令和４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

２０２３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４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
1.1.事業の趣旨と目的 1
1.2.事業の背景 1
1.2.1.プログラミング教育の重要性 1
1.2.2.プログラミング実習授業の課題 2
1.2.3.遠隔授業下での問題拡大 2
1.3.事業の運営・実施体制 4
1.3.1.委員会・作業部会からなる組織体制 4
1.3.2.委員会・作業部会の人員構成 5
1.3.3.各機関の役割・協力事項 6
1.3.4.実施委員会の開催 8
1.3.5.作業部会の開催 10
2.　事業活動の内容 12
2.1.遠隔教育の導入方策とそのモデル化の概要 12
2.1.1.課題解決のための４つの AI 12
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 12
2.1.3.遠隔教育の導入方策とそのモデル化の概要① 14
2.1.4.遠隔教育の導入方策とそのモデル化の概要② 15
2.1.5.遠隔教育の導入方策とそのモデル化の概要③ 16
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 20
2.2.1.実証概要 20
2.2.2.実証成果集計・分析 21
2.2.3.見学いただいた委員の意見 30
2.2.4.事前心理アンケート・事後アンケート 31
2.2.5.事前・事後テスト 34
2.2.6.運営・参加マニュアル 43
2.3.システム開発 56
2.3.1.システム開発の成果 56
2.3.2.プログラム評価基準案 61
3.　シラバス（プロトタイプ） .....
4.　課題付きテキスト開発 .....

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。

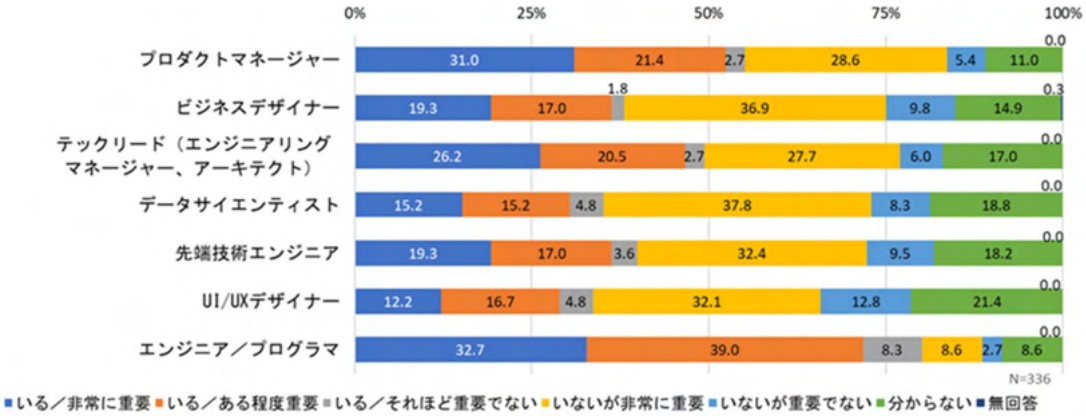


図 1. IT 企業における IT 人材種ごとの重要性の相違

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。

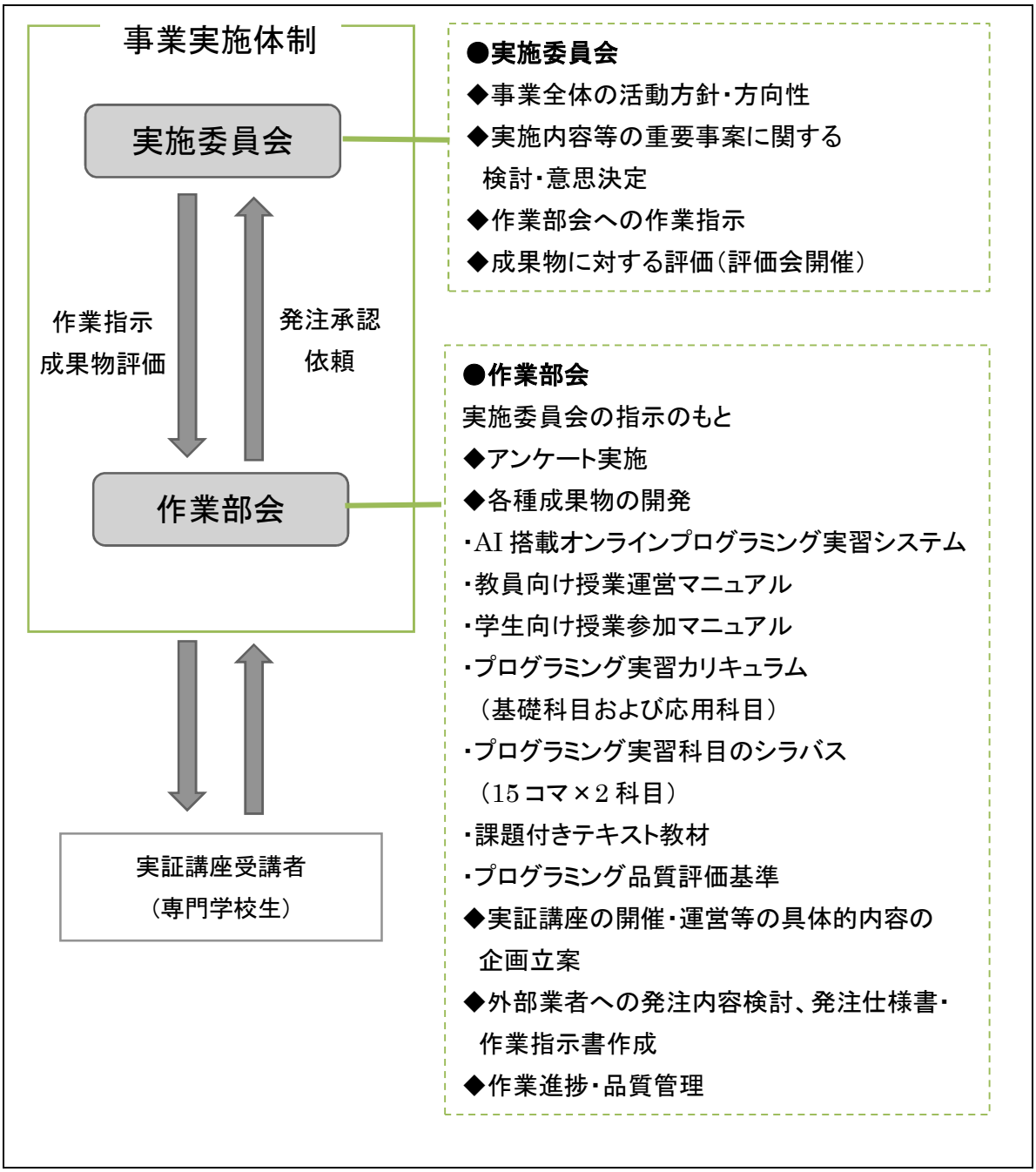
④教室と比較して開放的な学習環境（学生の自宅等）のため、学生の集中力の維持が困難になってしまった。

