題を導入することで、前述の 4 つの問題を解決する。AI の迅速かつ最適化されたフィードバックにより、実習の質が向上する。さらに、教員は教材を用いた解説、および、解説を理解できなかった学生のサポートなどの本来あるべき個別指導に集中できる。



①模範コードを提示するAI
理解状況に基づいて適切なコードを提示できる

②学生の個性に応じて動機づけるAI
AIにより学生が意欲的に課題に取り組める

③不正解の原因・誤り箇所を推定するAI
学生が課題の解答でつまずいても、AIの個別サポートで解ける。（教員は理解につまずいた一部学生への丹念な個別指導だけに集中できる）。

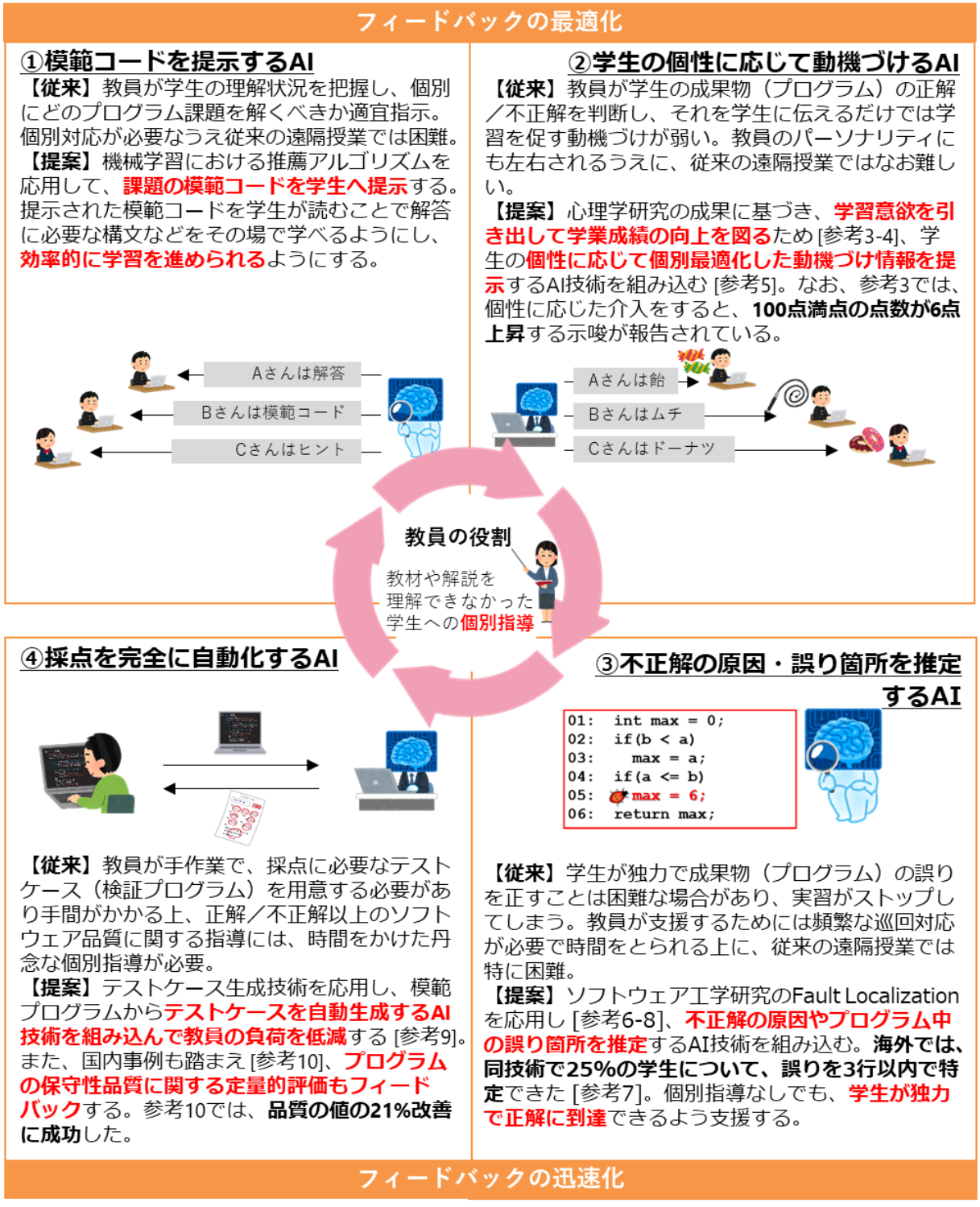
④採点・評価を完全に自動化するAI
AIにより迅速かつ質の高いフィードバックを返せるようになり、学生がフィードバックを踏まえて**再学習できる**

教員は、実習中に理解につまずいた一部学生の個別指導だけに集中できる

2.1.4. 遠隔教育の導入方策とそのモデル化の概要②

●提案システム・４つの AI の詳細

学生の成果物を自動採点するプログラミング学習システムは存在するが、前述の遠隔授業での課題解決に適していない。そこで、以下で示す４つの AI を搭載した、開発環境がなくとも（OS を問わず） **Web ブラウザ上で動作するプログラミング実習システムを導入するため、遠隔授業下で利用できる。**



2.1.5. 遠隔教育の導入方策とそのモデル化の概要③

本事業で開発するプログラミング実習モデルは、Python 言語に関する実習授業で、基礎 25 コマ、応用 5 コマから構成される。その内訳として、以下の通りである。

基礎コマ

章	節
1 プログラミング・Python の基本知識	1.1 コンピュータとソフトウェア 1.2 プログラム 1.3 プログラミング言語の種類 1.4 Python 言語
2 変数・データ型・代入・数値計算	2.1 変数 2.2 代入 2.3 データ型 2.4 演算子と数値計算 2.5 プログラム作成課題
3 制御フロー	3.1 リストの基礎 3.2 繰り返し（1） 3.3 条件分岐 3.4 繰り返し（2） 3.5 繰り返しの制御 3.6 パターンマッチ 3.7 プログラム作成課題
4 関数とメソッド	4.1 関数 4.2 オブジェクトとメソッド 4.3 プログラム作成課題
5 コレクション	5.1 コレクションの種類 5.2 リスト 5.3 タプル 5.4 集合 5.5 辞書 5.6 プログラム作成課題
6 クラス	6.1 名前空間とスコープ 6.2 クラス 6.3 オブジェクト指向プログラミング 6.4 継承 6.5 プログラム作成課題
7 モジュール	7.1 モジュール 7.2 パッケージ 7.3 標準ライブラリと外部ライブラリ 7.4 プログラム作成課題
8 入出力	8.1 標準入出力 8.2 ファイル入出力 8.3 クリーンアップ処理 8.4 プログラム作成課題

章	節
9 エラーと例外	9.1 エラーの種類 9.2 例外の処理 9.3 例外の送出 9.4 独自例外の定義 9.5 プログラム作成課題
10 ソフトウェア品質とコーディング規約	10.1 ソフトウェア品質 10.2 コーディング規約 10.3 プログラム作成課題
11 探索アルゴリズム	11.1 時間計算量・空間計算量 11.2 線形探索 11.3 二分探索 11.4 プログラム作成課題
12 ソートアルゴリズム	12.1 バブルソート 12.2 選択ソート 12.3 挿入ソート 12.4 マージソート 12.5 プログラム作成課題
13 アプリケーション開発 (1)	13.1 コンソールアプリケーションの説明 13.2 コンソールアプリケーションの実装 13.3 コンソールアプリケーションの追加実装 13.4 プログラム作成課題
14 アプリケーション開発 (2)	14.1 クラスを使ったコンソールアプリケーションの実装 14.2 クラスを使ったコンソールアプリケーションの追加実装 14.3 プログラム作成課題
15 アプリケーション開発 (3)	15.1 継承を使ったコンソールアプリケーションの実装 15.2 リファクタリング 15.3 プログラム作成課題
16 リストと配列	16.1 データ構造 16.2 単方向リスト 16.3 双方向リスト 16.4 循環リスト 16.5 Python における配列 16.6 プログラム作成課題
17 スタックとキュー	17.1 スタック 17.2 キュー 17.3 優先度付きキュー 17.4 プログラム作成課題
18 ハッシュテーブル	18.1 ハッシュテーブル 18.2 オープンアドレス法 18.3 チェイン法 18.4 プログラム作成課題
19 木構造	19.1 木構造 19.2 二分木 19.3 完全二分木 19.4 プログラム作成課題

章	節
20 二分探索木	20.1 深さ優先探索 20.2 幅優先探索 20.3 二分探索木 20.4 プログラム作成課題
21 複雑な木	21.1 平衡木 21.2 多分木 21.3 ヒープ 21.4 プログラム作成課題
22 ビット演算	22.1 N 進数 22.2 シフト演算 22.3 論理演算 22.4 ビットマスク 22.5 プログラム作成課題
23 文字列処理	23.1 Python の文字列処理 23.2 文字列アルゴリズム 23.3 データ圧縮 23.4 プログラム作成課題
24 デザインパターン 1	24.1 デザインパターン 24.2 Singleton パターン 24.3 Decorator パターン 24.4 Strategy パターン 24.5 プログラム作成課題
25 デザインパターン 2	25.1 Adapter パターン 25.2 State パターン 25.3 Factory Method パターン 25.4 Template Method パターン 25.5 プログラム作成課題

応用コマ

章	節
1 データ処理	1.1 CSV ファイル 1.2 CSV ファイルの読み込み 1.3 CSV ファイルの書き込み 1.4 CSV ファイルの集計・可視化 1.5 NumPy による配列の処理 1.6 プログラム作成課題
2 pandas (1)	2.1 pandas の概要 2.2 pandas によるファイル読み込み 2.3 pandas によるデータフレームの操作（基礎） 2.4 pandas による統計量の計算 2.5 プログラム作成課題
3 pandas (2)	3.1 pandas によるデータフレームの操作（発展） 3.2 pandas による欠損値や異常値の確認 3.3 pandas によるデータクレンジング 3.4 プログラム作成課題
4 Web スクレイピング	4.1 Web スクレイピング 4.2 HTTP リクエスト・レスポンス 4.3 HTML のパースおよびデータ抽出
5 Web スクレイピングしたデータの加工・分析	5.1 pandas を用いた Web スクレイピングデータの加工 5.2 pandas を用いた Web スクレイピングデータの分析 5.3 プログラム作成課題

2.2. 遠隔環境におけるプログラミング実習モデルの実証・検収

2.2.1. 実証概要

目的

本事業で開発した AI 搭載オンラインプログラミング実習システムおよび課題付きテキスト教材によって、専門学校のプログラミング実習の遠隔教育時における課題を解決可能か明確化することを本実証・検収の目的とする。

プログラミング実習の遠隔教育を「時間」と「効果」の観点から比較することにより、定量的な評価を行う。受講者（専門学校学生）を 2 群（A 群・B 群）に分け、A 群を実験群・B 群を統制群とする。実験群に対しては、本事業の成果物である AI 搭載オンラインプログラミング実習システム＋課題付きテキスト教材で授業を行い、統制群に対しては、従来のプログラミング実習指導＋課題付きテキスト教材で授業を行う。なお、従来のプログラミング実習は、対面授業で実施していた内容をできる限りそのまま遠隔授業で実施する形式に変更したものとする。実証講座の 5 コマ目で、実証講座で得た知識・スキルを評価するための事後テストを実施する。群間で本テストの採点結果を統計的に比較することで、「効果」（学生のプログラミング能力）の評価を行う。また、教員が授業の準備・運営等で費やした時間を測定して比較することで、教員が教育活動に要した「時間」（教育活動に必要なコスト）の評価を行う。宿題の採点など、教員が授業外で学生に対してフィードバックを行うための時間も「時間」コストに含める。

また、主となる実証の対象は河原電子ビジネス専門学校だが、加えて情報科学専門学校および静岡産業技術専門学校の学生（それぞれ 115 名、44 名）に対しても、AI 搭載オンラインプログラミング実習システムと課題付きテキスト教材を用いて授業・テストを行い、補助的な分析を行う。

実証の対象・期間・検証項目

以下に、河原電子ビジネス専門学校における実証の詳細を示す。

項目	概要
対象者	河原電子ビジネス専門学校のプログラミング関連学科の学生 46 名
群分け	統制群（23 名）：従来型のプログラミング実習授業 実験群（23 名）：本事業の成果物（AI 搭載オンラインプログラミング実習システム） 事前テストを実施し、両群の平均点に有意差がないことを統計的に確認したうえで実施する。
実証期間	全 5 コマのプログラミング遠隔授業で、AI 搭載オンラインプログラミング実習システム＋課題付きテキスト教材の検証を行う。
仮説	AI 搭載オンラインプログラミング実習システムの導入によって従来型の遠隔授業と比較して、学習効果・学習意欲が向上する。また、実習授業の準備や運営等で教員が教育活動に要する時間が短縮する。
検証項目	・事前・事後テストの採点結果が統計的有意に向上するか否か。 ・宿題提出状況およびアンケートから学習意欲が定量的に向上するか否か。 ・教員がプログラミング実習に要する準備・運営・事後処理等で費やした時間が減少するか否か。

2.2.2. 実証成果集計・分析

１．事前・事後テストの結果

本実証講座では、講座を実施する前後でそれぞれテストを 1 回ずつ実施した。テストの内容は実験群・統制群いずれも同一である。事後テストでは、プログラミング言語の細かい記法の知識について問うのではなく、if 文や for 文などの各種要素の組み合わせや、コレクション・クラスの応用によるプログラムの構築能力を問うことで、プログラミングの本質的な能力が向上したことの確認を意図しており、学生が Python の実行環境を使用して、自ら作成したプログラムの動作確認を行うことが必要となる。そのため、Python の実行環境を学生が確実に使用するよう、本事業で開発したオンラインプログラミング実習システムにてプログラムを提出する形式で実施した。本実習システムでは解答の提出時に自動的にプログラムが実行されるため、必ず動作の確認を行うことになる。また、事後テストにおいて事前テストと同様の内容を出題した場合、上位の学生は満点に近い点数を取り天井効果が生じることが想定されるため、全学生の実力を正確に測定できるように、基礎的な問題から発展的な問題まで幅広い難易度の問題を出題した。なお、実習システムに搭載されているヒントを提示する AI（誤り箇所推定 AI および類題提示 AI）は、事後テストにおいて無効化した。また、事後テストの実施前に、システム上でプログラムの作成・提出を練習する時間を両群に対して1時間半ほど与えることで、システムの使用経験の差が結果に影響を与えないようにした。

事前テストは 10 問で構成され、第 1 回講座で扱う制御構文（4 問）、第 2 回および第 3 回講座で扱うコレクション（4 問）、第 4 回講座で扱うクラス（2 問）である。事後テストは、第 1 回講座～第 4 回講座で扱う内容で構成された、基本問題 5 問と発展問題 11 問の計 16 問とした。基本問題は講義中に解説した問題の類題であり、発展問題は後述する宿題の類題である。事前テストは 1 問 10 点の 100 点満点とした。事前テストは全 16 問のため、1 問 6.25 点として 100 点満点になるよう調整した。事前テスト、事後テストのいずれにおいても部分点は用いず、正解なら満点、不正解なら 0 点とした。

表 1 に、事前テストと事後テストの点数を学生ごとに示す。テストを受験していない場合は「N/A」と記載している。事前テストを受験していない学生（23 番目）が実験群に 1 名存在するため、本節における以降の分析ではこの学生は除外する。

表 1. 事前・事後テストの学生ごとの点数

	群の振り分け	事前テスト（点）	事後テスト（点）	事前・事後テストの差分（点）
1	実験群	0	18.75	**18.75**
2	実験群	0	18.75	**18.75**
3	実験群	0	37.5	**37.5**
4	実験群	0	6.25	**6.25**
5	実験群	0	6.25	**6.25**
6	実験群	10	12.5	**2.5**

7	実験群	10	93.75	**83.75**
8	実験群	10	75	**65**
9	実験群	10	18.75	**8.75**
10	実験群	10	43.75	**33.75**
11	実験群	10	43.75	**33.75**
12	実験群	20	62.5	**42.5**
13	実験群	20	12.5	**-7.5**
14	実験群	20	50	**30**
15	実験群	20	75	**55**
16	実験群	20	100	**80**
17	実験群	20	6.25	**-13.75**
18	実験群	30	56.25	**26.25**
19	実験群	50	31.25	**-18.75**
20	実験群	50	81.25	**31.25**
21	実験群	70	100	**30**
22	実験群	70	93.75	**23.75**
23	実験群	N/A	37.5	**N/A**
24	統制群	0	6.25	**6.25**
25	統制群	0	12.5	**12.5**
26	統制群	0	31.25	**31.25**
27	統制群	0	6.25	**6.25**
28	統制群	0	6.25	**6.25**
29	統制群	10	37.5	**27.5**
30	統制群	10	31.25	**21.25**
31	統制群	10	6.25	**-3.75**
32	統制群	10	56.25	**46.25**
33	統制群	10	37.5	**27.5**
34	統制群	20	37.5	**17.5**
35	統制群	20	31.25	**11.25**
36	統制群	20	50	**30**
37	統制群	20	37.5	**17.5**
38	統制群	20	25	**5**
39	統制群	20	56.25	**36.25**
40	統制群	20	50	**30**
41	統制群	30	18.75	**-11.25**
42	統制群	50	50	**0**
43	統制群	50	56.25	**6.25**
44	統制群	50	18.75	**-31.25**
45	統制群	70	56.25	**-13.75**

46	統制群	90	93.75	**3.75**

表 2. 事前・事後テストの平均点

	事前テスト（点）	事後テスト（点）	事前・事後テストの差分（点）
実験群	20.5	47.4	26.9
統制群	23.0	35.3	12.3

表 2 に、事前テストと事後テストの平均点（四捨五入）を、実験群・統制群それぞれについて示す。本実証講座では、事前テストの結果をもとに、実験群・統制群の平均値が同等になるように群の振り分けを行ったため、事前テストの段階では各群の平均値は同程度となっている。一方、講座の実施後に行った事後テストでは、実験群と統制群の平均点に差が出ている。実験群における事後テストの平均点は統制群における平均点の約 1.34 倍であり、約 34%の向上がみられる。また、事前・事後テストの差分は、実験群では 26.9 点であったのに対し、統制群では 12.3 点であった。ただし、事前テストと事後テストでは問題の構成や難易度が異なるため、点数差分に関しては、実力の向上量の指標としては参考値に留まる。

事前テストの段階では実験群と統制群の間で点数に統計的に有意な差が無いことを確認するため、前述した学生を除く各学生の事前テストの点数を観測値として、分布に仮定を置かないノンパラメトリック検定である Mann-Whitney の U 検定を行った。その結果 $p > .1$（U=238、実験群の標本サイズ=22、統制群の標本サイズ=23、両側検定）であったため、事前テストの点数に関しては実験群と統制群の間に有意な差は認められなかった。

図 1 は、各群における事後テストの正答数のヒストグラムである。実験群には 10 問以上正答している学生が複数存在するのに対し、統制群は分布が 9 問以下の領域に偏っている。

事後テストの点数を実験群と統制群で比較したとき、統計的に有意な差があるか検証するため、各学生の事後テストの点数を観測値として Mann-Whitney の U 検定を行った。しかし、結果は $p > .1$（U=301、実験群の標本サイズ=22、統制群の標本サイズ=23、両側検定）となり、両群の間に統計的に有意な差は確認できなかった。