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	早稲田大学 理工学術院 基幹理工学部 プログラミング教育研究者	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	愛媛県情報サービス産業協議会　顧問	評価	愛媛県
15	藤本朋成	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	早稲田大学 理工学術院 基幹理工学部 プログラミング教育研究者	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	2022 年 7 月 29 日（金）16：00～17：30
場　所	対面：TKP 松山市駅前カンファレンスセンター 2F 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）、藤本朋成（愛媛県経済労働部産業支援局）、赤松民康（愛媛県情報サービス産業協議会）、赤松正教（一般社団法人　愛媛ニュービジネス協議会）、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（早稲田大学理工学術院基幹理工学部）、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）、大下貴央（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．事後連絡
配布資料	資料１　前年度実施委員会の評価結果と修正案 資料２　昨年度の成果と今年度の取り組み

◆第２回実施委員会

日　時	2022 年 11 月 4 日（金）16：00～17：30
場　所	対面：TKP 松山市駅前カンファレンスセンター 2F 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）、藤本朋成（愛媛県経済労働部産業支援局）、赤松民康（愛媛県情報サービス産業協議会）、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（早稲田大学理工学術院基幹理工学部）、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．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　現状課題一覧 資料２　河原電子ビジネス専門学校実証講座詳細案

◆第３回実施委員会

日　時	2023 年 1 月 31 日（火）16：00～17：30
場　所	対面：ホテルマイステイズ松山 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）、藤本朋成（愛媛県経済労働部産業支援局）、赤松民康（愛媛県情報サービス産業協議会）、赤松正教（一般社団法人愛媛ニュービジネス協議会）、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（早稲田大学理工学術院基幹理工学部）、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）、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．実証講座報告・開発進捗報告 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　実証講座報告 資料２　今年度開発課題付きテキスト教材　基礎 資料３　今年度開発課題付きテキスト教材　応用

1.3.5. 作業部会の開催

◆第１回作業部会

日　時	2022 年 7 月 21 日（木）16：30～18：00
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（早稲田大学理工学術院基幹理工学部）、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）、大下貴央（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　前年度実施委員会の評価結果と成果物の修正案 資料２　実習科目のシラバス案 資料３　AI 向けデータ収集に関する補足資料 資料４　発注仕様書案 資料５　参考書籍一覧

◆第２回作業部会

日　時	2022 年 10 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（早稲田大学理工学術院基幹理工学部）、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　課題一覧 資料２　河原電子ビジネス専門学校実証講座詳細案 資料３　河原電子ビジネス専門学校 Python 前期試験

◆第３回作業部会

日　時	2023 年 1 月 27 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（早稲田大学理工学術院基幹理工学部）、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　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）を導入する。

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する。

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む。

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 技術を組み込む。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 技術を組み込む。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にする。

④AI 技術 4：採点を完全に自動化する AI

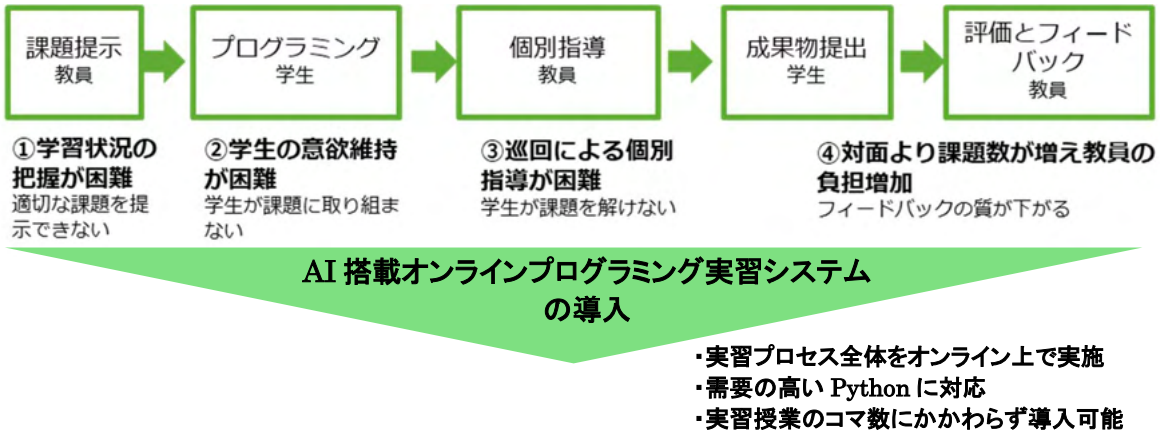
従来のオンラインジャッジは、プログラムを自動採点するために必要なテストケースを教員が用意する必要があり、テストケースの作成は容易ではないという問題があった。採点回数が十分に多ければ、採点コストがテストケースの作成コストを上回り、自動化によるコスト削減効果が出るものの、採点回数が少ないと、かえって採点の手間が増える問題がある。また、自動採点では正解・不正解の 2 値でしか評価できず、プログラムの品質を測定することはできない。そこで、ソフトウェア工学研究におけるテストケース生成技術を応用することで、与えられた模範解答からテストケースを自動生成する AI 技術を組み込み、さらに、ソフトウェアメトリク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

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）の解決をはかる。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

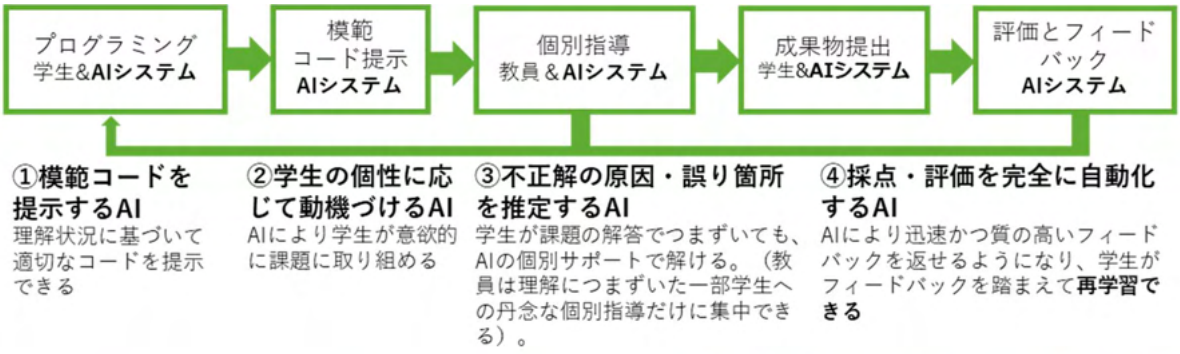
プログラミング実習は、プログラム言語の文法知識の暗記にとどまらない応用力を育成する活動で、プログラミング教育の中核をなす。対面授業では安定的に運営できたプログラミング実習授業のプロセスだが、遠隔授業下では 4 つの問題が生じている。



・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

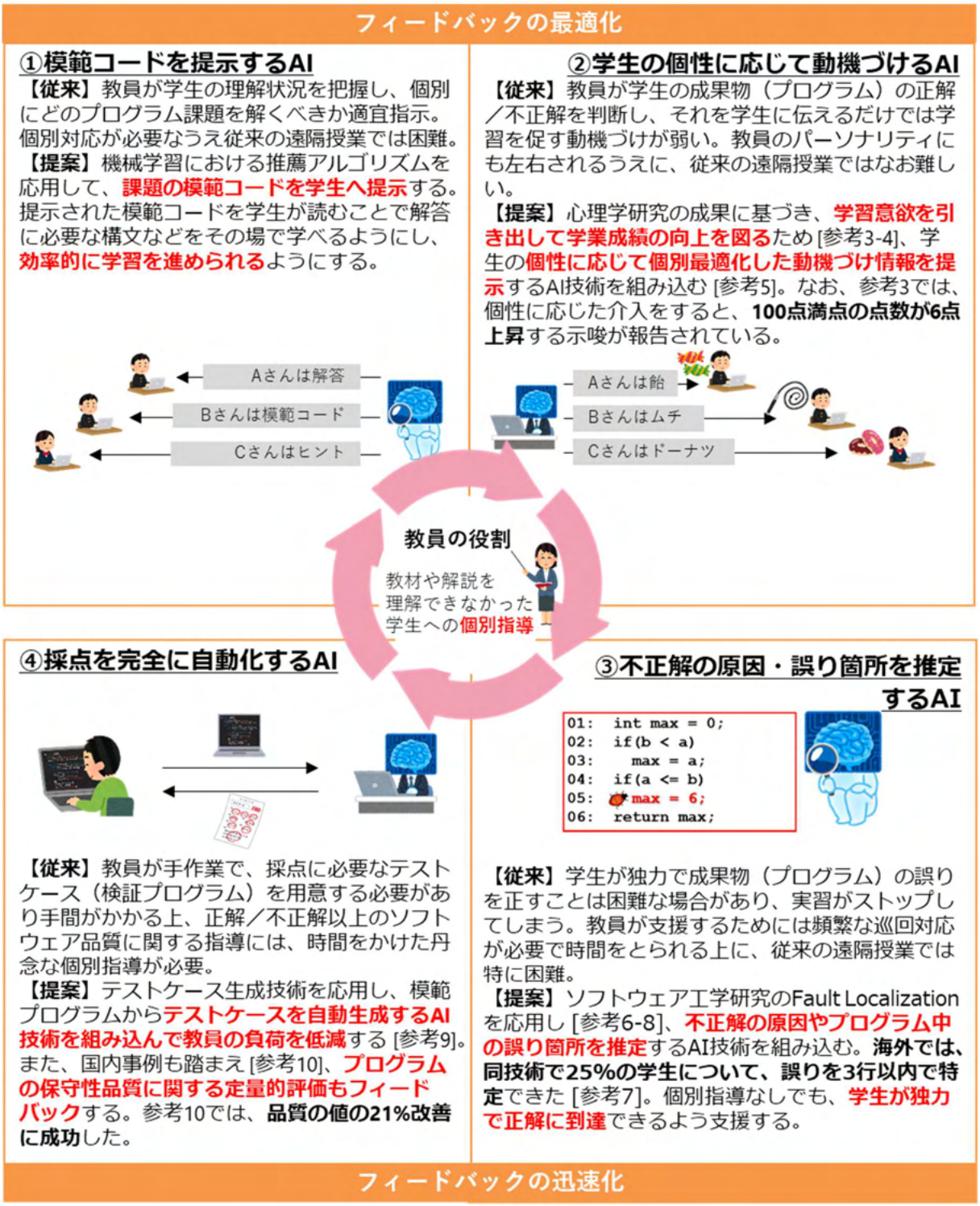
4 つの AI を搭載した実習システム・教材・課題を導入することで、前述の 4 つの問題を解決する。AI の迅速かつ最適化されたフィードバックにより、実習の質が向上する。さらに、教員は教材を用いた解説、および、解説を理解できなかった学生のサポートなどの本来あるべき個別指導に集中できる。