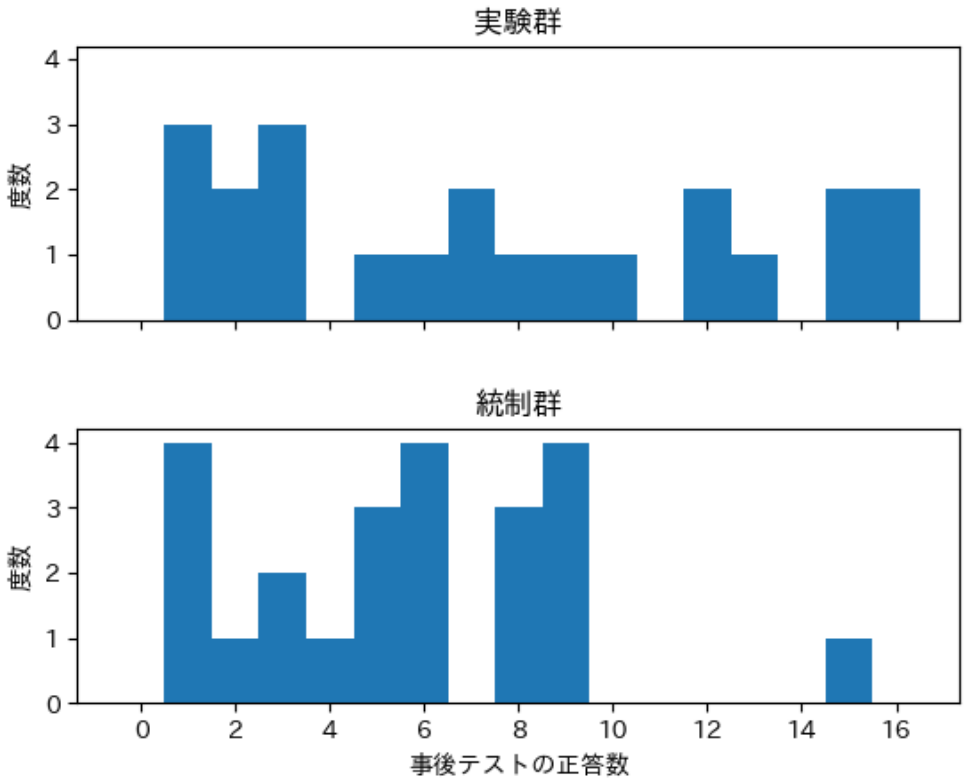


図 1. 各群における事後テストの正答数のヒストグラム

次に、事後テストで想定されている最低限の実力をつけたとみなせる学生に絞って比較を行った。事後テストは、基本問題 5 問・発展問題 11 問で構成されているため、5 問以上の正答（各単元における基本的な問題を解ける程度の実力がついているとみなす）を条件とした。

図 2 は、上記の条件を満たす学生に絞った時の、各群における事後テストの正答数のヒストグラムである。これは、図 1 における正答数が 5 問以上の領域を抽出したものとなっている。この図から、実験群のほうが、多くの問題に正答した学生の割合が大きいことが確認できる。

これらの学生間で比較したときに事後テストの点数に統計的に優位な差があるか検証するため、上記条件を満たす各学生の事後テストの点数を観測値として Mann-Whitney の U 検定を行った。その結果、$p < .05$（U=156、実験群の標本サイズ=14、統制群の標本サイズ=15、両側検定）となり、両群の間に統計的に有意な差があることを確認できた。

加えて、あくまでも参考値ではあるが、事前・事後テストの点数の差分についても実験群と統制群で比較を行った。まず、点数差分のヒストグラムを図 3 に示す。図からも確認できる通り、実験群・統制群ともに正規分布に近い形をしており、実験群のほうが分布の中心が右側にあることが確認できる。

これらの分布の間に統計的に有意な差があるか検証するため、各学生の事前テストと事後テストの差分を観測値として平均値の差の検定を行った。なお、これらの分布の特性を調べるために Shapiro-Wilk 検定（正規性の検定）および Bartlett 検定（等分散の検定）を行ったところ、それぞれ $p > .05$ となり、等分散な正規分布であると見なせたため、平均値の差の検定には Student の t 検定を用いた。その結果、$p < .05$（t=2.14、実験群の標本サイズ=22、統制群の標本サイズ=23、両側検定）であったため、両群の間に統計的に有意な差があることが確認された。

また、誤り箇所推定 AI のサポートが学習に与えた影響を調べるため、誤り箇所推定 AI の使用率と事後テストの点数の関係を分析した。図 4 は、各学生が実習システム上で正解した問題のうち、誤り箇所推定 AI を使用した問題の割合を横軸とし、事後テストの点数を縦軸とした散布図である。例えば、ある学生の誤り箇所推定 AI の使用率が 50%の場合、その学生の正解した問題のうちの半分で誤り箇所推定 AI を使用したことを表す。この図から、誤り箇所推定 AI の使用率が高いほど、事後テストの点数が低くなる傾向が読み取れる。両者の Kendall の順位相関係数は−0. 40 であり、$p < .01$ であったため、統計的に有意な中程度の負の相関があることが確認された。

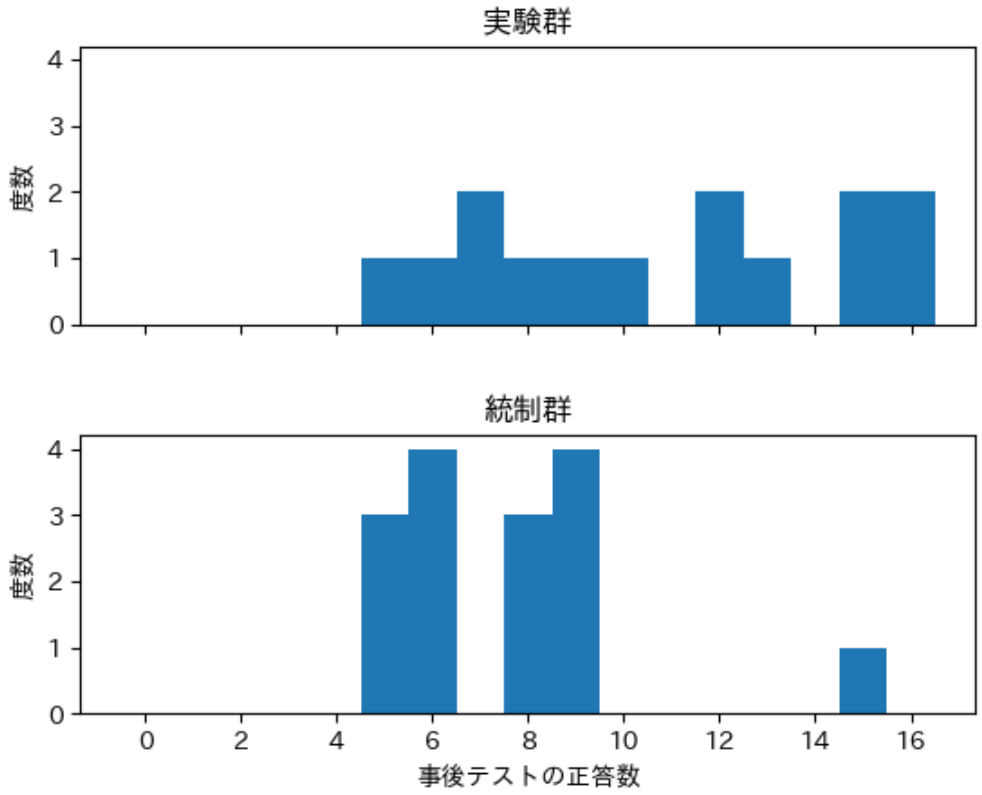


図 2. 各群における事後テストの正答数のヒストグラム

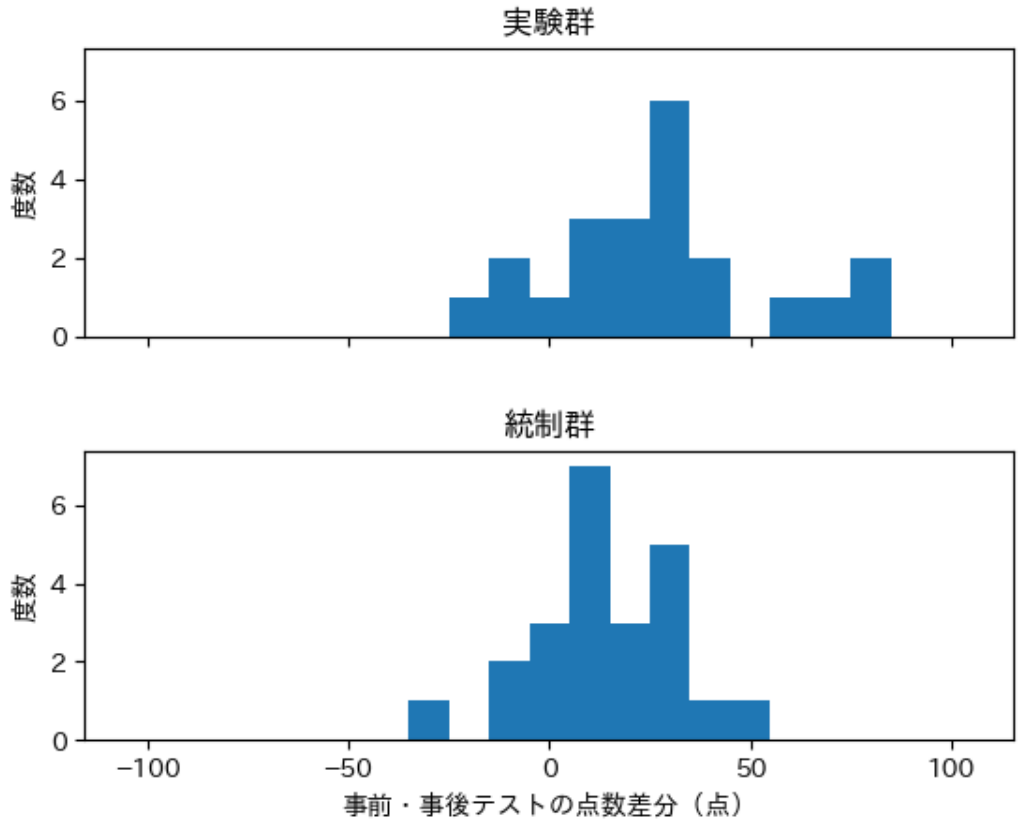


図 3. 各群における事前テストと事後テストの点数差分のヒストグラム

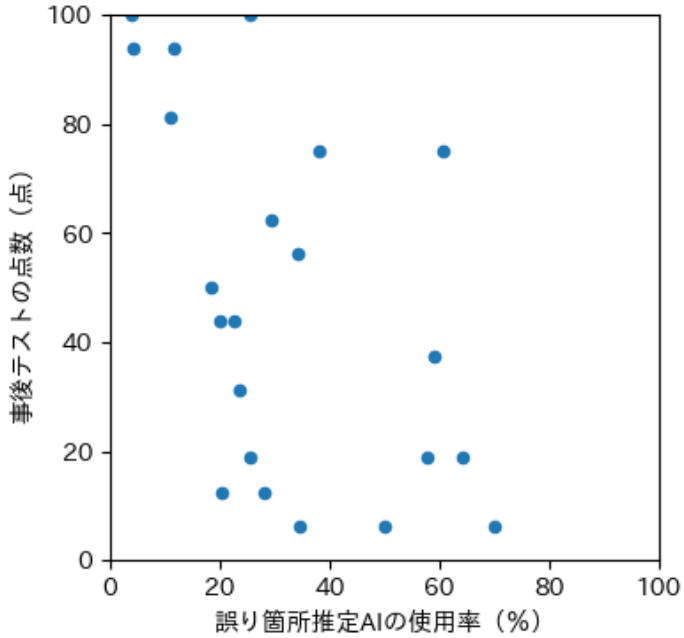


図 4. 実験群における誤り箇所推定 AI の使用率と事後テストの点数の関係

さらに、各学生のプログラミング等のスキルおよび学習意欲に関して教員が評価を行い、それを観点に加えた学習効果の分析を行った。以下の 7 項目を評価項目として、1（最低）～4（最高）の 4 段

階評価とした。

- 項目 1： プログラミングはどの程度得意ですか。(1：とても苦手、2：少し苦手、3：少し得意、4：とても得意)
- 項目 2： プログラミングに興味や熱意はどの程度ですか。(1：全く興味がない、2：あまり興味がない、3：少し興味がある、4：興味がある)
- 項目 3： 普段のプログラミング授業態度はどうですか。(1：消極的、2：少し消極的、3：少し積極的、4：積極的)
- 項目 4： 普段のプログラミング以外の授業態度はどうですか。(1：消極的、2：少し消極的、3：少し積極的、4：積極的)
- 項目 5： プログラミングの成績はどうですか。(1：GPA0～1、2：GPA1～2、3：GPA2～3、4： GPA3～4)
- 項目 6： プログラミング以外の成績はどうですか。(1：GPA0～1、2：GPA1～2、3：GPA2～3、4： GPA3～4)
- 項目 7： 普段の授業では質問は積極的ですか。(1：消極的、2：少し消極的、3：少し積極的、4：積極的)

このうち、項目 1、5、6 は学生のスキルに対する評価であり、項目 2、3、4、7 は学生の学習意欲に対する評価である。各学生について、スキルに対する評価項目の値を合算したものをスキルに対する評価スコアとし、同様に学習意欲に対する評価項目の値を合算したものを学習意欲に対する評価スコアとした。

表 3 に、スキルに対する評価スコアの高低によって実験群・統制群を層別したときの、事前・事後テストの平均点（四捨五入）を示す。実験群・統制群を合わせた全学生内での中央値を求め、中央値以下の評価スコアの学生を低評価、中央値より大きい評価スコアの学生を高評価とした。同様に、学習意欲に対する評価スコアの高低によって実験群・統制群を層別したときの、事前・事後テストの平均点を表 4 に示す。表 3、表 4 のいずれにおいても、実験群・高評価のほうが統制群・高評価よりも事前テストの平均点は低いが、事後テストの平均点は大幅に高くなっていることが確認できる。

表 3. スキルに対する評価スコアで層別した事前・事後テストの平均点

	事前テスト（点）	事後テスト（点）
実験群・低評価	14.0	28.8
統制群・低評価	13.6	30.4
実験群・高評価	25.8	63.0
統制群・高評価	37.8	43.1

表 4. 学習意欲に対する評価スコアで層別した事前・事後テストの平均点

	事前テスト（点）	事後テスト（点）
実験群・低評価	12.0	30.0
統制群・低評価	16.7	28.3
実験群・高評価	27.5	62.0
統制群・高評価	35.0	48.4

２．宿題の実施状況

本実証講座では講座回ごとに扱った内容に関する問題を宿題として提示し、解く問題や分量を学生自身の意思で決めさせて提出させた。第 1 回講座で 18 問、第 2 回講座で 11 問、第 3 回講座で 10 問、第 4 回講座で 8 問、加えて発展問題を 5 問出題し、全部で 52 問とした。実験群は AI 搭載オンラインプログラミング実習システム上で問題を解くことで提出させ、統制群は Google フォームで提出させた。

表 5 に、宿題の提出数・正解数の平均値（四捨五入）を実験群・統制群それぞれについて示す。実験群は実習システム上で問題を解き、判定結果が正解となったときにはじめて提出とカウントされるため、提出数と正解数は同一である。提出数、正解数いずれも実験群のほうが統制群と比べて多く、実験群と統制群の差はそれぞれ 9.1 問、19.5 問である。特に正解数において顕著な差があり、実験群は統制群と比べて、より宿題に正解するまで取り組む傾向がみられる。

図 5 は、各群における宿題の提出数および正解数のヒストグラムである。実験群は提出数・正解数ともに 52 問付近を峰とした右に偏った分布になっているのに対し、統制群については、提出数は 47 問付近を峰とした分布、正解数は 0 問から 47 問まで一様に近い分布となっている。

実験群と統制群の宿題の提出数・正解数の間に統計的に有意な差があるか検証するため、各学生の提出数・正解数を観測値として、提出数と正解数のそれぞれに対して Mann-Whitney の U 検定を行った。その結果、提出数は $p < .01$（U=429.5、実験群の標本サイズ=23、統制群の標本サイズ=23、両側検定）であり、正解数は $p < .001$（U=505.5、実験群の標本サイズ=23、統制群の標本サイズ=23、両側検定）であったため、両群の間に統計的に有意な差があることが確認された。

表 5. 宿題の提出数・正解数の平均値

	提出数 （問）	正解数 （問）
実験群	44.5	44.5
統制群	35.4	25.0

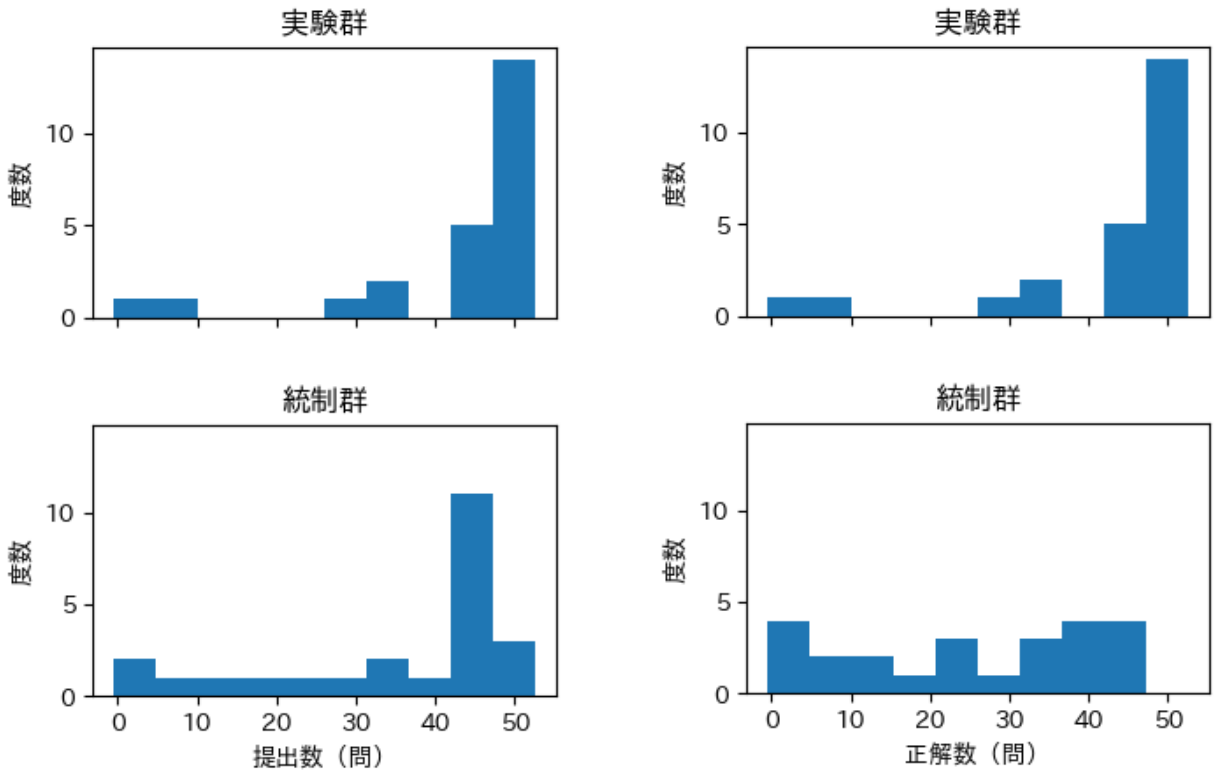


図 5. 各群における宿題の提出数（左）および正解数（右）のヒストグラム

表 6. スキルに対する評価スコアで層別した宿題の提出数および正解数

	提出数 （問）	正解数 （問）
実験群・低評価	45.2	45.2
統制群・低評価	37.9	24.6
実験群・高評価	43.9	43.9
統制群・高評価	31.9	25.7

表 7. 学習意欲に対する評価スコアで層別した宿題の提出数および正解数

	提出数 （問）	正解数 （問）
実験群・低評価	47.3	47.3
統制群・低評価	33.5	21.5
実験群・高評価	41.7	41.7
統制群・高評価	38.6	31.0

また、前節における事前・事後テストの分析と同様に、スキル・学習意欲に対する評価スコアを用いて宿題の提出数・正解数に関する分析を行った。表 6 および表 7 に、スキル・学習意欲それぞれに対する評価スコアで層別した宿題の提出数および正解数を示す。表 6 のスキルに対する評価スコアによる