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 15 コマ、応用 15 コマから構成される。その内訳として、以下の内容を想定している。

なお、来年度でテキストを作成する応用 6 章以降は、今後内容を変更する可能性がある。

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

章	節
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
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 オブジェクト指向によらないコンソールアプリケーションの実装とその問題点 14.2 オブジェクト指向によるコンソールアプリケーションの実装 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 リファクタリング 15.2 継承を使ったコンソールアプリケーションの実装

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 1.5 NumPy 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 による欠損値や外れ値の確認 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 BeautifulSoup を用いた Web スクレイピング 5.2 pandas を用いた前処理 5.3 pandas を用いたデータ分析 5.4 プログラム作成課題
6 ソフトウェア開発のデザインパターン	6.1 デザインパターン 6.2 Singleton パターン 6.3 Decorator パターン 6.4 Strategy パターン 6.5 プログラム作成課題
7 Web アプリケーションの基本知識	7.1 Web アプリケーション 7.2 簡易的な Web アプリケーションの作成 7.3 Web アプリケーションフレームワーク 7.4 Python の Web アプリケーションフレームワーク「Django」 7.5 プログラム作成課題
8 Django を使った Web アプリケーション開発の準備 1	8.1 プロジェクトの作成 8.2 開発用サーバの起動 8.3 投票アプリの作成 8.4 ビューの作成 8.5 プログラム作成課題
9 Django を使った Web アプリケーション開発の準備 2	9.1 データベースの設定 9.2 モデルの作成 9.3 モデルの有効化 9.4 API 9.5 Django Admin 9.6 プログラム作成課題

章	節
10 ビューとテンプレートシステム	10.1 複数のビューの作成 10.2 404 エラーの送出 10.3 テンプレートシステム 10.4 URL 名の名前空間 10.5 プログラム作成課題
11 フォームと汎用ビュー	11.1 フォーム 11.2 汎用ビュー 11.3 プログラム作成課題
12 テスト	12.1 自動テスト 12.2 ビューのテスト 12.3 プログラム作成課題
13 アプリケーションのカスタマイズ 1	13.1 HTML 13.2 CSS 13.3 静的ファイルの変更 13.4 プログラム作成課題
14 アプリケーションのカスタマイズ 2	14.1 Django Admin フォームのカスタマイズ 14.2 アプリケーション用のテンプレートのカスタマイズ 14.3 プログラム作成課題
15 再利用可能なアプリケーション	15.1 再利用性 15.2 アプリの公開 15.3 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。

実証の対象・期間・検証項目

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 35 名
群分け	統制群（17 名）：従来型のプログラミング実習授業 実験群（18 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（ベータ版）＋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統計的有意に向上するか否か。 ・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

表 1. 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	27.2	60.6	33.3
統制群	25.7	38.6	12.9

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。事前テストと事後テストはそれぞれ 10 問で構成され、解答に必要な知識や難易度がなるべく同程度になるよう類似の問題を扱った。出題内容は、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。事前・事後テストの難易度は後述する宿題と同程度とした。

表 1 に、事前・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。事前・事後テストいずれも、1 問 10 点の 100 点満点としている。一部の学生は事後テストを受験していないため、事前・事後テストの平均点の算出対象からは除外した。「事前・事後テストの差分」の列は、事前テストと事後テストの平均値の差分（四捨五入）である。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ほぼ同一となっている。一方、全 4 回の講座を実施した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おり、事前・事後テストの差分は、実験群では 33.3 点であったのに対し、統制群では 12.9 点であった。両者の差は 20.4 点であり、100 点中の割合としては 20.4%の差である。また、統制群の点数差分に対する実験群の点数差分の比は約 258%である。

表 2. 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	20	100	80
2	実験群	20	90	70
3	実験群	10	80	70
4	実験群	30	90	60
5	実験群	0	60	60
6	実験群	50	90	40
7	実験群	40	80	40
8	実験群	40	80	40
9	実験群	60	90	30

10	実験群	20	50	30
11	実験群	20	50	30
12	実験群	30	50	20
13	実験群	30	50	20
14	実験群	0	20	20
15	実験群	0	10	10
16	実験群	30	30	0
17	実験群	0	0	0
18	実験群	90	70	-20
19	統制群	10	70	60
20	統制群	50	100	50
21	統制群	20	60	40
22	統制群	0	40	40
23	統制群	40	60	20
24	統制群	0	10	10
25	統制群	100	100	0
26	統制群	100	100	0
27	統制群	0	0	0
28	統制群	0	0	0
29	統制群	10	0	-10
30	統制群	10	0	-10
31	統制群	10	0	-10
32	統制群	10	0	-10
33	統制群	100	N/A	N/A
34	統制群	20	N/A	N/A
35	統制群	0	N/A	N/A

表 2 に、事前・事後テストの学生ごとの点数を示す。各行は各学生のテスト結果を示しており、事前・事後テストの差分が大きい順に行を並び替えている。テストを受験していない場合は「N/A」と記載している。

図 1 は、各群における事前・事後テストの点数差分のヒストグラムである。実験群は 30 点前後を中心とした正規分布に近い分布になっているのに対し、統制群は 0 点前後を峰として全体的に左に偏った分布となっている。

事前・事後テストの点数の差分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、各学生の事前・事後テストの差分を観測値として、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 $p < .05$（U=181、実験群の標本サイズ=18、統制群の標本サイズ=14、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、実験群と統制群の間で、事前テストの段階では点数に統計的に有意な差が無いことを確認す

るため、事前・事後テストの両方を受験した学生の事前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 = .39（U=149、実験群の標本サイズ=18、統制群の標本サイズ=14、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。

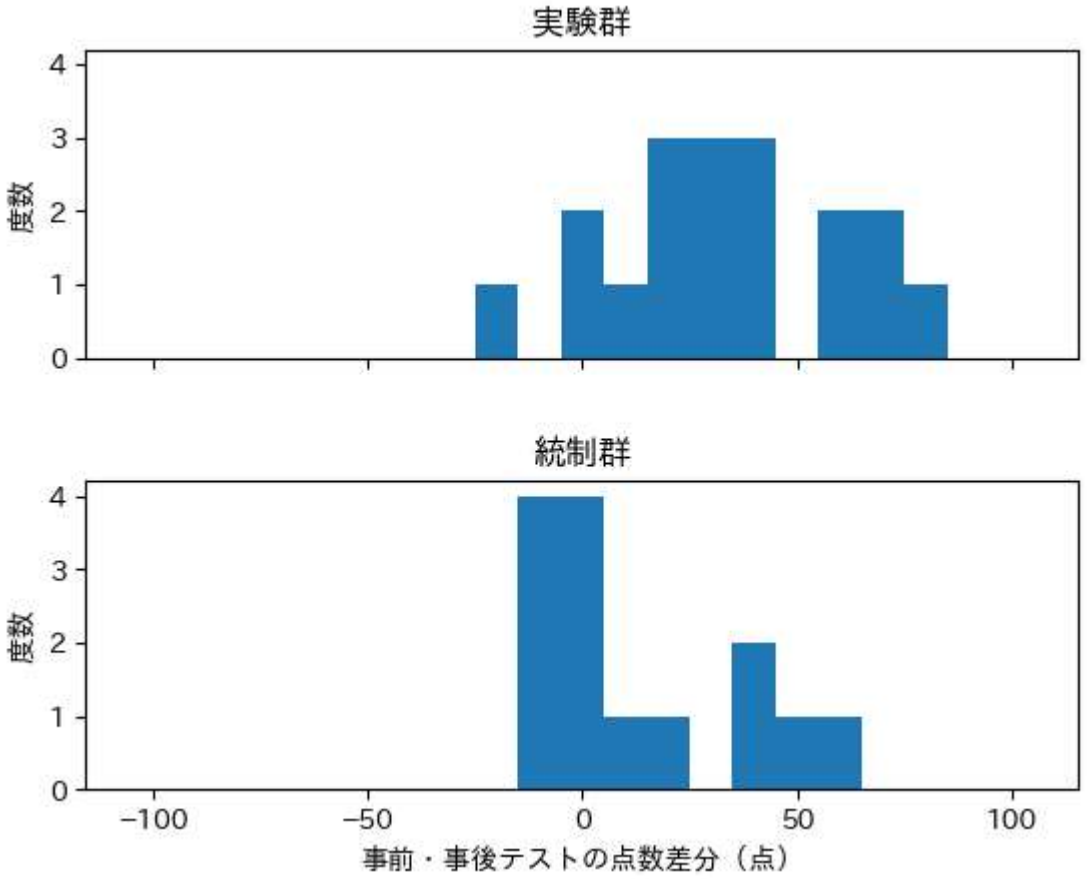


図 1. 各群における事前・事後テストの点数差分のヒストグラム

２．宿題の提出状況

表 3. 宿題の提出数・正解数の平均値

	提出数 （問）	正解数 （問）
実験群	35.3	35.3
統制群	16.6	14.1

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、学生自身の意思で解く問題や分量を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 問を出題し、全部で 47 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し、統制群は Google フォームで提出することとした。

表 3 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 18.7 問、21.1 問である。

図 2 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 47 問を峰とした右に偏った分布になっているのに対し、統制群は提出数・正解数いずれも 0 問を峰とした左に偏った分布になっている。

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=254、実験群の標本サイズ=18、統制群の標本サイズ=17、両側検定）であり、正解数は $p < .001$（U=244、実験群の標本サイズ=18、統制群の標本サイズ=17、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。