層別では、低評価・高評価の群の間で大きな差は見られず、表 5 における層別をしない場合と概ね同様の傾向となっている。一方、表 7 の学習意欲に対する評価スコアによる層別では、正解数において顕著な特徴がみられる。統制群では低評価の群の正解数が高評価の群に比べて大きく低下しているのに対し、実験群ではそういった低下は見られず、むしろ低評価の群の正解数が高評価の群を上回っている。

３．事後アンケートの結果

表 8. 実証講座の事後アンケート（本講座に対する評価の質問）の結果。回答者に提示した各質問の「良い／悪い」の選択肢は、質問の上から順に、1-4) 理解できた／できなかった、5-8) 役に立った／役に立たなかった、9) 分かり易かった／分かり難かった、10) 意欲的に取り組めた／意欲的に取り組めなかった、11) 増えた／減った、12) 感じた／感じなかった、13) 使い易かった／使い難かった、14) 役に立った／役に立たなかった、15) そう思う／そう思わないと表現した。また、11 番目の質問では「普通」に代えて「変わらなかった」と表現し、それ以外の質問では「普通」の選択肢を設けなかった。

	質問内容	群	とても良い	やや良い	普通	やや悪い	とても悪い	とても良い＋やや良い
1	講義の理解度（第 1 回）	実験	57%	35%		9%	0%	91%
		統制	32%	50%		14%	0%	82%
2	講義の理解度（第 2 回）	実験	48%	39%		13%	0%	87%
		統制	27%	50%		18%	5%	77%
3	講義の理解度（第 3 回）	実験	30%	43%		26%	0%	74%
		統制	14%	55%		27%	5%	68%
4	講義の理解度（第 4 回）	実験	26%	35%		35%	4%	61%
		統制	9%	50%		36%	5%	59%
5	講義と宿題は Python プログラミングの理解度向上に役立ったか（第 1 回）	実験	30%	65%		4%	0%	96%
		統制	36%	50%		5%	9%	86%
6	講義と宿題は Python プログラミングの理解度向上に役立ったか（第 2 回）	実験	26%	74%		0%	0%	100%
		統制	41%	45%		5%	9%	86%
7	講義と宿題は Python プログラミングの理解度向上に役立ったか（第 3 回）	実験	26%	61%		13%	0%	87%
		統制	36%	45%		9%	9%	82%
8	講義と宿題は Python プログラミングの理解度向上に役立ったか（第 4 回）	実験	26%	61%		13%	0%	87%
		統制	32%	50%		9%	9%	82%
9	講師の説明の分かりやすさ	実験	43%	43%		9%	4%	87%
		統制	27%	50%		18%	5%	77%
10	講座に意欲的に取り組めたか	実験	30%	43%		17%	9%	74%
		統制	36%	36%		23%	5%	73%
11	自主的な学習時間の変化	実験	13%	48%	39%	0%	0%	61%
		統制	0%	45%	50%	0%	5%	45%
12	Python プログラミングへの興味が増したと感じたか	実験	17%	52%		22%	9%	70%
		統制	9%	50%		32%	9%	59%
13	システムの使いやすさ	実験	43%	35%		4%	17%	78%
		統制	23%	32%		27%	18%	55%

14	システムは学習の役に立ったか	実験	52%	30%	13%	4%	83%
		統制	27%	45%	18%	9%	73%
15	システムを本講座以外でも使って学習したいか	実験	22%	65%	9%	4%	87%
		統制	18%	32%	32%	18%	50%

表 8 は、本実証講座の事後アンケートのうち、本講座の評価に関する質問について回答結果をまとめたものである。11 番目の質問（自主的な学習時間の変化）のみ 5 件法で、それ以外の質問では 4 件法で選択肢を設けた。実験群・統制群それぞれの群について、各選択肢を回答した学生の割合を示しており、最も多かった回答を灰色でハイライトしている。なお、割合は小数点以下を四捨五入としているため、合計は必ずしも 100%とならない。いずれの質問でも、実験群のほうが統制群と比べて「とても良い」と「良い」を合わせた回答の割合が多い。また、質問 4 と 11 を除く全ての質問において、実験群において最も多かった回答のほうが、統制群において最も多かった回答よりも良い評価か、もしくは同程度の評価である。

表 9 は本実証講座の難易度および問題の分量の適切さに関する質問の回答結果である。いずれの質問も 5 件法で選択肢を設けた。実験群において最も多い回答は全ての質問で「適切」であるのに対し、統制群では 3、4、7、8、9 番目の質問で「やや難しい」となっている。また、いずれの質問においても、「適切」と答えた割合は実験群のほうが統制群よりも多い。

表 9. 実証講座の事後アンケート（適切さに関する質問）の結果。回答者に提示した各質問の「簡単／難しい」の選択肢は、質問の上から順に、1-8) 簡単／難しい、9) 少ない／多いと表現した。

	質問内容	群	簡単	やや簡単	適切	やや難しい	難しい
1	講義の難易度（第 1 回）	実験	17%	13%	52%	9%	9%
		統制	27%	9%	41%	14%	9%
2	講義の難易度（第 2 回）	実験	13%	13%	57%	9%	9%
		統制	14%	14%	41%	23%	9%
3	講義の難易度（第 3 回）	実験	13%	4%	48%	22%	13%
		統制	5%	9%	23%	55%	9%
4	講義の難易度（第 4 回）	実験	4%	17%	39%	17%	22%
		統制	9%	0%	18%	59%	14%
5	宿題の難易度（第 1 回）	実験	17%	13%	57%	9%	4%
		統制	18%	9%	45%	18%	9%
6	宿題の難易度（第 2 回）	実験	9%	13%	52%	22%	4%
		統制	9%	14%	45%	23%	9%
7	宿題の難易度（第 3 回）	実験	9%	4%	48%	30%	9%
		統制	0%	5%	36%	36%	23%
8	宿題の難易度（第 4 回）	実験	4%	4%	39%	35%	17%
		統制	5%	0%	23%	45%	27%

9	講義・宿題の問題の分量	実験	9%	0%	48%	35%	9%
		統制	0%	5%	27%	41%	27%

表 10. 実証講座の事後アンケート（AI 搭載オンラインプログラミング実習システムの AI 機能に対する評価の質問）の結果。回答者に提示した各質問の「良い／悪い」の選択肢は、1) 見ていた／見ていなかった、2) 上がった／上がらなかった、3, 5, 7) 使っていた／使っていなかった、4, 6, 8) 役に立った／役に立たなかったと表現した。

	質問内容	とても良い	やや良い	やや悪い	とても悪い	とても良い＋やや良い
1	動機促進 AI の閲覧頻度	30%	43%	17%	9%	74%
2	動機促進 AI による学習意欲の向上	13%	52%	26%	9%	65%
3	誤り箇所推定 AI の使用頻度	30%	57%	13%	0%	87%
4	誤り箇所推定 AI は学習の役に立ったか	61%	35%	0%	4%	96%
5	類題提示 AI の使用頻度	22%	39%	26%	13%	61%
6	類題提示 AI は学習の役に立ったか	26%	52%	17%	4%	78%
7	ステップ実行機能の使用頻度	22%	48%	17%	13%	70%
8	ステップ実行機能は学習の役に立ったか	35%	43%	9%	13%	78%

表 10 は、本実証講座で使用した AI 搭載オンラインプログラミング実習システムの AI 機能およびステップ実行機能の評価に関する質問の回答結果である。AI 搭載オンラインプログラミング実習システムは実験群のみが使用したため、質問は実験群に対してのみ行った。いずれの質問においても、最も回答が多いのは「とても良い」または「やや良い」である。「とても良い」と「やや良い」を合算すると、いずれの質問でも 60%以上となっている。特に、誤り箇所推定 AI は評価が高く、使用頻度については 87%、役に立ったかについては 96%が「とても良い」または「やや良い」と回答している。

事後アンケートでは、表 8 から表 10 で示した質問に加えて、自由記述で今回使用したシステムに対する意見や要望も求めた。この回答のうち、AI 搭載オンラインプログラミング実習システムに関するものは以下の通りであった。なお、回答中の「ヒント」は誤り箇所推定 AI 機能または類題提示 AI 機能のことを指す。

- できれば卒業するまでは使えるようにしてほしいです
- ヒントはもう少し出しやすくしてもいいと思います。
- とても楽しく勉強することができました。ありがとうございました。
- 問題の文字数が多いとき画面の 2/3 が問題 1/3 がコードブロックと見にくかった、コードブロックを大きくできるが、逆に問題が見えなくなるので改行を入れるかしてほしいです。
- オンラインジャッチに関してはとてもよかったです。ヒントがあるのがとてもよかったです。
- 最終的には講義全てと、テキスト教材の基本編をすべてやらせていただき、全体順位 2 位という結果で終わらさせていただきました。順位が出ているためとてもモチベーションが上がりました。人より多く使った視点として、いくつか意見させていただきます。まず、品質についてです。全

体的にバグがほとんどなく、動作も特別重いことはほとんどありませんでした。しかし、整数型で入出力するべきところを文字列型ですると、提出ボタンを押した後に無限に結果が終わらない、という現象に出会った時がありました。そういう点のバリデーションを用意されていたらなと思います。次に採点方法です。今回試験的な面もあり、正誤判定のための入出力例が 1 つか 2 つのみの用意だったため、問題の意図とは異なったアプローチでも正解してしまうことが技術的には可能なのかと思われます。本実装ではたくさんの入出力例が用意されることが必要だと考えております。次に、手を挙げる機能です。アプリケーション内でのアクションの為、その通知がどういう手段で教員側に知らされるにせよ、その後のやり取りが Slack 等の別のアプリケーションになってしまうことに違和感を感じました。オンラインジャッジ内で相談できる機能であったり、そもそも手を挙げる機能を付けないという、どちらかに統一したデザインの方がシンプルで、学習以外の事に引っ掛かりを覚えない形になるのかなと思われます。最後に、ヒントについてです。別解を提示したり、正解を即座に提示する点は良いと思いました。また、ちょっとしたコードミスを修正してくれる点が個人的にとても良いと感じました。ただ、AI 側も 1 つの回答に固執していそうな動作をすることもあったため、今後 AI がコードを書く人に寄り添って誘導できるような成長をしてくれることを願ってます。

４．他校における事前・事後テストの結果、宿題の実施状況、および事後アンケートの結果

前節までは河原電子ビジネス専門学校における分析を行ったが、本節では情報科学専門学校および静岡産業技術専門学校における結果を分析する。なお、両校とも全学生が AI 搭載オンラインプログラミング実習システムを使用しており、本節では両校の学生を合わせて一つの群として分析を行う。

図 6、図 7 にそれぞれ事前・事後テストのヒストグラムを、表 11 に事前・事後テストの平均点を示す。事前・事後テストの片方あるいは両方を受験していない学生が約半数存在するため、これらの学生は除外する。河原電子ビジネス専門学校の実験群における結果と同様に、事後テストにおいて全体的に正答数の向上がみられ、平均点も事前テストと比べて 21 点向上していることが確認できる。

表 12、表 13 に、スキル・学習意欲に対する教員の評価スコアで層別した事前・事後テストの平均点を示す。教員の評価スコアは河原電子ビジネス専門学校において行ったのと同様の手法で算出した。なお、一部の学生については教員による評価が困難であり、各評価項目が欠損値であったため、それらの学生の評価値は 1～4 の中間の値の 2.5 とした。河原電子ビジネス専門学校の実験群ほど顕著な差は見られないものの、スキルに対する評価スコアが高評価の群のほうが、低評価の群と比べて、事前・事後テストの差分が大きい。

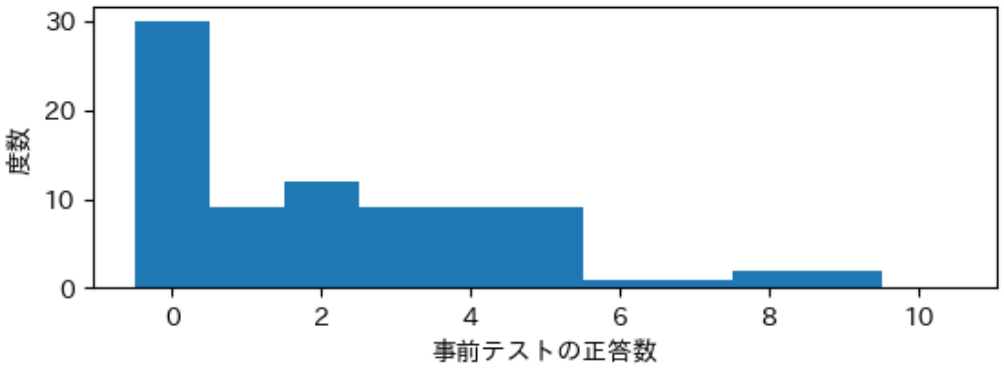


図 6. 事前テストの正答数のヒストグラム（他校）

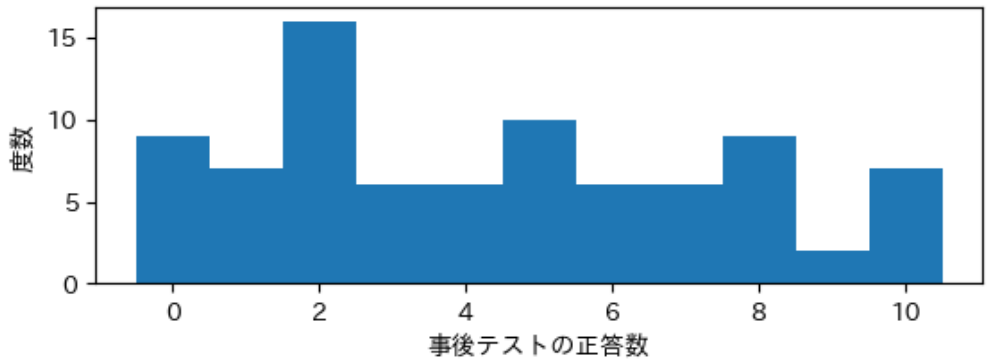


図 7. 事後テストの正答数のヒストグラム（他校）

表 11. 事前・事後テストの平均点（他校）

	事前テスト（点）	事後テスト（点）	事前・事後の差分（点）
全体	22.4	43.9	21.5

表 12. スキルに対する評価スコアで層別した事前・事後テストの平均点（他校）

	事前テスト（点）	事後テスト（点）	事前・事後の差分（点）
低評価	12.2	31.1	18.9
高評価	34.1	58.7	24.6

表 13. 学習意欲に対する評価スコアで層別した事前・事後テストの平均点（他校）

	事前テスト（点）	事後テスト（点）	事前・事後の差分（点）
低評価	15.4	36.8	21.4
高評価	36.4	58.2	21.8

図 8 に、宿題の提出数・正解数のヒストグラムを示す。なお、講義中で取り扱った問題も含めた全問題のうち 1 問も実施していない学生は、実証講座に参加していないとみなし、対象外とした。ヒストグラムから、宿題の提出数が 0 問付近の学生が一定数存在するものの、概ね河原電子ビジネス専門学校の実験群と同様に、多数の宿題を実施した学生が大部分を占めることが読み取れる。

表 14、表 15 に、スキル・学習意欲に対する評価スコアで層別した宿題の提出数・正解数を示す。河原電子ビジネス専門学校では実験群・低評価でも多数の提出・正解がみられたが、これらの表からはその傾向は見られない。一方で、高評価の群は河原電子ビジネス専門学校と同様に、多数の宿題を提出・正解している。

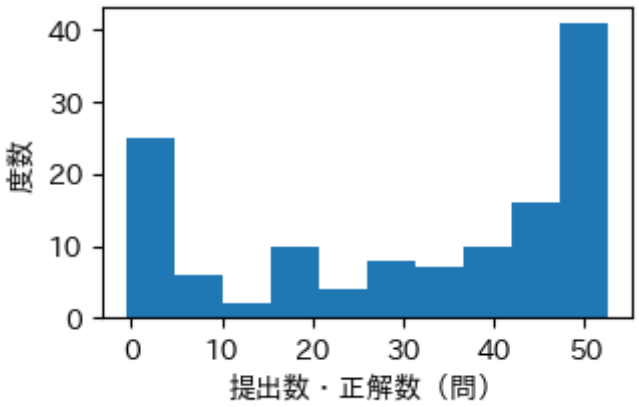


図 8. 宿題の提出数・正解数のヒストグラム（他校）

表 14. スキルに対する評価スコアで層別した宿題の提出数・正解数（他校）

	提出数・正解数 （問）
低評価	26.5
高評価	40.5

表 15. 学習意欲に対する評価スコアで層別した宿題の提出数・正解数（他校）

	提出数・正解数 （問）
低評価	26.1
高評価	38.5

表 16 に、AI 搭載オンラインプログラミング実習システムの評価、および各 AI 機能に対する評価のアンケート結果を示す。河原電子ビジネス専門学校の実験群よりも評価が落ちる項目がいくつか存在するものの、最も多い回答は「やや良い」または「良い」が大部分を占め、全体的に肯定的な評価である。また、河原電子ビジネス専門学校の実験群と同様に、誤り箇所推定 AI の評価が最も高い。

表 16. 実証講座の事後アンケート（AI 搭載オンラインプログラミング実習システムの評価、および各 AI 機能に対する評価の質問）の結果（他校）。回答者に提示した各質問の「良い／悪い」の選択肢は、表 8・表 10 で記載したものと同様である。

	質問内容	とても良い	やや良い	やや悪い	とても悪い	とても良い＋やや良い
1	システムの使いやすさ	19%	49%	29%	4%	69%
2	システムは学習の役に立ったか	32%	51%	11%	6%	83%
3	システムを本講座以外でも使って学習したいか	12%	59%	21%	8%	71%
4	動機促進 AI の閲覧頻度	17%	36%	26%	21%	52%
5	動機促進 AI による学習意欲の向上	9%	38%	34%	19%	47%
6	誤り箇所推定 AI の使用頻度	45%	31%	19%	4%	77%
7	誤り箇所推定 AI は学習の役に立ったか	51%	27%	12%	9%	79%
8	類題提示 AI の使用頻度	14%	34%	27%	25%	48%
9	類題提示 AI は学習の役に立ったか	18%	34%	31%	17%	52%
10	ステップ実行機能の使用頻度	17%	26%	36%	21%	42%
11	ステップ実行機能は学習の役に立ったか	17%	37%	26%	21%	54%

また、自由記述の回答のうち、AI 搭載オンラインプログラミング実習システムに関するものは以下の通りであった。

- ヒントとてもよかった
- 何行か移したい時にペーストできなくて書き直しなのが不便
- 使いやすくてよかったです。
- 画面のレイアウトがおかしいときがあったので直せるとよかったかなと思います
- 回答を判定するのが AI のためあっていても間違っていることになり先に進まないことがあった
- ヒント通り入力してもヒントが出てきたりしたので困惑した
- 作成したプログラムに自分で入力を行わなくてもテスト実行してくれるのは楽でよかったです
- オンラインジャッジのテスト欄をデフォルトで表示してほしい 問題が変わるごとに手動で表示させるのは面倒だし、提出する際にどちらにせよテスト欄を一度表示させるためデフォルトで表示でもよいと思った
- ヒント表示が分かりやすくて良かった
- 問題文の幅が大きすぎてプログラムを書く欄が見えにくくなることがあったので、そこを直していただきたいです。
- 問題文の誤りで正解できない問題は今後出ないようにしていただきたいです。宿題はなるべく解くという形なので助かっていましたが、答えがぎりぎりまで表示されない形式だったので解き進めるのに非常に時間がかかりました。

５．システムの利用状況

本実証講座の期間中における、AI 搭載オンラインプログラミング実習システム上での挙手機能の利用回数と、Slack 上での質問回数を計測した。表 17 に、各学校における挙手回数と質問回数を示す。質問は実験群・統制群ともに受け付けていたが、実際に質問があったのは実験群のみであった。質問に対しては、AI 搭載オンラインプログラミング実習システムにおける学生の解答ソースコードを確認する機能等を活用して回答を行った。また、実習システムにおいて挙手があったことを確認した直後に、Slack で当該学生に連絡して質問内容を問いかけたが、全 3 件の内訳は、1.挙手をしたがもう少し自身で取り組んでみるとの回答、2.誤って挙手機能を使用したとの回答、3.回答なし、であった。