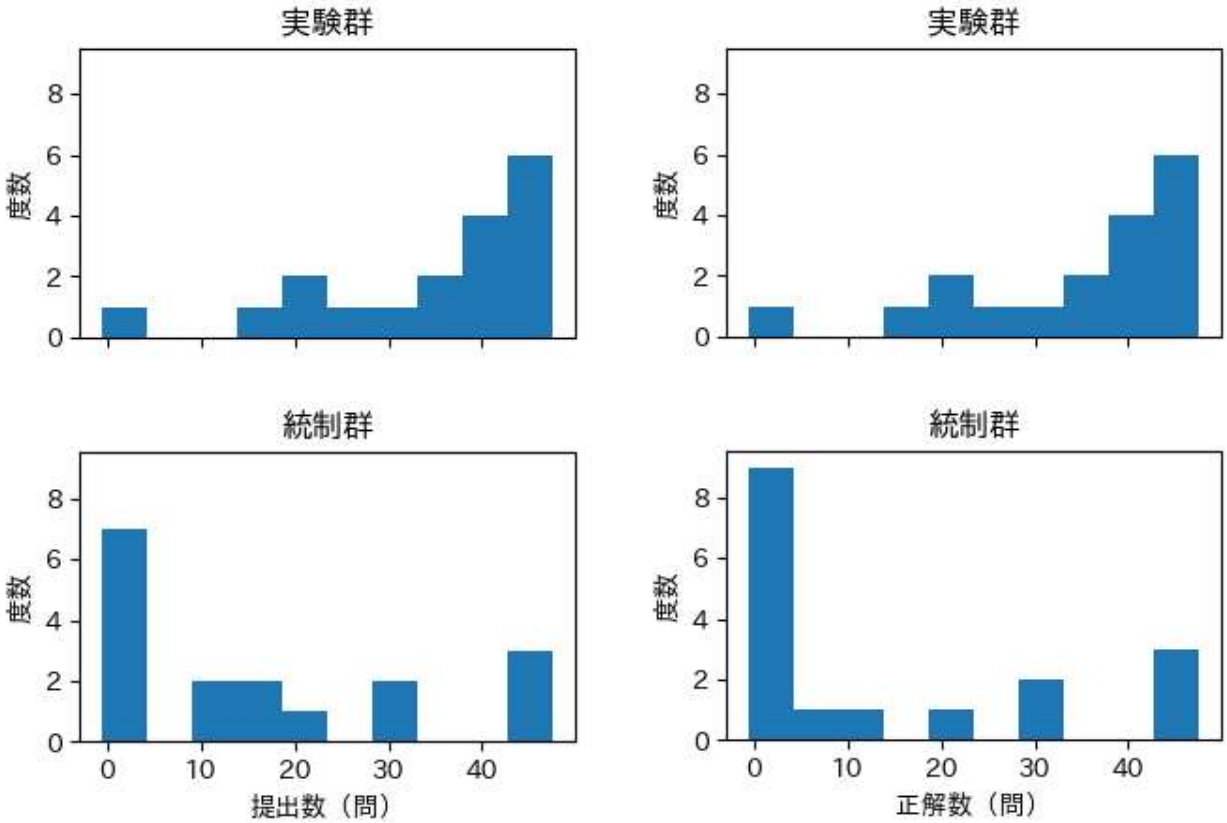


図 2. 各群における宿題の提出数（左）および正解数（右）のヒストグラム

３．事後アンケートの結果

表 4. 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。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
1	講義の理解度（第 1 回）	実験	65%	35%		0%	0%
		統制	15%	62%		23%	0%
2	講義の理解度（第 2 回）	実験	47%	47%		6%	0%
		統制	23%	46%		31%	0%
3	講義の理解度（第 3 回）	実験	29%	65%		6%	0%
		統制	8%	46%		46%	0%
4	講義の理解度（第 4 回）	実験	47%	41%		12%	0%
		統制	15%	31%		54%	0%
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	82%	18%		0%	0%
		統制	46%	46%		8%	0%
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	88%	12%		0%	0%
		統制	54%	31%		15%	0%
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	82%	12%		6%	0%
		統制	54%	38%		8%	0%
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	82%	18%		0%	0%
		統制	54%	38%		8%	0%
9	講師の説明の分かりやすさ	実験	65%	29%		6%	0%
		統制	46%	31%		23%	0%
10	講座に意欲的に取り組めたか	実験	76%	24%		0%	0%
		統制	46%	31%		23%	0%
11	自主的な学習時間の変化	実験	24%	59%	18%	0%	0%
		統制	0%	54%	46%	0%	0%
12	Python プログラミングへの興味が増したと感じたか	実験	71%	29%		0%	0%
		統制	23%	46%		31%	0%
13	システムの使いやすさ	実験	65%	29%		6%	0%
		統制	31%	31%		38%	0%
14	システムは学習の役に立ったか	実験	71%	29%		0%	0%
		統制	23%	54%		23%	0%
15	システムを本講座以外でも使って学習したいか	実験	47%	53%		0%	0%
		統制	8%	62%		31%	0%

表 4 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」の回答割合が多い。また、半数以上の質問において、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であり、それ以外の質問では同程度の評価である。

実験群と統制群の回答の間に統計的に有意な差があるか検証するため、各学生の回答を数値に置き換えて観測値としたうえで、各質問項目に対して Mann-Whitney の U 検定を行った。回答を数値に置き換える際は、4 件法の質問については 1（とても良い）から 4（とても悪い）とし、5 件法の質問については 1（とても良い）から 5（とても悪い）とした。その結果、3、4、5、6、11、13 番目の質問において $p < .05$ であり、1、12、14、15 番目の質問において $p < .01$ であったため、全体の 2/3 の質問において、実験群と統制群の回答の間に統計的に有意な差が認められた（実験群の標本サイズ=17、統制群の標本サイズ=13、両側検定）。

表 5. 実証講座の事後アンケート（適切さに関する質問）の結果。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	65%	24%	12%	0%	0%
		統制	23%	15%	15%	23%	23%
2	講義の難易度（第 2 回）	実験	41%	24%	35%	0%	0%
		統制	15%	15%	15%	23%	31%
3	講義の難易度（第 3 回）	実験	18%	18%	47%	18%	0%
		統制	8%	15%	23%	23%	31%
4	講義の難易度（第 4 回）	実験	24%	12%	47%	12%	6%
		統制	15%	0%	23%	23%	38%
5	宿題の難易度（第 1 回）	実験	53%	18%	29%	0%	0%
		統制	23%	8%	23%	8%	38%
6	宿題の難易度（第 2 回）	実験	41%	18%	35%	6%	0%
		統制	15%	8%	23%	8%	46%
7	宿題の難易度（第 3 回）	実験	18%	6%	47%	29%	0%
		統制	8%	8%	23%	23%	38%
8	宿題の難易度（第 4 回）	実験	18%	6%	41%	29%	6%
		統制	15%	8%	15%	8%	54%
9	講義・宿題の問題の分量	実験	6%	0%	71%	24%	0%
		統制	0%	0%	46%	46%	8%

表 5 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「簡単」もしくは「適切」であるのに対し、統制群では1番目と9番目の質問を除く全ての質問で「難しい」となっている。1番目の質問では、実験群で最も多い回答のほうが統制群で最も多い回答より「簡単」に近い評価であり、9番目の質問では、実験群のほうが統制群より「適切」の割合が多い。