表 17. 挙手回数および質問回数

	挙手回数	質問回数
河原電子ビジネス専門学校	1 名×1 回（計 1 回）	1 名×4 回（計 4 回）
情報科学専門学校	0 回	1 名×6 回、2 名×1 回ずつ（計 7 回）
静岡産業技術専門学校	2 名×1 回ずつ（計 2 回）	1 名×1 回（計 1 回）
合計	3 回	13 回

また、誤り箇所推定 AI の使用状況を分析した。図 9 に、河原電子ビジネス専門学校における、各学生が誤り箇所推定 AI を使用した問題の割合のヒストグラムを示す。1 つの観測値は 1 人の学生に対応し、使用率はその学生が正解した全問題のうち誤り箇所推定 AI を使用した問題の割合を表す。例えば、ある学生の使用率が 50%の場合、その学生が正解した問題のうち半分で誤り箇所推定 AI を使用したことを表す。20%～30%前後の使用率が最も多いが、全く使用していない学生や、大部分の問題で使用している学生も存在し、各学生が必要なサポート量に応じて AI 機能を使用していることが伺える。

図 10 に、第 1 回講義と第 4 回講義の問題に絞ったときの、誤り箇所推定 AI を使用した問題の割合のヒストグラムを示す。第 1 回講義は基本構文に関する問題で構成されるため、誤り箇所推定 AI の使用率が低い学生が多いが、第 4 回講義はクラスに関する問題で構成され、難易度も比較的高くなるため、誤り箇所推定 AI の使用率もそれに伴い高くなっている。ここでは図を割愛するが、他の講義回においても、難易度が高くなるにつれ誤り箇所推定 AI の使用率も高くなる傾向がみられた。

図 11 に、情報科学専門学校および静岡産業技術専門学校における、各学生が誤り箇所推定 AI を使用した問題の割合のヒストグラムを示す。河原電子ビジネス専門学校と比べて、使用率が低い学生と高い学生により幅広く分布しているが、やはり各学生とも必要な量に応じて機能を使用していることがわかる。

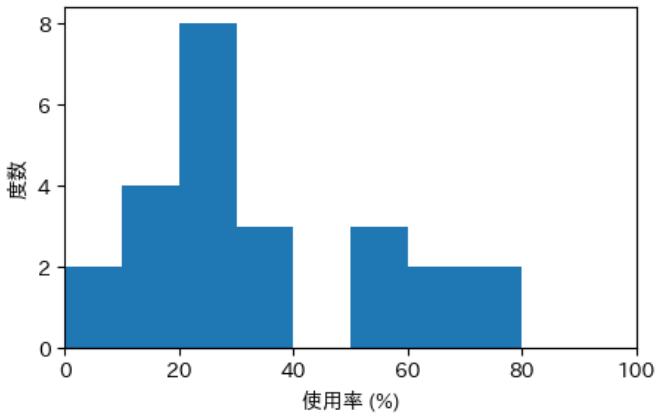


図 9. 誤り箇所推定 AI を使用した問題の割合のヒストグラム

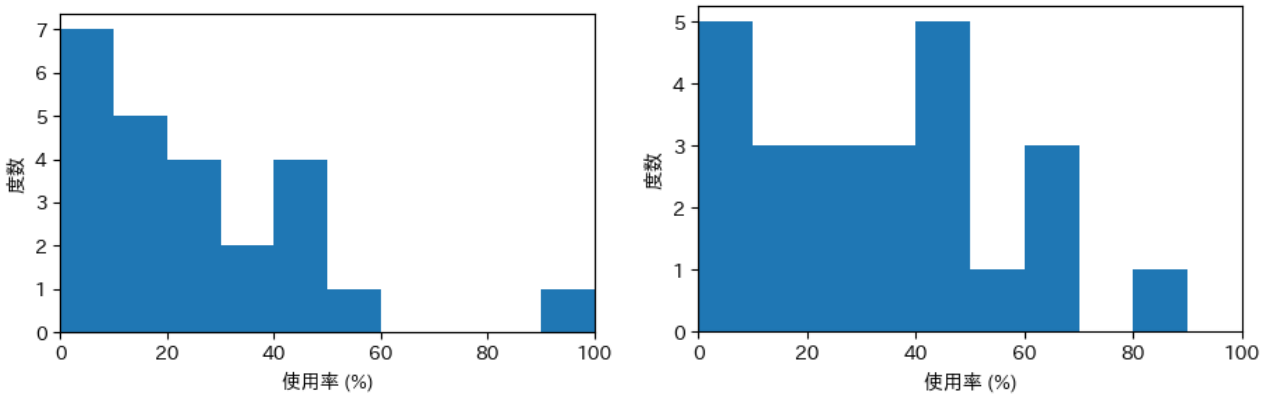


図 10. 誤り箇所推定 AI を使用した問題の割合のヒストグラム
（左：第 1 回講義の問題、右：第 4 回講義の問題）

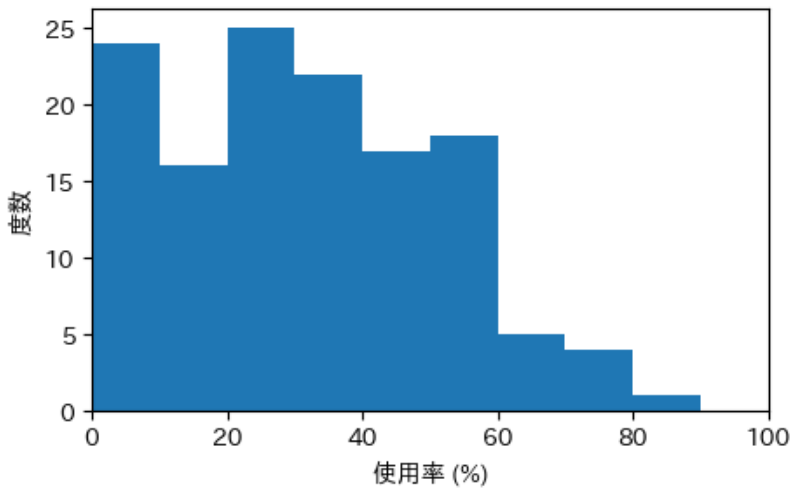


図 11. 誤り箇所推定 AI を使用した問題の割合のヒストグラム（他校）

さらに、学生が講義および宿題の各問題をどの程度の時間をかけて解いているか分析した。図 12 に、河原電子ビジネス専門学校における、正解した問題の平均解答時間のヒストグラムを示す。1 つの観測

値は 1 人の学生に対応し、その学生が正解した全問題の解答時間の平均値を表す。なお、時間の計測方法の関係で、例外的な操作をした一部学生の解答時間が非常に大きな値になっていたため、平均解答時間が 60 分より大きい学生は外れ値として除外した。大部分の学生は 1 問あたり 20 分以内に収まっており、自主学習が概ね円滑に行えていたことが確認できる。

図 13 に、情報科学専門学校および静岡産業技術専門学校における、正解した問題の平均解答時間のヒストグラムを示す。本図でもほとんどの学生が 1 問あたり 20 分程度までに収まっており、河原電子ビジネス専門学校と同様の傾向が確認できる。

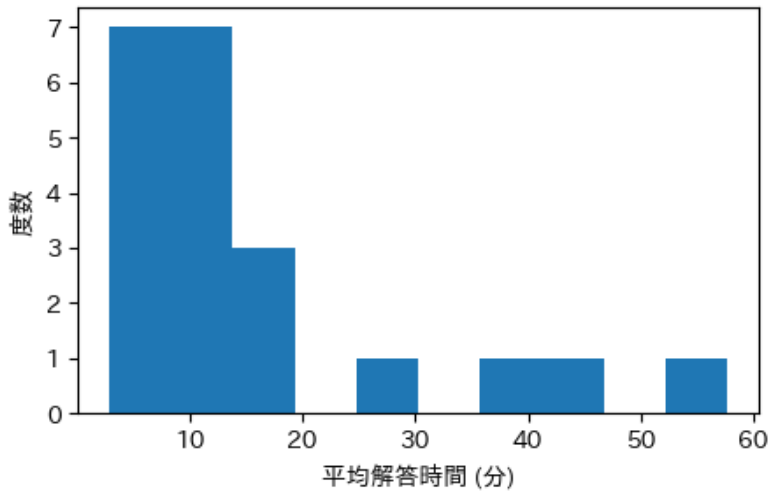


図 12. 正解した問題の平均解答時間

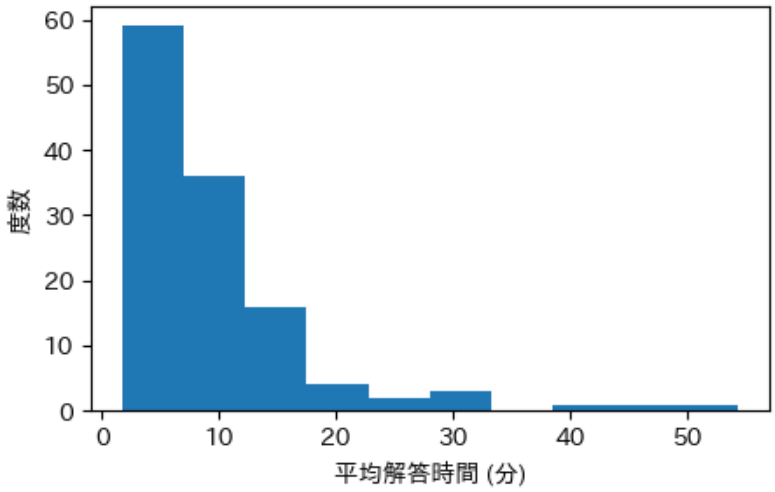


図 13. 正解した問題の平均解答時間（他校）

６．教員がプログラミング実習に要する準備・運営・事後処理等で費やした時間

AI 搭載オンラインプログラミング実習システムを用いて遠隔授業を行う際に削減できる教員の所要指導時間を評価するため、実験群と統制群のそれぞれに対して教員が講座の準備・運営・事後処理等に要する時間を計測した。その結果、講座 1 回分あたり、各群に対して以下の時間を要することが分かった。

- **実験群：** 授業予習 1 時間、課題の採点添削 0 時間、個別補習 2 時間
  （合計 3 時間）
- **統制群：** 授業予習 1 時間、課題の採点添削（5 問全員分）4.2 時間、個別補習 2 時間
  （合計 7.2 時間）

このうち、授業予習及び個別補習については本年度の実証講座の実施形態では計測できないため、昨年度の値を使用した。両群の差は講座 1 回あたり 4.2 時間（短縮率は約 58.3%）であり、本システムの提供する課題提示、誤り箇所提示、誤り修正方法の提示、類題提示、自動採点により削減できる所要指導時間として考えられる。なお、課題の採点添削にかかる時間は、教員が実際に手動で採点を行って計測した結果、1 問あたり平均で 2 分 30 秒であった。

7．考察

事後テストの点数を実験群と統制群の間で比較すると、実験群は統制群と比べて約 34%点数が向上した。また、図 1、図 2 のヒストグラムから、実験群では高得点の学生が統制群と比べて多いことがわかる。さらに、事後テストにおいて 5 問以上に正解した、最低限の実力をつけたとみなせる学生に絞って比較すると、両群の間に統計的に有意な差が確認された。なお、事前テストの段階では両群の間に有意な差は認められず、平均点も同程度であった。これは、AI 搭載オンラインプログラミング実習システムによる介入の結果として実験群の学習効果が高まり、統制群と比較して事後テストの点数が向上したことを示唆する。特に、一定以上の実力をつけた学生における点数は実験群のほうが高かったことに関しては、表 3、表 4 において、実験群のうち教員による評価スコアが高い群の点数の伸びが大きいことからも支持される。すなわち、実力や学習意欲が高い学生は、AI 搭載オンラインプログラミング実習システムにより大幅に実力を向上できると考えられる。

一方、図 4 における誤り箇所推定 AI の使用率と事後テストの点数の関係の分析から、誤り箇所推定 AI を多く使用した学生ほど事後テストの点数が低い傾向にあることが示された。因果関係については断定できないものの、誤り箇所推定 AI が強力であったため、一部の学生は宿題の解答において AI に依存しすぎてしまい、理解が十分ではないまま学習を終えてしまった結果、事後テストの点数が想定よりも向上しなかった可能性が考えられる。

宿題の提出状況においては、提出数・正解数いずれも実験群が統制群を大きく上回っており、統計的な有意差も確認された。また、図 5 のヒストグラムから、統制群では宿題の提出数や正解数が低い学生が多いのに対し、実験群では多くの学生が多数の問題を提出および正解していることがわかる。AI 搭載オンラインプログラミング実習システムによる動機促進 AI や誤り箇所推定 AI を中心とした学習を補助する機能に加え、自動採点による即時のフィードバックなどにより、学習意欲や学習効率が向上した結果、実験群の宿題の解答を促進したと考えられる。また、表 6、表 7 から、実験群のうち教員による評価スコアが低い群においても宿題の提出・正解数が多いことがわかる。このことから、実力や学習意欲の評価が低い学生でも、AI 搭載オンラインプログラミング実習システムのサポートにより、自主学習を多く行えていると考えられる。

事後アンケートの結果のうち、表 8、表 9 に示した本講座に対する評価の質問の回答では、ほとんどの項目において実験群のほうが統制群より良い結果となっている。また、表 10 に示した AI 搭載オンラインプログラミング実習システムの AI 機能の評価に関する質問では、多くの学生が「とても良い」「やや良い」に相当する回答をしており、実習システムの AI 機能の使用状況や、学習意欲の向上、学習のサポートにおいて、良好な評価を得ている。特に誤り箇所推定 AI の評価が非常に高く、学生の自主学習を強力にサポートできていると考えられる。

事後アンケートの自由記述では、AI 機能やシステムのユーザビリティ、問題の正誤判定に関する回答があった。好意的な意見が多い一方で、システムの改善点についてもいくつか触れられている。これらのうち、誤り箇所推定 AI の提示内容の質に関しては、実運用を重ねていきデータが蓄積されるにつれて、より改善されていくことが見込まれる。

情報科学専門学校および静岡産業技術専門学校における結果の分析でも、河原電子ビジネス専門学校の実験群と類似の傾向がみられ、AI 搭載オンラインプログラミング実習システムの効果を確認でき

る。なお、表 6、表 7 における、実験群のうち教員による評価スコアが低い群においても宿題の提出・正解数が多い結果については、他校においては異なる傾向がみられたため、この点における結果の解釈は慎重に行う必要がある。

本システムの導入により削減できる教員の所要指導時間は、講座 1 回あたり 4.2 時間であった。通常の講義であれば、1 単位あたり 10 回～15 回の講義を行うことになるため、42～63 時間ほど短縮できることになると考えられる。これにより教員の負荷が低減されることで、学生個別の指導などの本来時間をかけるべき領域に注力できるようになることが期待される。

2.2.3. 見学いただいた委員の意見

実証講座を見学いただいた委員の評価コメントを記載する。

◆廣井委員

・レスポンスなども非常によくストレスなく授業が進められており本格稼働に十分耐えうるシステムになっていたのではないかと感じている。

◆井坂委員

・全体的に授業態度もよく、寝ている学生はほぼ見当たらず、良い雰囲気だった。
・統制グループと実験グループがバラバラに混在していたのは、同じ空気感のもと授業を受けているので、検証するのにバイアスがかからず、良いと感じた。
・見た目の印象では、実験グループの方が、学習の進捗が良いような感じた。
・実験グループは、ヒント機能が使えるため、おどおどせずにスムーズに進めている感じを受けた。
・先生の講義中は、講義画面（Zoom）はプロジェクタに投影し、学生はテキストを参照するという進め方であったため、なかには説明を聞かずに先にテキストで学習を進めている学生もいた。逆にいうと、そういう進め方もできるシステムになっているとも言える。
・システムは使い勝手もよく教材として武器になるとは感じたが、ネガティブな感想としては授業の中での実習の場合、課題の時間（5 分間でやって）が設定される。そのため、時間内に課題をこなせば良いと考える学生に対しては、ヒント等の閲覧回数が増え、課題はこなせるが身にならないということも考えられる。そこで、講師の判断により、ヒントを与えるタイミングが変更できるようになると、より教育効果が高まると感じた（ある時間まではヒントを出さず考えさせ、何分か後にヒントを提示など）。
・実験グループの何人かの学生に聞いたところ、「痛いところに手が届くシステムで良いと感じる」という感想が得られた。

◆伊藤委員

・学生個々の進捗のバラツキを AI システムがうまく吸収している。
・手も足も出ない学生が観察した範囲では存在せず、苦手気味の学生も手が止まらずに取り組んでいた点はたいへん好感がもてる。
・slack 上での質問のやり取りは当事者以外から見ても良い気づきを得られそうに思う。ただ、活用する学生は少なかったので、コミュニケーション手段については工夫の余地がありそう。
・学校を超えて受講しているので、せっかくなら、学生もカメラオンで授業を実施したほうが、臨場感があってよかったのではないかと思う。あとは冒頭で各校の担当教員にマイクを回して、実況中継みたいな演出をしても面白いかもと思った。

2.2.4. 事前心理アンケート・事後アンケート

１．事前心理アンケート

事前心理アンケートは、学生の個性に応じて動機づける AI の入力データとして用いるため、実証講座の前に各学生に回答を求めた。1～16 の質問には「あてはまる」「ややあてはまる」「どちらともいえない」「ややあてはまらない」「あてはまらない」の 5 件法で選択肢を設け、17～21 の質問には下記の選択肢を設けた。

1. 何かにチャレンジするとき、成功をイメージすることが多い。
2. 私はどちらかというと落ちこぼれだ。
3. 自分が目立つチャンスがあれば、目立ちたい。
4. 将来役に立つことは、今役にたたなくても、やりたいと思う。
5. やりたいと思ったことは、すぐに取りかかる。
6. 人をバカにしたことは一度もない。
7. 他の人が自分のことをうわさしていると、とても気になる。
8. テストや評価を受けるとき、悪い結果にならないか心配になる。
9. 欲しいものを手に入れるために努力しようと思う。
10. まわりの人にどう思われているか、気になる。
11. みんなで何かをやるよりも、ひとりでやるほうが好きだ。
12. 上手くいくかどうかよりも、失敗しないかどうかの方が気になる。
13. 自分はダメな人間だと思うことがよくある。
14. 他の人がやるかどうか迷っていることでも、自分からやるほうだ。
15. いつだって、みんなから注目されていたい。
16. 何かに取り組むときは、ほかの人と協力したいと思う。
17. どのメッセージを読むと、もっとがんばろうと思いますか？
    a. 「もっとがんばれば成功できるよ」
    b. 「もっとがんばらないと失敗するよ」
    c. わからない
18. どのメッセージを読むと、もっとがんばろうと思いますか？
    a. 「おめでとう、友人 A さんに勝ったぞ！」
    b. 「おめでとう、前回よりも良い結果だぞ！」
    c. 「おめでとう、目標を達成したぞ！」
    d. わからない
19. 友人 A さんと競争した結果負けたときに、どのメッセージを読むと、もっとがんばろうと思いますか？
    a. 「友人 A さんに勝てなかったよ。」
    b. 「友人 A さんに勝てなかったよ。残念！目標達成ならず！！」

c. 「友人 A さんに勝てなかったよ。次は頑張ろう！」
d. 負けたことを表示しない
e. わからない

20. あなたの年齢をお答えください。
a. 6〜9 才
b. 10〜12 才
c. 13〜15 才
d. 16〜19 才
e. 20〜29 才
f. 30〜39 才
g. 40〜49 才
h. 50〜59 才
i. 60 才以上
j. 回答しない

21. あなたの性別をお答えください。
a. 男性
b. 女性
c. その他
d. 回答しない

２．事後アンケート

事後アンケートの設問のうち、25〜32 は AI 搭載オンラインプログラミング実習システムに関するものであるため、実験群のみに回答を求めた。各設問の回答の選択肢は、「実証成果集計・分析」を参照のこと。

1. 各回で取り扱った内容は理解できましたか。[第 1 回（制御構文）]
2. 各回で取り扱った内容は理解できましたか。[第 2 回（コレクション基礎）]
3. 各回で取り扱った内容は理解できましたか。[第 3 回（コレクション応用）]
4. 各回で取り扱った内容は理解できましたか。[第 4 回（クラス）]
5. 各回の講義で扱った問題の難易度はいかがでしたか。（宿題を除く）[第 1 回（制御構文）]
6. 各回の講義で扱った問題の難易度はいかがでしたか。（宿題を除く）[第 2 回（コレクション基礎）]
7. 各回の講義で扱った問題の難易度はいかがでしたか。（宿題を除く）[第 3 回（コレクション応用）]
8. 各回の講義で扱った問題の難易度はいかがでしたか。（宿題を除く）[第 4 回（クラス）]
9. 各回の宿題で扱った問題の難易度はいかがでしたか。[第 1 回（制御構文）]
10. 各回の宿題で扱った問題の難易度はいかがでしたか。[第 2 回（コレクション基礎）]

11. 各回の宿題で扱った問題の難易度はいかがでしたか。[第 3 回（コレクション応用）]
12. 各回の宿題で扱った問題の難易度はいかがでしたか。[第 4 回（クラス）]
13. 各回の講義と宿題は、Python プログラミングの理解度を向上させるのに役立ちましたか。[第 1 回（制御構文）]
14. 各回の講義と宿題は、Python プログラミングの理解度を向上させるのに役立ちましたか。[第 2 回（コレクション基礎）]
15. 各回の講義と宿題は、Python プログラミングの理解度を向上させるのに役立ちましたか。[第 3 回（コレクション応用）]
16. 各回の講義と宿題は、Python プログラミングの理解度を向上させるのに役立ちましたか。[第 4 回（クラス）]
17. 講義と宿題で扱った問題の分量は適切でしたか。
18. 講師の説明や講義資料はわかりやすかったですか。
19. 本講座に意欲的に取り組めましたか。
20. 本講座では、 他の講座と比べて自主的に学習する時間は増えましたか。
21. 本講座を通じて、Python プログラミングに対する興味を今までよりも持てたように感じましたか。
22. 今回使用したシステムの使いやすさはいかがでしたか。
23. 今回使用したシステムは学習を進めるうえで役に立ちましたか。
24. 今後、本講座以外でも、今回使用したシステムを使って学習したいと思いますか。
25. 問題に正解したときに表示されるポイント情報（レベルやランキングなど）は、どの程度見ていましたか。
26. 問題に正解したときに表示されるポイント情報（レベルやランキングなど）により、学習の意欲は上がりましたか。
27. ヒント機能（誤り箇所の特定・修正）はどの程度使っていましたか。
28. ヒント機能（誤り箇所の特定・修正）は学習を進める上で役に立ちましたか。
29. 類題表示機能はどの程度使っていましたか。
30. 類題表示機能は学習を進める上で役に立ちましたか。
31. ステップ実行機能はどの程度使っていましたか。
32. ステップ実行機能は学習を進める上で役に立ちましたか。
33. 今回使用したシステムについて、意見や要望などがあればご記入ください。
34. その他、講座全般に対する意見や要望などがあればご記入ください。

2.2.5. 事前・事後テスト

事前テストおよび事後テストは、問題文で指定された仕様を満たすように、途中まで書かれたプログラムに追記する形でソースコードを記述する問題とした。

１．事前テスト

問 1
異なる整数 a と b のうち、大きいほうの数を出力せよ。 a と b は標準入力から 1 行ずつ与えられる。

```
a = int(input())
b = int(input())
# ここに解答を入力
```

問 2
整数が格納されているリスト li の 0 番目（先頭）の要素と 1 番目の要素の和を出力せよ。 ただし、追記するプログラムの中では、li の中に格納されている値（8 など）を直接記載せず、li の要素を添え字で参照すること。

```
li = [8, 4, 7, 2, 5, 9]
# ここに解答を入力
```

問 3
1 から 100 の整数のうち、3 の倍数または 5 の倍数である整数を出力せよ。 各整数につき 1 行ずつ出力すること。

```
for i in range(1, 101):
    # ここに解答を入力
```

※ range(1, 101) は、1 から 100 までの整数が順番に格納されたリスト [1, 2, (中略), 99, 100] を作成する処理である。

問 4
3 の n 乗（n は任意の整数）のうち、10000 を超える最小の整数を出力せよ。

```
value = 3
while value < 99999:
```

```
    # ここに解答を入力
    value *= 3
print(value)
```

問 5

整数が格納されているリスト li の各要素を先頭から順番に見ていき、正の値なら p、負の値なら n と出力せよ。 P または n は、li の各要素につき 1 行ずつ出力すること。

```
li = [2, -1, -4, 3, 8, 5, -9, 8, -4, 5]
for element in li:
    # ここに解答を入力
```

問 6

複数の正の整数が標準入力から 3 10 2 4 7 のように空白区切りで与えられる。 与えられた整数のうち最大値を出力せよ。

```
li = [int(x) for x in input().split()]
ans = 0
for value in li:
    # ここに解答を入力
print(ans)
```

※ li = [int(x) for x in input().split()] は、問題文に記載された形式の入力を読み込んで、各整数を要素に持つリスト li を作成する処理である。 例えば、標準入力から 3 10 2 4 7 を読み込んだ場合は、li は [3, 10, 2, 4, 7] となる。

問 7

整数が格納されているリスト li に対して、指定した位置の要素を末尾に移動するという操作を複数回行って、簡易的にリストをシャッフルしたい。 移動する要素の添え字が 2 0 1 3 のように空白区切りで標準入力から与えられるので、各添え字に対応する li の要素に対して順番に操作を行った後のリストを出力せよ。

例えば li が [2, 5, 8, 4, 6] のとき、標準入力から 2 0 が与えられたとする。
li の状態：
[2, 5, 8, 4, 6]

まず、添え字 2 の要素 (li[2]) の 8 を末尾に移動すると li は [2, 5, 4, 6, 8] となる。
[2, 5, 4, 6, 8]

```
       |        ^
       |        |
       +--------+
```

次に、添え字 0 の要素 (li[0]) の 2 を末尾に移動すると li は [5, 4, 6, 8, 2] となる。

```
[   5,    4, 6, 8, 2]
 |                ^
 |                |
 +----------------+
```

したがって、最終的なリストの状態は [5, 4, 6, 8, 2] である。

```
li = [2, 5, 8, 4, 6]
pos = [int(x) for x in input().split()]

for p in pos:
    # ここに解答を入力
print(li)
```

※ pos = [int(x) for x in input().split()] は、問題文に記載された形式の入力を読み込んで、各添え字を要素に持つリスト pos を作成する処理である。 例えば、標準入力から 2 0 1 3 を読み込んだ場合は、pos は [2 0 1 3] となる。

問 8

ユーザ名とそのユーザの得点のペアが 5 個与えられる。 同じユーザ名に対応するペアが複数回出現することもある。 各ユーザの得点の合計を出力せよ。

ユーザ名と得点は、以下の例のように 2 行で一つのペアとして与えられる。

```
Alice
5
Bob
3
Alice
2
Bob
3
Carol
8
```

上記の例に対する出力は以下のようになる。

```
('Alice', 7)
('Bob', 6)
('Carol', 8)
```

```
d = {}
for i in range(5):
    name = input()
    number = int(input())
    # ここに解答を入力
for item in d.items():
    print(item)
```

※ range(5) は、0 から 4 までの整数が順番に格納されたリスト [0, 1, 2, 3, 4] を作成する処理である。

問 9

プログラムを実行したときに以下のような出力になるように、クラス MeasureClass を定義せよ。

```
170cm
54.6kg
```

```
# ここに解答を入力

height = MeasureClass(“cm”)
print(height.measure(“170”))

weight = MeasureClass(“kg”)
print(weight.measure(“54.6”))
```

問 10

プログラムを実行したときに以下のような出力になるように、クラス HelloClass を定義せよ。

```
Hello
```

ただし、 HelloClass は以下の制約を満たすこと。

・GreetingBaseClass を継承する

・message メソッドをオーバーライドする
・greet メソッドはオーバーライドしない

```
class GreetingBaseClass:
    def greet(self):
        print(self.message())

    def message(self):
        pass

# ここに解答を入力

hello = HelloClass()
hello.greet()
```

２．事後テスト

問 1～問 5 が基本問題、問 6～問 16 が発展問題である。

問 1

a から b を引いた結果を出力せよ。a と b は標準入力から 1 行ずつ与えられる。

```
a = int(input())
b = int(input())

# ここに解答を入力
```

問 2

整数が格納されているリスト li の各要素を先頭から順番に見ていき、正の値のみ 2 倍した値を表示せよ。なお、負の値のときは何も表示しないこと。なお、負の値のときは何も表示しないこと。
リスト li は初期プログラムによって定義されたものを使うこと。

```
li = [2, -1, -4, 3, 8, -9]
for element in li:
    # 以下の pass を削除してから解答を入力
    pass
```

問 3

ある学生のテストの点数が格納されている辞書 dict がある。辞書のキーと値は、それぞれ科目名と点数を意味する。 国語の点数を 15 点減らし、理科の点数を 84 点として要素を新たに追加した後、dict から各科目の値を取得して平均点を出力せよ。
辞書 dict は初期プログラムによって定義されたものを使うこと。 なお、国語の点数に直接 70 を代入してはいけない。

```
dict = {"国語": 85, "算数": 56, "社会": 90}
total = 0
count = 0

# ここに解答を入力

print(total / count)
```

問 4

整数が格納されているリスト li から、指定した添字の要素を取り出し、取り出した値が偶数なら末尾に移動して、奇数なら 10 を足して元の位置に戻す。 取り出す要素の添字が、2 0 1 のように空白区切りで標準入力から与えられるので、各添字に対応する li の要素を順番に操作した後の最終的なリストを 1 行で出力せよ。
リスト li は初期プログラムによって定義されたものを使うこと。

```
li = [1, 2, 3, 4, 5]
pos = [int(x) for x in input().split()]

# ここに解答を入力

print(li)
```

問 5
プログラムを実行したときに以下のような出力になるように、Greeting クラスを定義せよ。
正しいプログラムを記述した場合、以下の結果が出力される。

```
Good morning at 7:00
Good evening at 21:00
```

```
class Greeting:
    # 以下の pass を削除して解答を入力
    pass



morning = Greeting("7:00")
print(morning.greet("Good morning"))
evening = Greeting("21:00")
print(evening.greet("Good evening"))
```

問 6
start から end の整数のうち、整数 n で割り切れるものを、小さい順に 1 行ずつ出力せよ。
ただし、 start < end かつ n > 0 とする。 start, end, n は標準入力から 1 行ずつ与えられる。

```
start = int(input())
end = int(input())
n = int(input())
for i in range(start, end + 1):
```

```
    # 以下の pass を削除してから解答を入力
    pass
```

問 7

任意の正の整数 m が与えられたとき、3 の n 乗（n は整数）が m を初めて超えるときの n の値を求めよ。
m は標準入力から与えられる。

```
value = 3
n = 1
m = int(input())
while value < 99999:

    # ここに解答を入力

    value *= 3
print(n)
```

問 8

ある学生のテストの点数が格納されている辞書 dict がある。辞書のキーに科目、値にその科目の点数が格納されている。
文字列 subject で指定された科目の点数に、整数 score の値を加算せよ。 もし subject で指定された科目が辞書 dict に含まれなければ、文字列 subject を科目名、整数 score を点数として、辞書 dict に要素を追加せよ。
その後、全科目の科目名と点数を次の形式で出力せよ。

```
科目 1 の科目名
科目 1 の点数
科目 2 の科目名
科目 2 の点数
:
```

辞書 dict は初期プログラムによって定義されたものを使うこと。 文字列 subject と整数 score は標準入力から 1 行ずつ与えられる。

```
dict = {"国語": 85, "数学": 56, "理科": 38, "社会": 78, "英語": 90}
subject = input()
score = int(input())
```

```
    # ここに解答を入力
```

問 9

空のリストに対して、いくつかのクエリが以下の形式で与えられる。

- `-1` 処理を終了する。
- `0 x y` リストの添字 `x` の要素の位置に `y` を挿入する。
- `1 x` リストの添字 `x` の要素を取り除く。
- `2 x y` リストの添字 `x` の要素を `y` に置き換える。

これらのクエリを処理するプログラムを記述せよ。なお、入力は全て整数で標準入力から与えられる。
各クエリに従ってリストを操作したとき、リストの途中状態がどうなっているかをクエリごとに出力せよ。

```
li = []
while True:
    query = [int(x) for x in input().split()]
    q = query[0]
    # ここに解答を入力
    print(li)
```

問 10

あるクラスの出席状況が格納されている辞書 `dic` がある。キーに学生の苗字、値に出席状況が格納されている。 初期状態では、各学生の出席状況は未登録となっている。
出席した学生と欠席した学生の苗字が、以下のようにそれぞれ空白区切りで標準入力から与えられる。

出席した学生の苗字 `1` 出席した学生の苗字 `2` `...`
欠席した学生の苗字 `1` 欠席した学生の苗字 `2` `...`

辞書 `dic` に格納されている値を、出席した学生は出席、欠席した学生は欠席と変更した後、各学生の苗字と出席状況のペアを出力せよ。
もし、存在しない学生の苗字が与えられた場合は、その学生については特に操作を行わないこと。また、入力で与えられてない学生については、出席状況を未登録のままにしておくこと。
なお、出席した学生と欠席した学生の苗字は、それぞれ重複しないことが保証されている。

※ students = input().split()は、問題文に記載された形式の入力を読み込んで、各文字列を要素に持つリスト students を作成する処理である。 例えば、標準入力から田中 山田を読み込んだ場合は、students は["田中", "山田"]となる。

辞書 dic は初期プログラムによって定義されたものを使うこと。

```
dic = {"田中": "未登録", "山田": "未登録", "林": "未登録"}
attendees = input().split()
absentees = input().split()
# ここに解答を入力
for item in dic.items():
    print(item)
```

問 11

ユーザ名とそのユーザの得点のペアが 5 個与えられる。 同じユーザ名に対応するペアが複数回出現することもある。
各ユーザにおいて、0 より大きい得点の個数を出力せよ。

※ range(5) は、0 から 4 までの整数が順番に格納されたリスト [0, 1, 2, 3, 4] を作成する処理である。

ユーザ名と得点は、標準入力から、以下の例のように 2 行で 1 つのペアとして与えられる。

```
Alice
1
Bob
3
Alice
2
Bob
-1
Carol
0
```

```
d = {}
for i in range(5):
    name = input()
    number = int(input())
```

```
        # ここに解答を入力
for item in d.items():
    print(item)
```

問 12

プログラムを実行したときに以下のような出力になるように、Unit クラスを定義せよ。
正しいプログラムを記述した場合、以下の結果が出力される。

```
1g = 1000mg
1g = 0.001kg
```

```
class Unit:
    def __init__(self, scale, unit):
        self.scale = scale
        self.unit = unit

    def output(self, num):
        return "1g = " + str(self.convert(num)) + self.unit

    # ここに解答を入力


small = Unit(1000, "mg")
print(small.output(1))
large = Unit(0.001, "kg")
print(large.output(1))
```

問 13

NumberList クラスは、リスト numbers を持つクラスである。 プログラムを実行したとき、以下のような出力になるように、NumberList クラスに以下の 3 つのメソッドを定義せよ。

- リスト numbers の末尾に指定した数値を追加する add メソッド
- リスト numbers から指定した数値を削除する delete メソッド
- リスト numbers を表示する print メソッド

なお、NumberList クラスの add メソッドを実装する際はリストの append メソッドを、NumberList クラスの delete メソッドを実装する際はリストの remove メソッドを使うと良い。

```
class NumberList:
    def __init__(self):
        self.numbers = []

    # ここに解答を入力


list1 = NumberList()
list2 = NumberList()

list1.add(1)
list1.add(4)
list1.add(3)
list1.delete(4)

list2.add(5)
list2.add(1)
list2.add(7)
list2.add(4)
list2.delete(7)

list1.print()
list2.print()
```

問 14

プログラムを実行したときに以下のような出力になるように、RichBox クラスを定義せよ。ただし、RichBox クラスは以下の制約を満たすこと。

- Box クラスを継承する。
- output メソッドはオーバーライドしない。
- head と tail メソッドをオーバーライドする。

正しいプログラムを記述した場合、以下の結果が出力される。

```
------
Python
------
```

```
======
Python
++++++

        class Box:
            def output(self, text):
                print(self.head(text))
                print(text)
                print(self.tail(text))

            def head(self, text):
                return "-" * len(text)

            def tail(self, text):
                return "-" * len(text)


        # ここに解答を入力

        box = Box()
        box.output("Python")

        print()

        rich_box = RichBox()
        rich_box.output("Python")
```

問 15

ある学生のテストの点数が格納されている辞書 scores がある。scores のキーと値は、それぞれ科目名と点数を意味する。

さらに、点数から評価（A+〜F）を決めるための辞書 grades がある。grades のキーと値は、それぞれ評価とその評価の最低点を意味する。 なお、辞書の要素は評価が良い順（評価の最低点の高い順）に並んでいる。 例えば、grades が{"A+": 90, "A": 80, "B": 70, "C": 60, "F": 0}のとき、100〜90 点は A+、89〜80 点は A、...、59〜0 点は F となる。

scores と grades を用いて、それぞれの科目に対する評価を以下の形式で出力せよ。科目と評価のペアは 1 行ずつ出力すること。

辞書 scores と辞書 grades は初期プログラムによって定義されたものを使うこと。

正しいプログラムを記述した場合、以下の結果が出力される。

国語は A
数学は C
理科は F
社会は B
英語は A+

```
scores = {"国語": 85, "数学": 66, "理科": 38, "社会": 78, "英語": 90}
grades = {"A+": 90, "A": 80, "B": 70, "C": 60, "F": 0}

# ここに解答を入力
```

問 16

整数が格納されているリスト li を昇順にソートしたい。ソートする手順は以下のように行う。

1. 添字 0 （先頭）の要素を確定させる
   リスト全体で最小の値を見つけ、現在の先頭と入れ替える。その後、print(li)を用いて、現在のリストの内容を表示する。
2. 添字 1 の要素を確定させる
   添字 1 以降の要素で最小の値を見つけ、現在の添字 1 の要素と入れ替える。その後、print(li)を用いて、現在のリストの内容を表示する。
3. リストの末尾に到達するまで繰り返す
   以下、添字 2 以降の要素についても同様に行う。

リスト li の各要素が標準入力から空白区切りで与えられるので、以上の手順でソートを行うこと。

```
li = [int(x) for x in input().split()]
for i in range(len(li)):
    # 以下の pass を削除してから解答を入力
    pass
```

2.2.6. 教員向け運営マニュアル

実証講座の運営手順概要

事前準備

1. https://online-judge-production.willbooster.dev/ にアクセスする
2. 問題・科目・講義データを作成する
3. 学習者のアカウントを発行する
4. 学習者を科目に追加する
5. 学習者にアカウント情報およびシステムURLを配布する