実験群と統制群の回答の間に統計的に有意な差があるか検証するため、表3の質問に対して行ったのと同様に、各学生の回答を 1（簡単）から 5（難しい）の数値に置き換えて観測値としたうえで、各質問項目に対して Mann-Whitney の U 検定を行った。その結果、4、5、7 番目の質問において $p < .05$ であり、1、2、6 番目の質問において $p < .01$ であったため、全体の 2/3 の質問において、実験群と統制群の回答の間に統計的に有意な差が認められた（実験群の標本サイズ=17、統制群の標本サイズ=13、両側検定）。

表 6. 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。各質問の「良い／悪い」の選択肢は、質問の上から順に、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) 使っていた／使っていなかった、4) 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
1	動機促進 AI の閲覧頻度	35%	47%	6%	12%
2	動機促進 AI による学習意欲の向上	18%	47%	35%	0%
3	誤り箇所推定 AI の使用頻度	12%	59%	12%	18%
4	誤り箇所推定 AI は学習の役に立ったか	53%	18%	24%	6%

表 6 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、1 番目の質問は 71%、2 番目の質問は 71%、3 番目の質問は 65%、4 番目の質問は 82%となり、いずれの質問でも 6〜8 割程度の回答が良い評価である。

事後アンケートでは、表 3 から 6 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、1 番目と 2 番目の回答中の「ヒント」は誤り箇所推定 AI 機能のことを指し、4 番目の回答中の「スコアボード」は動機促進 AI 機能の一部である。

- ヒントが特定のコードをうたないと見れないようになっているので、見当もつかないような問題に当たった時に解き方がわからないといったことが起こる
- ヒントをあらかじめ決めておいてヒントのレベル的なものを作ったらいいと思いました

- 宿題をやっているときに、自分がどの問題をやり終わったかが覚えていないといけなかったので、提出したことのある問題にはなにか目印があるとよかったです。
- 問題に正解した時の得点が 100 点で固定なのにスコアボードに 120 点とか 121 点とか 100 点刻みじゃない人がいると bot なのがまるわかりなので 点数じゃなくて正解した問題数を表示すればいいと思った

４．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（5 問 3 人分）0.6 時間、個別補習 2 時間（合計 3.6 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）

両群の差は講座 1 回あたり 3.6 時間であり、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、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。

５．考察

事前・事後テストの点数の差分を実験群と統制群の間で比較すると、100 点中の割合として 20%を超える統計的に有意な差が出ている。また、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。また、図 1 のヒストグラムから、実験群では事前テストの点数と比べて事後テストの点数が全体的に増加しているのに対し、統制群ではほとんど点数が増加していない学生が多数存在するほか、点数の増加量も実験群と比較して全体的に少ないことがわかる。このことから、統制群における従来型の教育では効果的に学習できない学生が多いが、実験群における AI 搭載オンラインプログラミング実習システムを使用した教育では効果的に学習でき、その成果が点数に表れやすいと考えられる。

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、P 値も低い値を示している。また、図 2 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。このこ

とから、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、統制群では Google フォームにて解答を提出した後、一定期間後に教員が手動で採点を行って初めてフィードバックが与えられるのに対し、実験群では実習システム上で問題の解答を行った直後に正誤のフィードバックが与えられる点も、実験群の学習意欲を向上させる一因となっていると考えられる。

事後アンケートの結果のうち、表 4 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。統計的に有意な差が認められた項目として、システムの評価に関する質問（13～15 番目）は自明であるが、講義の理解度および Python プログラミングの理解度向上への貢献に関する質問の半数以上（1、3～6 番目）や、自主学習時間や興味の向上に関する質問（11、12 番目）があり、学習の主観的な理解度合いや学習への積極性にも向上がみられる。

表 5 に示した講座の難易度および問題の量に関する質問では、難易度に関する全ての項目（1～8 番目）において実験群のほうが「簡単」に近い評価となっており、ほとんどの項目で統計的に有意な差が認められる。これは、AI 搭載オンラインプログラミング実習システムが実験群の学習理解度を向上させたことにより、主観的な難易度が和らいだためと考えられる。また、問題の分量に関する質問（9 番目）では、実験群は 71%が「適切」と回答したのに対し、統制群は 54%が「やや多い」または「多い」と回答している。これは、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により効率的に学習できたのに加え、問題に意欲的に取り組めたため、問題の量が適切に感じられたためと考えられる。

表 6 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。一方で、「やや悪い」「とても悪い」に相当する回答をした学生も一定数存在しており、これらの学生に対しても AI 機能が有効に働くようにすることが今後の課題である。これには、AI 機能のチューニングを行うほか、AI 機能を有効に活用できるようにするためにシステムの UI/UX を改善していくことが考えられる。

事後アンケートの自由記述では、AI 機能およびシステムのユーザビリティに関する回答があった。これらの意見は今後の実習システムの開発において検討していくものであるが、この中でも特に「ヒント」（誤り箇所推定 AI 機能）の提示の仕方に関しては、教育効果への影響も考慮しつつ、AI 機能の補助を受けられる機会を増やすことが考えられる。

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 3.6 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、36～54 時間ほどの時間を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

◆赤松（正教）委員

・実証講座を見学して、今後ますます AI の品質が上がり効果が授業全般に現れてくるようになれば、授業のあり方や教員の立ち回り方が根本的に変わってくるのではないかと感じた。

◆本田委員

・本システムを利用することによりプログラムの文法的や内容的の間違いについて気づきやすく、またどのように修正すればよいかを受講者にわかりやすく教えていると感じた。
・ヒント機能にて受講者の能力に応じて出力する内容を変えなければ、受講者が考えることを放棄する可能性があり、どのようなヒントを提示すればよいか難しい課題だと感じた。
・教員は受講者が提出したプログラムの内容を確認することができるが、受講者の学習進捗を把握することができず、つまずいている学生を特定することが難しいので、各受講生の進捗を教員に提示する機能が必要である。