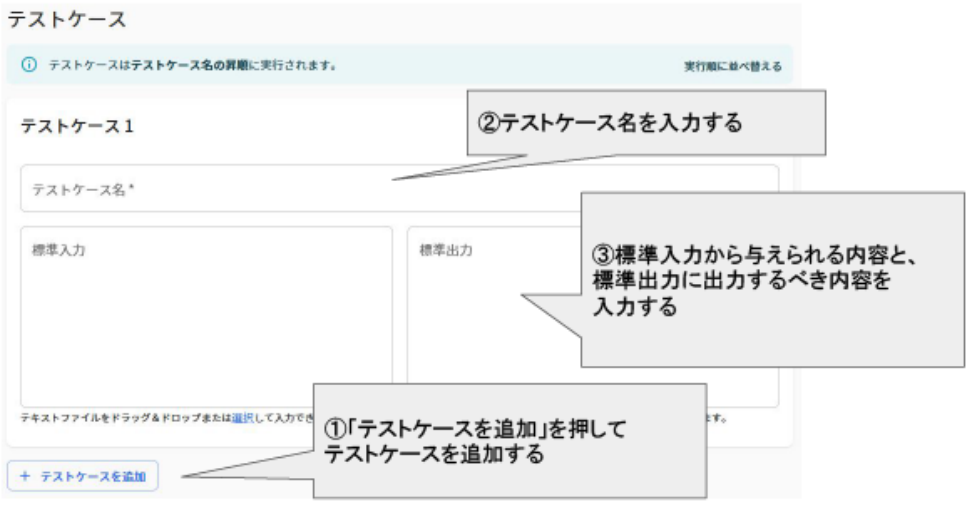
運営中の学習者の管理

- 学習者の進捗を確認する
- 答案のステータス・ソースコードを確認する

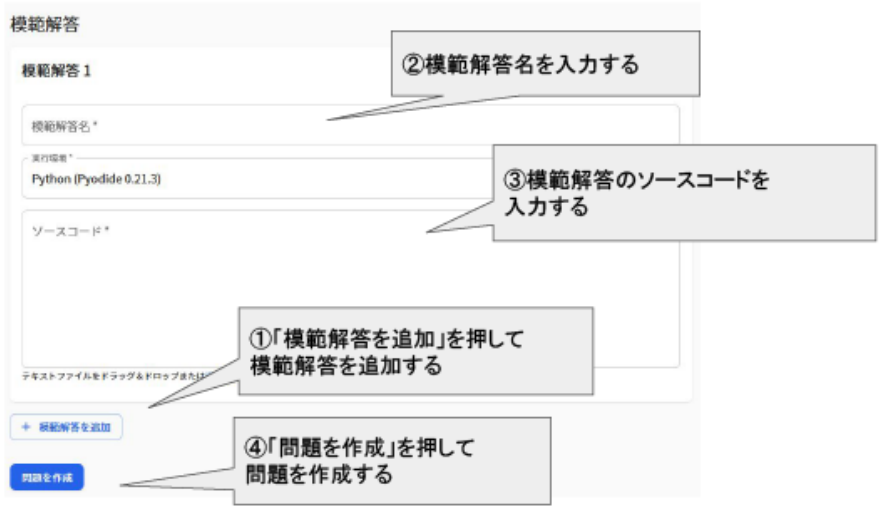
問題作成（1/5）

問題作成(2/5)

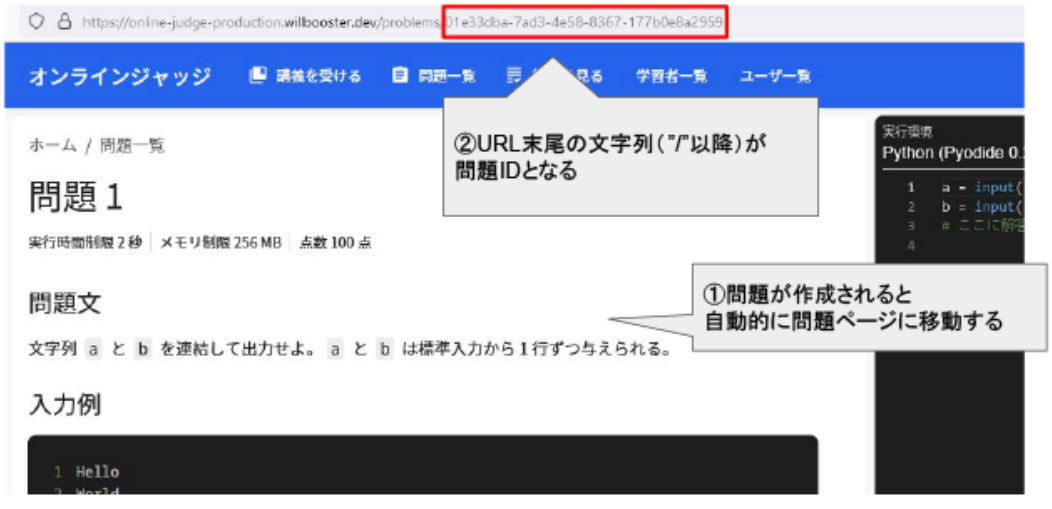
問題作成(3/5)



問題作成(4/5)



問題作成(5/5)



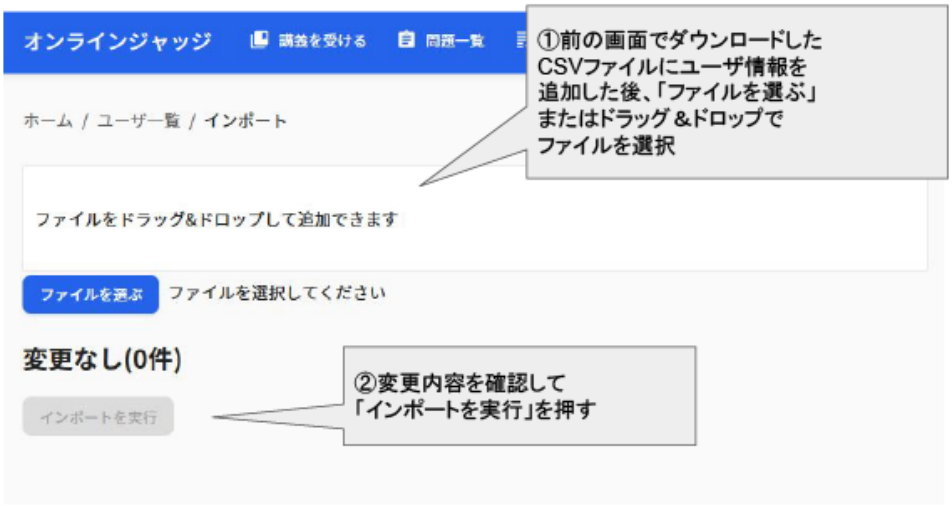
科目作成(1/3)

科目作成(2/3)

科目作成(3/3)

受講者のアカウント発行(1/2)

受講者のアカウント発行（2/2）

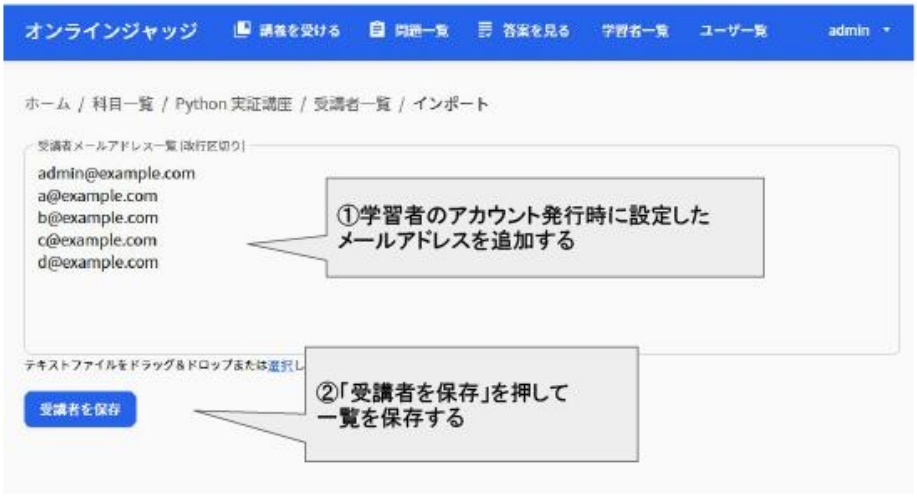


科目への学習者追加（1/4）

科目への学習者追加(2/4)

科目への学習者追加(3/4)

科目への学習者追加(4/4)



学習者の進捗状況の確認

答案の確認(1/2)

答案の確認(2/2)

2.2.7. 学生向け参加マニュアル

実証講座への参加方法

1. https://online-judge-production.willbooster.dev/ にアクセスする
2. 事前に配布したアカウント情報でログインする
3. 「科目一覧」→「Python 実証講座」または講義資料に記載されているURLから問題のページにアクセスする
4. 次ページ以降の操作方法に従って問題に解答する

オンラインジャッジとは

- Web上でプログラミング、プログラムの実行、提出、正誤判定まで、演習に必要なことを一通り実施できる。

問1

文字列 a と b を連結して出力せよ。a と b は標準入力から1行ずつ与えられる。

プログラム

```
a = input()
b = input()
# ここに解答を入力
```

入力例

```
Hello
World
```

出力例

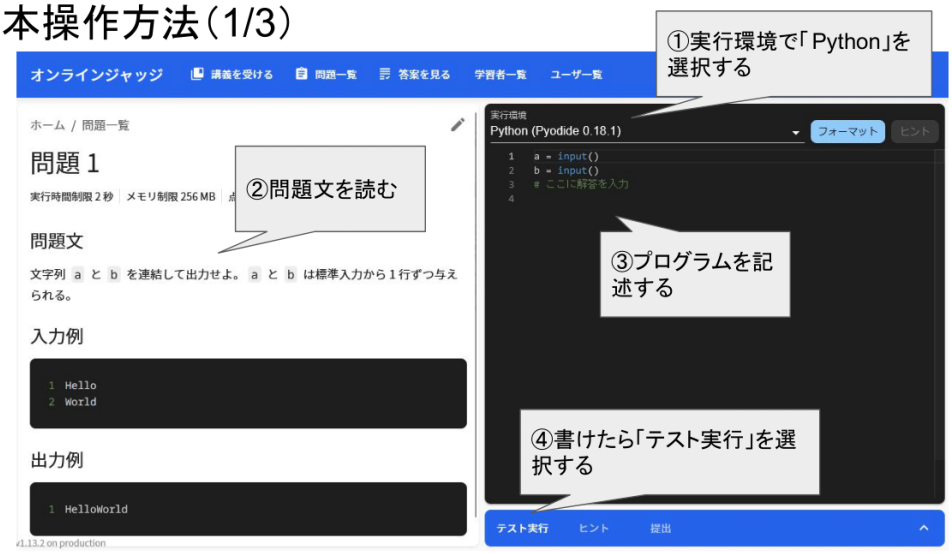
```
HelloWorld
```

正解プログラム

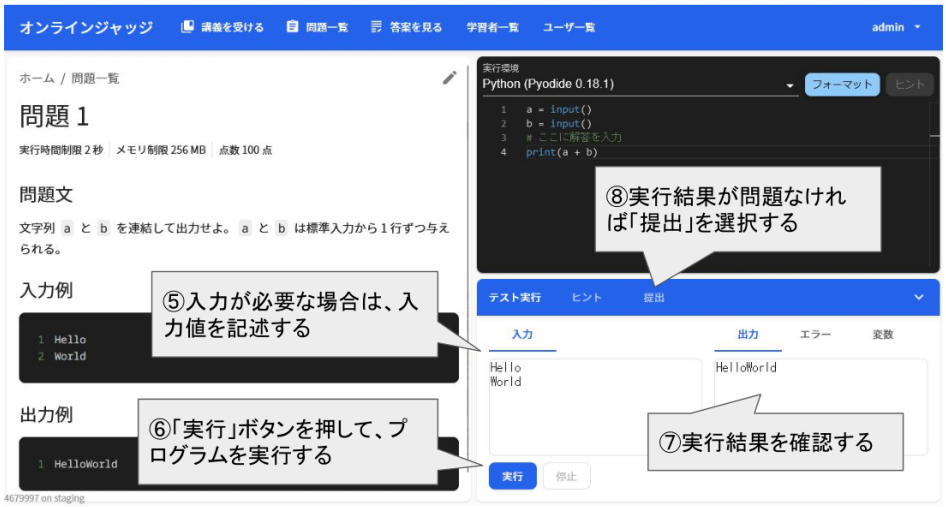
```
a = input()
b = input()
print(a + b)
```

問題URL
https://online-judge-production.willbooster.dev/problems/01e33dba-7ad3-4e58-8367-177b0e8a2959

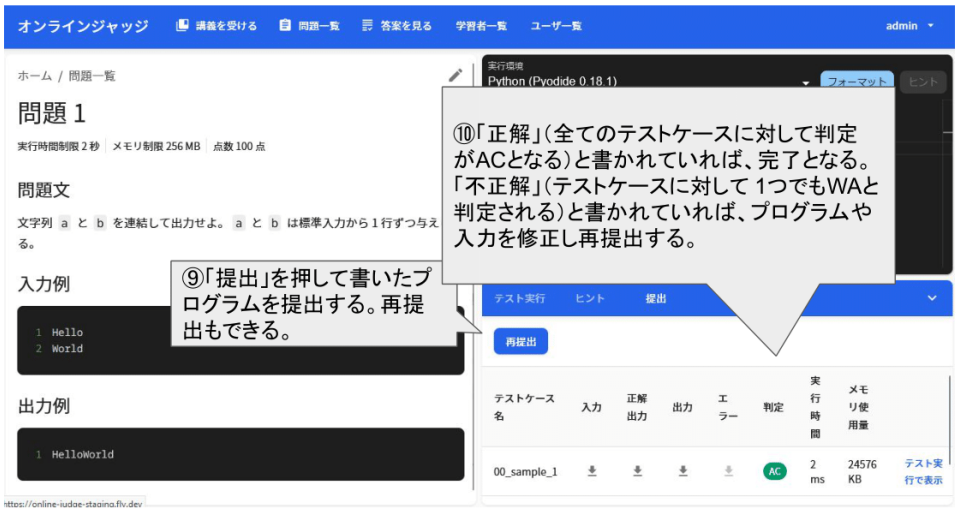
基本操作方法(1/3)



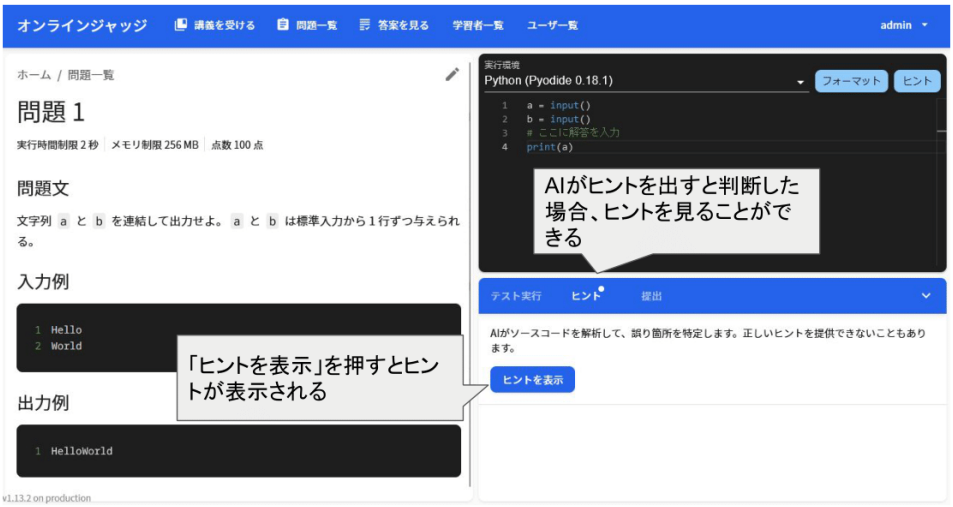
基本操作方法(2/3)



基本操作方法(3/3)



ヒントについて



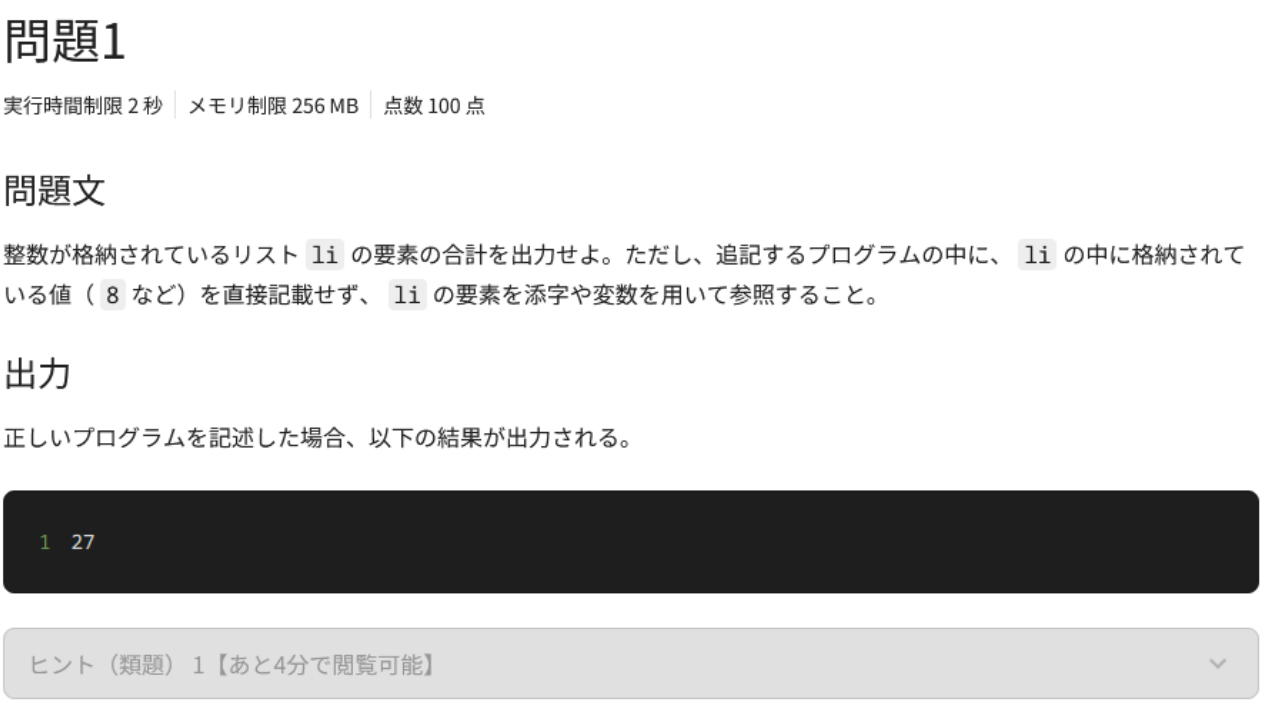
エラーについて

2.3. システム開発

2.3.1. システム開発の成果

【課題に応じて模範となるコードを提示する AI の調整】

前年度の段階では、課題に応じて模範となるコードを提示する AI が提示した模範コードについて、学習者はいつでも閲覧することができた。しかし、問題によっては模範解答に近いコードが提示されるため、学習時に一定時間が経過しないと閲覧できないように改良した。また、単に問題ページを開いたタイミングからの経過時間ではなく、問題ページ上でユーザが何らかの操作を行っている合計時間を用いて、模範コードを閲覧できるか否かを判定できるようにした。下図の最下部が模範コードを表示するためのユーザインタフェースとなっており、操作をしている合計時間が 5 分を超えるまでは閲覧できないようになっている。



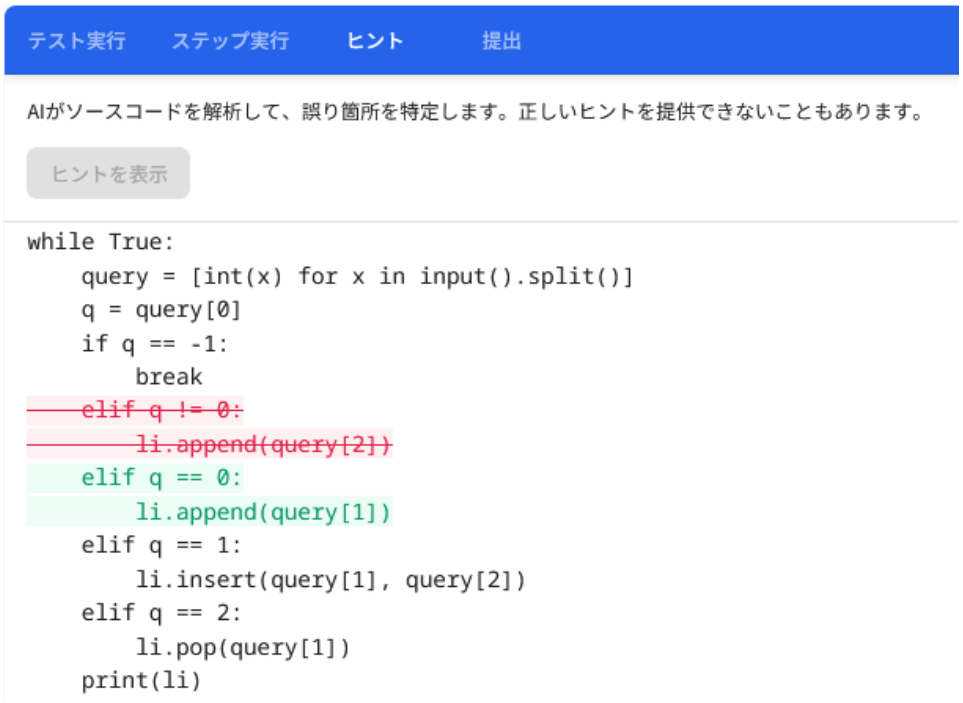
課題に設定された模範コードの例（画面下部、一定時間経過前）

また、実証講座を通して、問題によっては、解答を考える上で参考にならない模範コードが提示されるケースがあることが分かった。そこで、AI が提示する模範コードだけでなく、教員が自ら作成した類題および模範コードを入力できるような仕組みを追加した。さらに、ChatGPT 等の生成 AI を活用することで、教員が類題や模範コードを作成する際の手間を省けることを確認した。下図は、ChatGPT を活用して作成した類題および模範コードをシステム上で表示させている様子を示す。課題に応じて模範となるコードを提示する AI と生成 AI を組み合わせることで、実証講座の範囲内では、どのような問題であっても、参考になる類題や模範コードを提供できることを確認した。

課題に設定された模範コードの例（一定時間経過後）

【不正解の原因・誤り箇所を推定する AI の調整】

前年度の段階では、不正解の原因・誤り箇所を推定する AI は、誤り箇所が 1 行の場合のみ動作していた。推定アルゴリズムを改良することで、下図のように 2 行以上の誤りに対処できるように AI を改善した。さらに、本 AI を活用することで、学生の答案に対して部分点を与えられるように改善した。

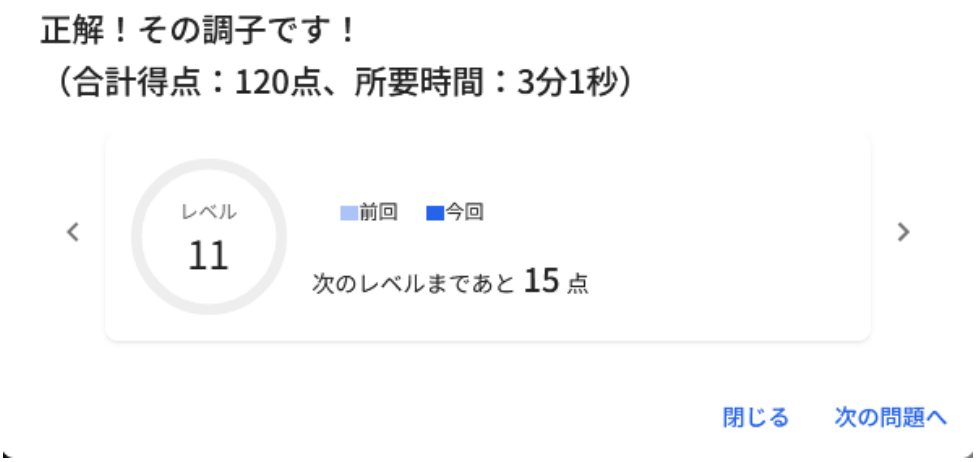


2 行以上にまたがる誤り箇所の修正例

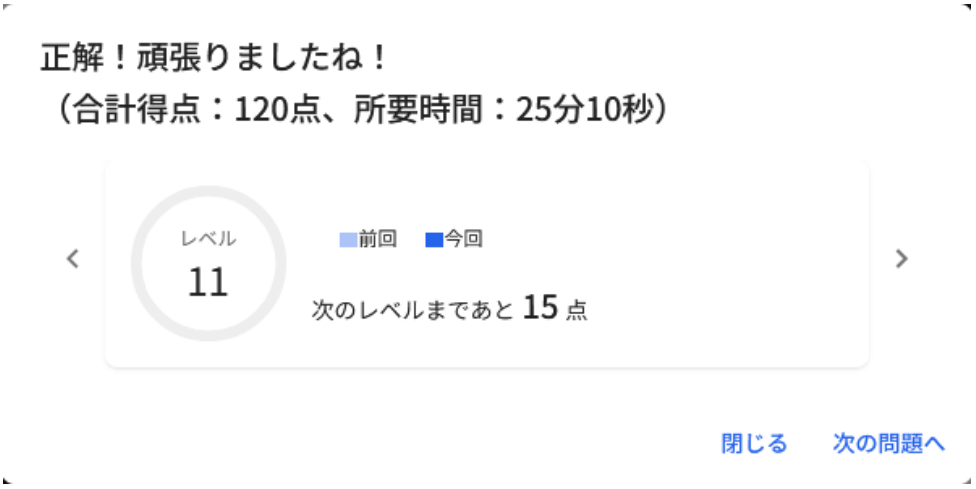
【学生の個性に応じて動機づける AI の調整】

前年度の実証講座で蓄積した学習データに加え、今年度の実証講座を開始するまでの期間で、他校においてシステムの試験運用を行って学習データを蓄積することで、より多くの学習データをもとに正確な動機付け促進要素を推薦できるようにした。また、前年度の実証講座において「レベルアップ」「自己ベストランキング」「対ボットランキング」に加えて新たに「対人ランキング」を追加したが、特に追加分の「対人ランキング」について十分な学習データが集まっていなかった。前述の試験運用による学習データの蓄積により、「対人ランキング」ついても十分な学習データをもとに推薦できるようにした。

また、問題に正解した時に表示する動機促進ダイアログについて、学習者に対してより高頻度に情報の確認を促す工夫と、学習者の学習状況に応じたメッセージの変化を加えた。まず、前年度の実証講座において、動機促進ダイアログの内容をほとんど見ずに閉じている学生が多数見受けられたため、ダイアログに次の問題へのリンクを表示して導線に組み込むことにより、問題正解後に動機付けダイアログの内容を毎回確認しやすくなるように誘導した。また、問題を解くのにかかった時間と不正解の回数に応じて、正解を褒めるメッセージの内容を変化させることで、学習者の努力に応じて動機付けを行えるようにした。



短時間かつ少ない不正解で正答したときのダイアログ



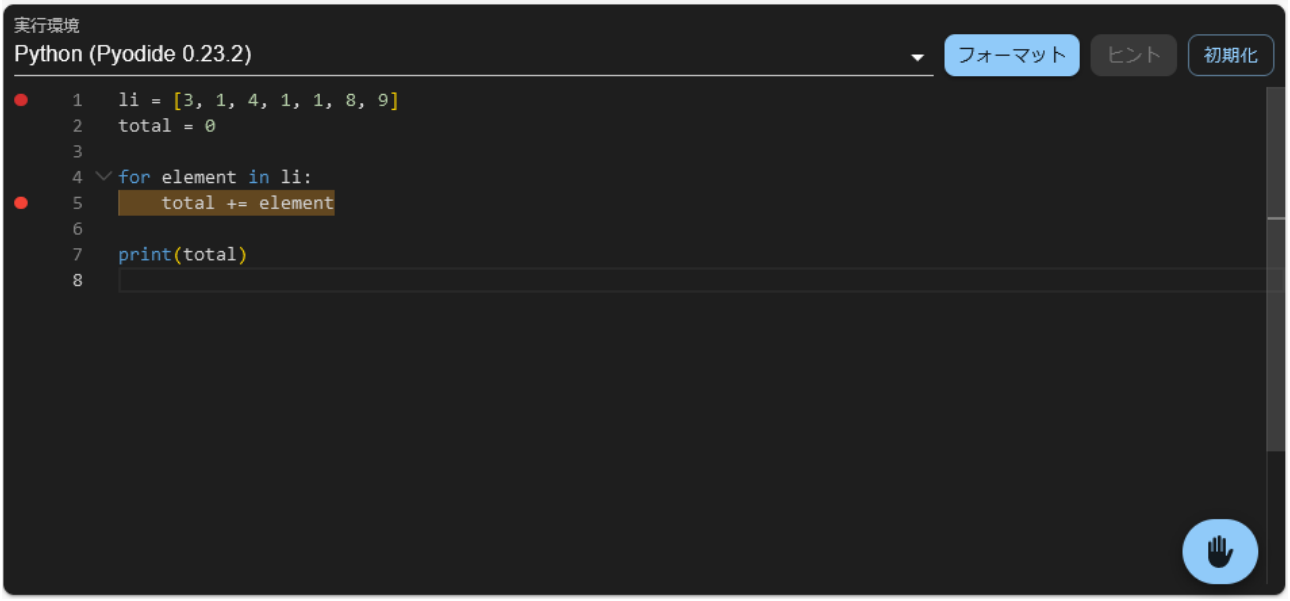
長時間または多くの不正解で正答したときのダイアログ

【採点を完全に自動化する AI】

前年度の段階では、採点を完全に自動化する AI を定量的に評価する仕組みがなく、AI を改善しても、どの程度改善できたかを評価することが難しかった。そこで、AI に与えた問題のうちテストケースを自動生成できた問題の割合、さらに、生成したテストの質を測る上でラインカバレッジ（プログラムを構成する行のうち、テストで実行した行の割合）を測定する仕組みを実装した。その上で、AI が利用するテストケースの自動生成アルゴリズムおよびそのパラメータをチューニングした。最終的に、94 問のうち 67 問（71%）にテストケースを自動生成でき、ラインカバレッジの平均値が 96%となった。ラインカバレッジの値は 100%に近ければ近いほど、採点ミス（本来は不正解なのに誤って正解と判断してしまう状況）が減ることが分かっており、十分にテストの質が高いことを示す。

【ステップ実行機能】

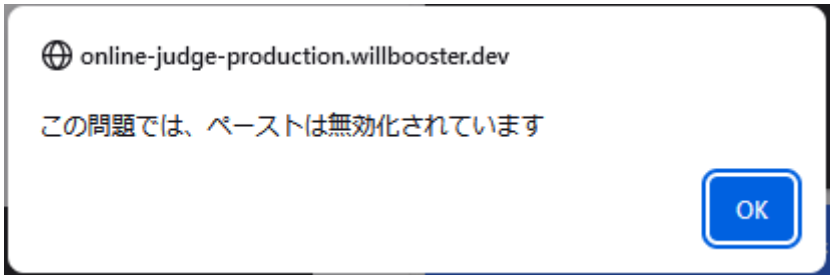
プログラムをデバッグする際は、実行途中の変数の値を考えながら、プログラムをトレースする必要があり、実行後の各変数の値だけでは不十分である。そこで、プログラムをステートメント単位で実行した際の各変数の値を記録しておき、プログラム実行後に、記録した各変数の値と実行したステートメントの位置を再生できるようにした。1 行ずつプログラムを実行し、現在着目しているステートメントをハイライトすることで、直前・直後のステートメントの位置を確認したり、該当ステートメント実行後の各変数の値を確認したりできるようにした。学習者は本機能を使うことで、プログラムの実行をトレースでき、効率的にデバッグできるようになる。その他、Visual Studio Code (https://code.visualstudio.com/) を参考にして、同ソフトウェアが備えているデバッグ機能も搭載した。



ステップ実行中の画面例

【剽窃防止機能】

本システムのエディタ上では、基本的にコピー&ペーストができるため、剽窃が容易である。また、システムで扱う課題はシンプルなものが多い上、AI による自動修正機能が存在するため、どの学習者も提出するプログラムがほとんど同じ内容になり、内容や作業履歴だけでは、偶然同じなのか、剽窃しているのか区別することは困難である。そこで、剽窃を防ぐために、システム上のエディタでペースト（貼り付け）を禁止する機能を作成した。ただし、学習者がシステム以外のエディタ（例えば、Visual Studio Code）でプログラムを作成して、システムに投稿することを許可したいケースもある。その場合、ペーストを禁止することはできないので、問題ごとにペーストを禁止するか否かを設定できる仕組みを搭載した。なお、ペーストが禁止されているときにカット（切り取り）を行うと、その後のペースト操作ができず、ソースコードが意図せず削除されてしまうため、ペーストを禁止した場合はカットも禁止とした。



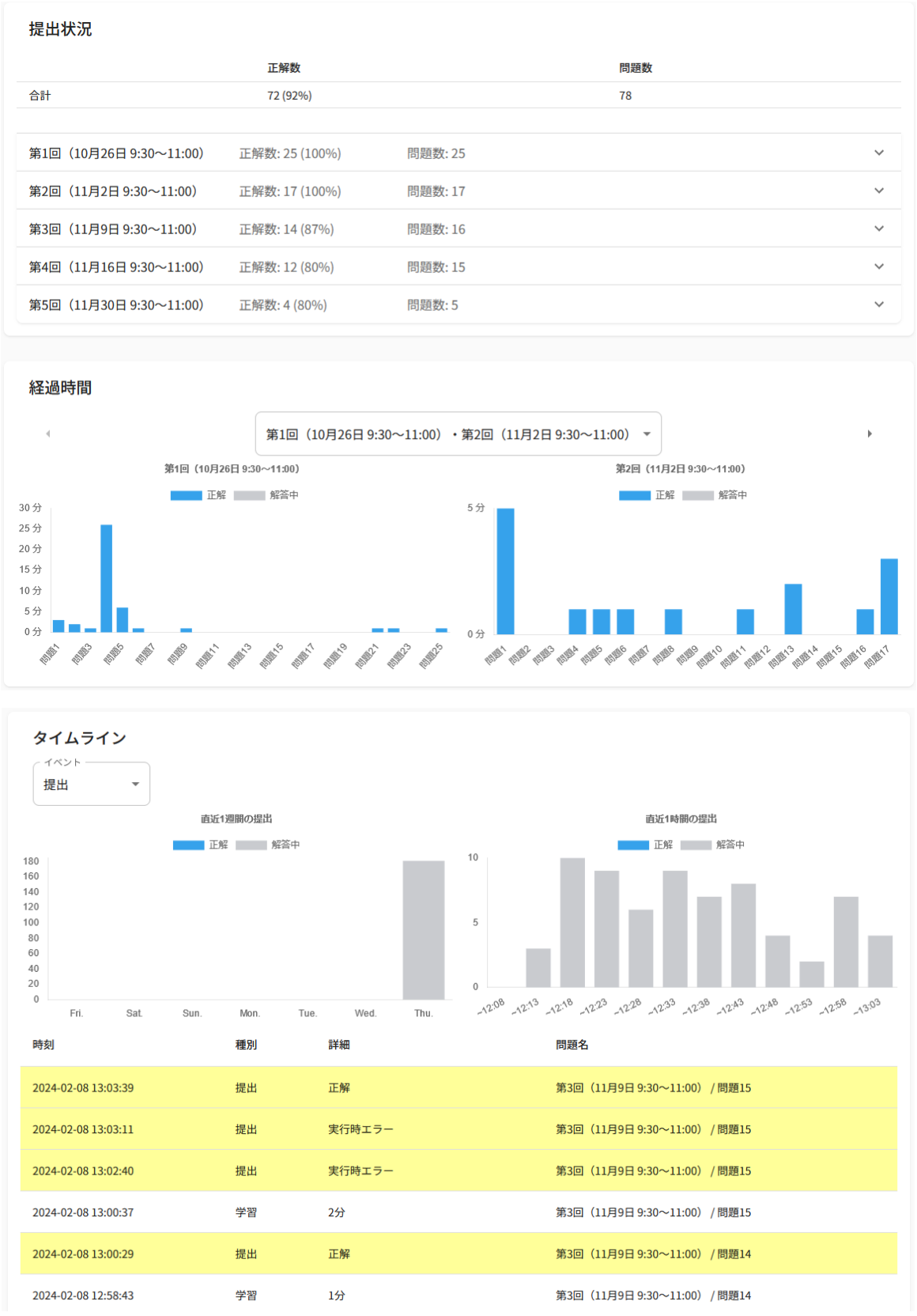
ペーストが禁止されている問題でペースト操作を行ったときに表示されるダイアログ

【学習状況の分析・可視化機能】

前年度の段階では、全学習者のプログラムの提出履歴の一覧を表示したり、各学習者が各問題に正解しているか否かの表を表示したりできた。しかし、それだけでは、学習者全体の進捗が良好なのか否かを判断したり、学習に詰まっている学習者を特定したりすることは困難である。そこで、学習者の学習状況を管理する画面により、学習の進捗状況を俯瞰する仕組み、および、学習に詰まっている学習者を特定するための仕組みを搭載した。

各学習者について、現在取り組んでいる問題の学習状況や、誤り箇所推定 AI・類題提示 AI などの使用状況、科目内の問題の進捗状況などを俯瞰して確認できる。また、学習者の氏名をクリックすることで、該当する学習者の詳細な学習状況を確認できるようにした。具体的には、特定の期間において問題を投稿した日時および投稿頻度を可視化したり、各問題の解答時間の一覧を可視化したりできるようにした。

さらに、後述する「コミュニケーション支援機能」において、教育者からサポートが必要な学生を発見できるよう、挙手をしている学生を表示するようにした。



特定の学習者の詳細な学習状況を確認する画面

【粒度の細かい評価機能】

学習意欲を高めるためには、学習者が学習の進展を実感することが重要であり、単に不正解という値だけでなく、100 点満点中の 50 点のような評価を下すことが必要である。また、正解となった場合であっても、プログラムの品質を高める余地が残っているケースが多々ある。そのような場合は、さらなる高みを目指せるように、120 点など 100 点満点を超えたような評価を下せることが望ましい。そこで、後述するプログラム評価基準案に従って、部分点や追加点を付与するようにした。

不正解プログラムを提出した際、正解プログラムに近い場合は部分点が付与される。その際、不正解のメッセージに加えて、部分点が付与された旨を表示するようにした。また、プログラムの品質を改善することで追加点がもらえる場合は、正解プログラムの提出時に改善点がある旨を表示し、クリックすると改善点が記載されたテキストファイルをダウンロードできるようにした。

正解に近い不正解プログラムにより部分点が付与されたときの表示

プログラムの品質改善による加点の余地があるときの表示

【コミュニケーション支援機能】

現状のオンライン授業では、Teams，Zoom，Slack，Discord など、様々なコミュニケーションツールを利用することで、教員が学習者とコミュニケーションを取っている。これらのツールは他の科目の授業でも導入しているため、プログラミング学習においても利用したい。しかし、前年度の段階のシステムには、これらのツールを活用して、教員と学習者のコミュニケーションを支援する仕組みは存在しない。そこで、学習者がプログラムエディタに表示されている挙手ボタンを押すことで、教員に

対して困っている旨を通知でき、教員は学習状況の管理画面に表示される「挙手している学生」から挙手の通知を受け取れるようにした。

コードエディタ上に表示される挙手ボタン（画面右下）

挙手している学生の一覧表示

【解答時間の測定機能】

問題ページを開いてから正答を投稿するまでの期間において、問題ページを閲覧していた時間を測定することで、問題ごとに解答に要した時間を測定する仕組みを搭載した。これにより測定した解答時間を、学習状況の分析・可視化機能における学習状況の可視化に使用した。また、課題に応じて模範となるコードを提示する AI において一定時間経つまではコードの表示を無効化する機能や、学生の個性に応じて動機づける AI における解答時間に応じた表示メッセージの変更にも活用した。

【ヒント無効機能】

前年度の段階のシステムでは、すべての問題においてヒント機能（誤り箇所推定 AI によるサポート）を利用できた。しかし、学習者に独力で解くことを求める問題も存在するため、剽窃防止機能と同様に、問題ごとにヒント機能の有効・無効を設定できる仕組みを搭載した。

2.3.2. プログラム評価基準案

委員会での議論を踏まえ、さらに、教育上の有用性と実装上の実現可能性を加味して、プログラム評価基準は、コーディング規約への準拠と不正解の原因・誤り箇所を推定する AI の 2 つの観点に基づいて評価値を算出する。その経緯および具体的な評価方法について、以下で詳細を述べる。

過去の委員会では PEP8 などのコーディング規約が取り上げられたが、実務の場においては、コーディング規約を遵守するために flake8 (https://github.com/pycqa/flake8) などのコードチェッカーが導入されている。こういったツールによるコーディング規約への準拠により品質を判定し、プログラムの評価を行うことは妥当だと考えられる。また、委員会において誤り箇所を推定する AI のデモを実施した際に、AI がプログラムの修正案を生成できることを応用して、正解プログラムに近い不正解プログラムに対して部分点を与えられないかという議論がなされた。AI がプログラムを修正できる場合、該当する不正解プログラムは正解プログラムに近いということが言えるため、それに応じて部分点を与えるというモデルは妥当だと考えられる。なお、昨年度はソースコードメトリクスも評価に組み込むことを提案したが、評価基準となる要素が増えて複雑になり、学習者の観点からどういった要因で点数が変動しているのか理解しづらくなるため除外した。

以上を踏まえて、本評価基準として追加点と部分点を採用する。まず、追加点を採用する理由として、コーディング規約への準拠は減点方式との相性が良いと考えられるものの、減点方式にするのは学習意欲の観点から好ましくないことに加え、軽微な違反を含めた全警告への対応を過度に推奨するのは学習者に大きな労力を要求し、本筋のプログラミング学習に支障をきたす可能性があることが挙げられる。追加点方式を採用することで、コーディング規約への準拠に対してインセンティブを与えつつ、過度な労力を割くことがないようにする。また、部分点を採用する理由として、正解に近い不正解プログラムを評価することで、学習者が正解する前に諦めてしまうことを防ぐことが挙げられる。従来のオンラインジャッジに類似するシステムでは、基本的に正解・不正解の 2 値で採点を行うことが一般的である。一方、部分点を導入することで、通常の不正解と正解に近い不正解を区別して、正解に近い不正解に対して加点を行うことができる。これにより、正解に近い不正解者の学習意欲を高めることを目指す。

追加点を実現するため、正解プログラムが提出された際、コーディング規約への違反がある場合は問題に設定されている満点分を与え、違反がない場合は満点分に加えて追加点を与える。なお、昨年度は正解プログラムへの近さや違反の数などを加味することを提案したが、前述したソースコードメトリクスと同様に、複雑な評価基準は学習者にとって理解が困難になるため、なるべく単純な基準となるようにした。また、部分点を実現するため、不正解プログラムが提出された際、誤り箇所推定 AI により修正の提案が可能かどうか判定し、可能な場合は正解に近い不正解であるため部分点を与える。

具体的な評価基準は以下の通りである。

- 正解で、flake8 による警告がない場合：120 点
- 正解で、flake8 による警告がある場合：100 点
- 不正解で、誤り箇所推定 AI によって修正コードの提示ができる場合：50 点
- 不正解で、誤り箇所推定 AI によって修正コードの提示ができない場合：0 点

2.4. 実施委員会委員による最終評価

最後に、本事業の成果物、および、実証講座結果に対する実施委員会各委員からの評価コメントを記載する。

◆廣井委員

・実験に参加した学生の声は概ね良好であり、今後、本研究を踏まえた遠隔授業の円滑な実施が期待される。
・一方、インターフェイスの改善などについて一部意見もあったことから研究を踏まえた課題等については、さらに検証していただき今後の改善につなげていただきたい。

◆赤松（民康）委員

・限られた条件の下、考えうる様々な条件の下のテスト、お疲れ様でした。
・結果は確信を持つほどの成果を上げたとは断言できるものではなかったかもしれませんが、トライアンドエラーにて今後の成長を成し遂げて頂きたいと思います。

◆赤松（正教）委員

・結果について統計的に有意な結果が出たかどうかはともかくとしても、意欲や自信など定量的な測定では現れない定性的な部分での効果が確認できた点が非常に興味深いと感じた。
・表面的な成績以上に教育効果が高いようなら、積極的に利用していく理由にはなり得ると思われる。
・また、教育現場における業務負担の軽減において効果が大きいことはこれまでも証明されていたし、改めて証明するまでもなく明らかではあるが人手不足が大きな社会問題となる中、優先順位は高まっており、こういった側面でも更なる広域な運用が期待される。

◆松浦委員

・本プロジェクトの成果物である実習システムは遠隔授業でのプログラミング実習における課題の多くを解決し、教育現場での高い実用性が期待できるものとなった。
・特に、スキル評価の低い生徒が早い段階で躓いてしまい実習を継続できなくならないよう適切に AI がヒントを与え、実習を完遂できるようフォローする仕組みは、レベルの異なる多人数を教育する専修学校において高い効果を発揮すると感じている。
・スキル評価の高低によってヒントの出し方を変えるなどの改善点は残っていると感じているので、今後も講師・生徒の意見を取り入れて改善を続け、より良いシステムへと発展していくことを期待している。

◆國廣委員

・目標にされていた「教員の業務負荷：60%削減」、「学生の学習効果：25%アップ」が実現できており、なにより学生アンケート「システムは学習の役に立ったか」、「システムを本講座以外でも使って学習したいか」で 90%近い学生が良い評価をしていたことから、本当に大きな成果のあった事業

だったと思う。
・一方、誤り箇所推定 AI については、評価が高かった半面、使いすぎると成績が落ちてしまうという副作用があり、バランス調整に課題があるように感じた。
・また、このシステムで蓄積した学生の学習結果や特性などを、就職活動にも活かしていけると面白いのかなと思う。

◆本田委員
・Python を利用したプログラミングの基礎から始まり、プログラムの設計の考え方であるデザインパターンも含み、ソフトウェア開発者が学ぶべき内容をおおよそ網羅できる内容と考えられる。
・また、応用科目についてはデータ分析をテーマとして、それに沿った内容でソフトウェアライブラリの具体的な利用方法を勉強できる大変すばらしい教材だと考えられる。

◆鈴木委員
・このプロジェクトの成果物は AI を駆使したヒント機能があることにより、普段わからなくて諦めてしまう学生が、集中してプログラムを書いている姿を見て、教員として驚きを覚えた。
・既存のサービスでは、躓いてしまうと諦めてしまう学生が多く、その後授業に乗り切れないといった状況になりがちであったが、このシステムを使えば大半の学生が自主学習、本当にできない学生のみを教員がフォローするといったことが可能になる。今後多くの方々にこのシステムを使っていただけることを期待したい。

◆伊藤委員
・今回のオンライン学習システムを活用することで、学習者にとっては以下のようなメリットがあり、学習成果の向上につながると感じた。
①習熟度上位層にとっては先生が解説するのを待つという「手待ち時間」がなくなるため、単位時間当たりの学習量を増やすことができる。また、「手待ち時間」が発生することによるモチベーション低下を防ぐ効果もあると考えられる。
②習熟度下位層にとっては躓きポイントを AI が支援してくれるため、投げ出さずに考える行動を促し、プログラミング科目にありがちな「全く理解できないままに授業を終える」というケースを減らす効果があると考えられる(実証実験のデータでもそのように出ていた)。
③習熟度上位層の学生同士が難問に取り組む際にクラスメートと意見交換する様子が見られ、インストラクショナルデザインを工夫すれば協調学習の活性化にもつながると感じた。
・また、今後の開発の際に、以下のような改善が行われることを期待している。
①上記のように近くの席の学生との協調学習は発生していたが、オンラインの特性を活かし、オンライン上での協調学習が活性化する仕組みを工夫できるとさらに良いのではないかと思う。
②継続学習を促すためにゲーミフィケーション的な機能を強化するとよいと思う。問題を解いていく過程をクエストのようにして「ノーヒントで正答」、「受講者の中で最初に正答」、「総学習時間が〇時間を突破」など達成感とワクワク感を感じられる仕組みがあると自習システムとしての発展も期待できる。

◆柿本委員

・本システムの導入による学習面での効果が実証されており、充実した意義ある取り組みになっていると感じる。

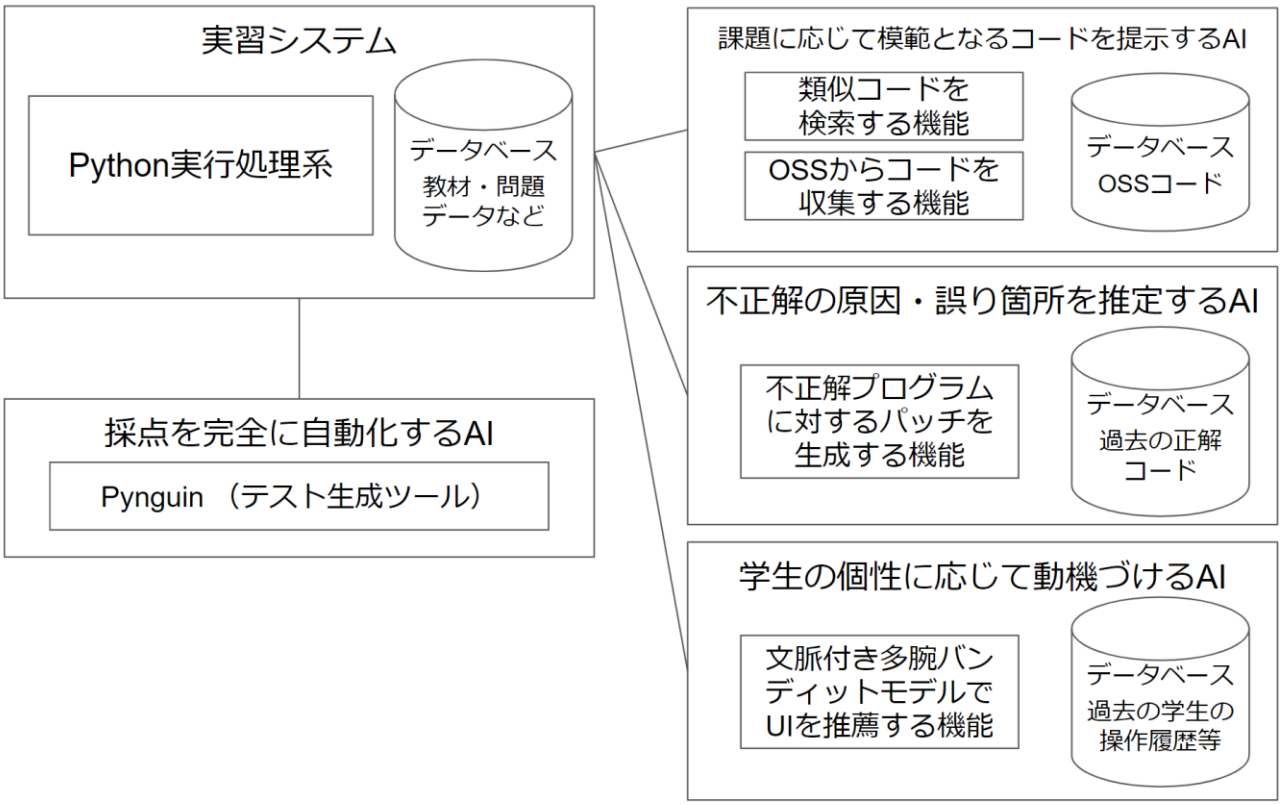
・次年度以降のブラッシュアップでは、動機促進のために、ぱっと見の印象で楽しそうに感じるような画面デザインや、学習者に適したゲーミフィケーションの導入などを検討いただくのも良いかと感じた。

・詳細な実証成果をご共有いただき、ありがとうございました。

3. AI 搭載オンラインプログラミング実習システムの開発技術・手法

1．ソフトウェアアーキテクチャの全体像および各構成要素の説明

下図で、ソフトウェアアーキテクチャの全体像および各構成要素を示す。



アーキテクチャの全体像

以下で、上図に現れる各構成要素の詳細を説明する。

- オンラインプログラミング実習を行う Web システム
  - 同システムは Python の実行処理系を内包しており、学生が提出した Python プログラムを実行して、問題で提示している仕様を満たしているか否かを自動判定する。また、同システムは教材や問題データを格納したデータベースを有しており、同データベースの内容を Web ページ上で表示できる。さらに、後述の AI 技術 1～3 と連動して、学習者に AI の出力結果を表示する機能を提供する。
- AI 技術 1：課題に応じて模範となるコードを提示する AI
  - 同 AI は GitHub などから OSS ソフトウェアのソースコードを収集する機能を有する。また、収集したソースコードを格納するデータベースを有する。同データベースを活用することで、与えられた問題の模範解答に類似するソースコードを出力できる。
- AI 技術 2：不正解の原因・誤り箇所を推定する AI
  - 同 AI は過去の正解プログラムを可能するデータベースを有する。同データベースを活

用することで、与えられた不正解プログラムに対して、正解できるようなパッチを生成する機能を提供する。

- AI 技術 3：学生の個性に応じて動機づける AI
  - 同 AI は過去の学習者の操作履歴を学習データとして、文脈付き多腕バンディットのモデルを学習する機能を有する。学習したモデルを用いることで、各学習者に対して最も学習の動機づけ効果が期待できる UI を推薦する機能を提供する。
- AI 技術 4：採点を完全に自動化する AI
  - 同 AI は後述するテスト自動生成ツール（Pynguin）を活用することで、模範解答プログラムから採点に必要なテストケースを自動生成する機能を提供する。テストケースの生成に必要な計算量・実行時間は比較的大きい・長いため、作問時にテストケースを生成しておき、事前に実習システムに登録する必要がある。

２．各 AI の開発方法

【AI 技術 1：課題に応じて模範となるコードを提示する AI】

Yoshimura らが論文誌で発表した論文「WOJR: A Recommendation System for Providing Similar Problems to Programming Assignments」[1] では、与えられたプログラミング課題の類題を推薦するためのアルゴリズムを提案しており、さらに、提案アルゴリズムによって類題および類題の模範解答を提示することで、大学のプログラミング講義における演習課題の正答率を改善できることを評価している。そこで、本 AI は上記論文において提案されている類題推薦アルゴリズムを実装した。なお、論文では、プログラミング課題を公開している Web サービスから問題を収集していたが、本事業ではそのような Web サービスが提供していない、より基礎的な問題を中心に扱っているため、適切な類題を見つけられなかった。そこで、類題のかわりに GitHub からオープンソースソフトウェア（OSS）のプログラムを収集して、模範プログラムと類似する OSS プログラムを提示することとした。

具体的な AI の処理手順を以下で示す。

1. AI が類似する OSS プログラムを探す問題の模範プログラム（プログラム A）を受け取る。
2. AI が保有する類似する OSS プログラム集において、プログラム A とそれぞれの OSS プログラムとの類似度を計算する。
3. AI が手順 2 で求めた類似度の高い順に OSS プログラムを出力する。

なお、問題内容によっては、十分に類似する OSS プログラムを発見できないケースがあった。そのような場合、教員が ChatGPT 等の生成 AI を活用することで、類題の問題文および模範プログラムを生成して、手動でシステムに類題として登録できるような工夫を凝らした。

【AI 技術 2：不正解の原因・誤り箇所を推定する AI】

Hu らが ASE で発表した論文「Re-factoring based program repair applied to programming assignments」[2] では、不具合のあるプログラムに対して、既知の正答プログラムとの差分を計算することで、正解となるようなパッチを自動生成する手法を提案している。提案手法は模範プログラムをリファクタリングすることで、学習者が提出した不正解プログラムに類似した模範プログラムを生

成し、できる限り小さなパッチを生成する工夫を凝らしている。しかし、このような工夫はパッチ生成に必要な計算時間を増やしてしまうため、学習者に対する UX としては優れておらず、また、システム運用費が増大するという問題がある。そこで、吉村の論文「オンラインジャッジにおける初学者向けプログラミング教育を支援する仕組みの提案」を参考にして、パッチの内容を 1 つのコード断片（昨年度は 1 行のみ、本年度は複数行）に限定することで、高速かつ安価にパッチを生成する手法を実装した。パッチを用いることで、学生が提出した不正解プログラムの誤り箇所を特定したり、正解する上で必要な修正内容を提示したりできる。

具体的な AI の処理手順を以下で示す。

1. 事前に、AI が各問題に対する模範プログラムおよび学生が提出した正解プログラムを受け取り、正規化した上で正解プログラムのデータベースに蓄積する。
2. AI が学生の提出した不正解プログラムを受け取る。
3. AI が変数名の匿名化などを用いて、学生の提出した不正解プログラムを正規化する。
4. AI が正規化した不正解プログラムとデータベースに記録されている正規化済みの各正解プログラムを比較して、1 箇所のコード断片の追加・削除・置換によって修正できることを確認して、修正できるケースにおいてできる限り小さなパッチを生成する。
5. AI パッチを実習システムに送付し、実習システムが誤り箇所や修正内容を可視化する。

【AI 技術 3：学生の個性に応じて動機づける AI】

Li らが WWW で発表した論文「A contextual-bandit approach to personalized news article recommendation」[3] および Saito らが NeurIPS で発表した論文「Open Bandit Dataset and Pipeline: Towards Realistic and Reproducible Off-Policy Evaluation」[4] を参考にして、各学習者の心理アンケートの結果に対して、多腕バンディットアルゴリズムにより最も動機づけ効果の高い画面を推定する AI を実装した。

具体的な AI の処理手順を以下で示す。

1. AI が学習者の心理アンケートの回答結果を受け取る。
2. AI が心理アンケートの回答結果をコンテキストとして、線形モデルを用いた文脈付き多腕バンディットアルゴリズムにより、最も効果の見込める動機づけ情報を推定する。
3. AI が推定した動機づけ情報を実習システムに送付し、実習システムが該当する動機づけ画面を学習者に表示する。
4. 実習システムは学習者が提示した動機づけ画面を操作したかどうかを記録して、AI に送付する。
5. 一定量の記録が溜まったら、文脈付き多腕バンディットのモデルを更新して、AI の精度を改善する。

【AI 技術 4：採点を完全に自動化する AI】

Lukasczyk らが ICSE で発表した論文「Pynguin: Automated unit test generation for python」[5] を参照して、Python の模範プログラムからテストケースを自動生成する AI を実装した。Pynguin は、様々な研究者によって提案されたアルゴリズムを実装したツールとなっており、Randoop, MIO, MOSA,

DynaMOSA などの著名なアルゴリズムを利用できる。そこで、これらのアルゴリズムを比較・検討し、本事業で開発した問題に適したアルゴリズムとして MIO を選び、該当アルゴリズムによってテストケースを自動生成する AI を実装した。

具体的な AI の処理手順を以下で示す。

1. AI がテストケースを生成する問題の模範プログラムを受け取る。
2. AI が Pynguin で処理しやすいように模範プログラムを自動変換する。
3. AI が Pynguin を用いて模範プログラムからテストケースを自動生成する。

参考文献

[1] Yoshimura, R., Sakamoto, K., Washizaki, H., & Fukazawa, Y. (2022). WOJR: A Recommendation System for Providing Similar Problems to Programming Assignments. Applied System Innovation, 5(3), 53.

[2] Hu, Y., Ahmed, U. Z., Mechtaev, S., Leong, B., & Roychoudhury, A. (2019, November). Re-factoring based program repair applied to programming assignments. In 2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE) (pp. 388-398). IEEE.

[3] Li, L., Chu, W., Langford, J., & Schapire, R. E. (2010, April). A contextual-bandit approach to personalized news article recommendation. In Proceedings of the 19th international conference on World wide web (pp. 661-670).

[4] Saito, Y., Aihara, S., Matsutani, M., & Narita, Y. (2020). Open bandit dataset and pipeline: Towards realistic and reproducible off-policy evaluation. arXiv preprint arXiv:2008.07146.

[5] Lukasczyk, S., & Fraser, G. (2022, May). Pynguin: Automated unit test generation for python. In Proceedings of the ACM/IEEE 44th International Conference on Software Engineering: Companion Proceedings (pp. 168-172).

4. カリキュラム

本事業で完成したカリキュラムを次ページ以降に掲載する。

5. シラバス

本事業で完成したシラバスを次ページ以降に掲載する。

6. 課題付きテキスト開発

令和 5 年度に作成した課題付きテキストを次ページ以降に掲載する。