◆井坂委員

・本システムはオンライン授業用に作られたシステムであるが、従来の対面授業環境下で併用して活用しても効果を発揮できるシステムになっていると感じた。
・従来の学習スタイルでは、内容を理解できなくなった学生はその時点でギブアップすることが多く、かといってひとりひとりを教員が個別にフォローすることも難しい状況であったが、本システムではヒント等のサポートがあるため、教員はほとんど手を貸さなくても学生は諦めることなく学習できる仕組みになっていた。
・従来の対面授業では、各学生の学習進捗が分かりづらいため、全学生に対して同じペースで授業を進めていることがほとんどで、学力格差が大きい専門学校では優秀な学生の満足度を下げる要因になっていた。しかし、本システムを活用した授業では、学生自身のペースで授業を受講することができ、現代にマッチした最適な授業スタイルであり、学力格差があっても全学生に対して満足度を与えることができるシステムだと感じた。
・今後は、教員管理側の機能を充実させ、システムではフォローの出来なかった学生を即時教員がフォローできるような仕組みが必要だと思われる。

◆伊藤委員

・従来の一斉形式の授業では学生ごとの進捗速度の格差が問題になったが、このシステムを利用した場合は、個別の進捗に応じた即時フィードバックがえられるため、一斉形式授業で生じがちな優秀学生の待ち時間によるモチベーション低下を防ぐことができ、学生自身のペースで学習できるため受講しやすい授業になっていると感じた。
・また、即時フィードバックに加えて学生同士が相談し合うことで、周囲との協調学習により試行錯誤が進みやすく、納得しながら正解を模索している様子がうかがえた。このような協調学習での連合記憶により学習内容が確実に定着することが期待できるように思われる。

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、25～28 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. ヒント機能はどの程度使っていましたか。
28. ヒント機能は学習を進める上で役に立ちましたか。
29. 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。
30. 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

異なる整数 a と b のうち、小さ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 3 番目の要素の積を出力せよ。 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、5 の倍数で、かつ 10 の倍数ではない整数を出力せよ。 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、100 で割った余りが 61 とな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

正の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、偶数なら e、奇数なら o と出力せよ。 e または o 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。 ただし、与えられる整数はすべて 100 以下である。 与えられた整数のうち**最小値**を出力せよ。

```
li = [int(x) for x in input().split()]
ans = 100
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**先頭**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 4 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 4 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 を先頭に移動すると li は [8, 2, 5, 4, 6] となる。

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

次に、添え字 4 の要素 (li[4]) の 6 を先頭に移動すると li は [6, 8, 2, 5, 4] となる。

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

したがって、最終的なリストの状態は [6, 8, 2, 5, 4] 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 4 1 3 を読み込んだ場合は、pos は [2 4 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。得点は正の整数である。 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。
各ユーザについて、そのユーザの得点の最大値を出力せよ。

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
4
Carol
8
```

上記の例に対する出力は以下のようになる。

```
('Alice', 5)
('Bob', 4)
('Carol', 8)
```

```
d = {}
for i in range(5):
    name = input()
    score = 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 Person を定義せよ。

```
Name: Taro
Gender: Male
Name: Hanako
Gender: Female
```

```
# ここに解答を入力

taro = Person("Taro", "Male")
print("Name: " + taro.get_name())
```

```
print("Gender: " + taro.get_gender())

hanako = Person("Hanako", "Female")
print("Name: " + hanako.get_name())
print("Gender: " + hanako.get_gender())
```

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 RichDecorator を定義せよ。

```
$Python$
$$$Python$$$
```

ただし、 RichDecorator クラスは以下の制約を満たすこと。

・Decorator クラスを継承する
・decorate メソッドをオーバーライドする
・output メソッドはオーバーライドしない

```
class Decorator:
    def output(self, text):
        print(self.decorate(text))

    def decorate(self, text):
        return "$" + text + "$"

# ここに解答を入力

decorator = Decorator()
decorator.output("Python")

richDecorator = RichDecorator()
richDecorator.output("Python")
```

2.2.6. 運営・参加マニュアル

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

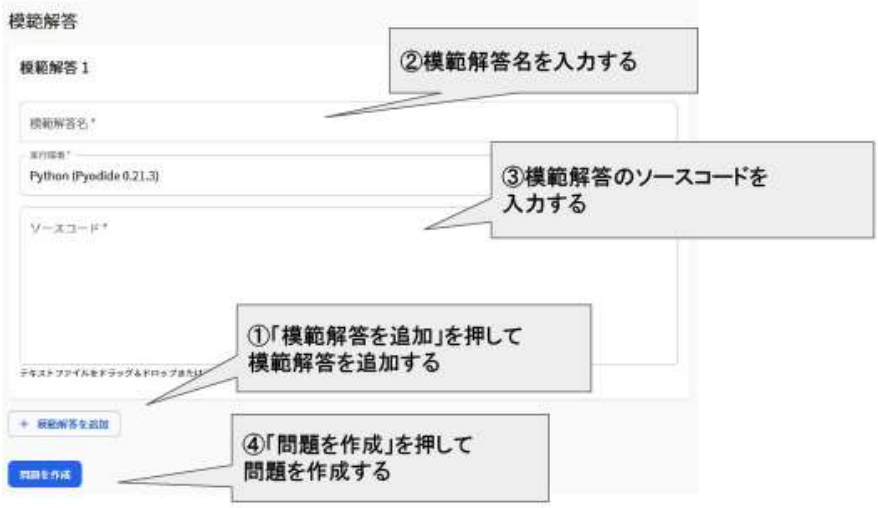
- 学習者の進捗を確認する
- 答案のステータス・ソースコードを確認する

問題作成（1/5）

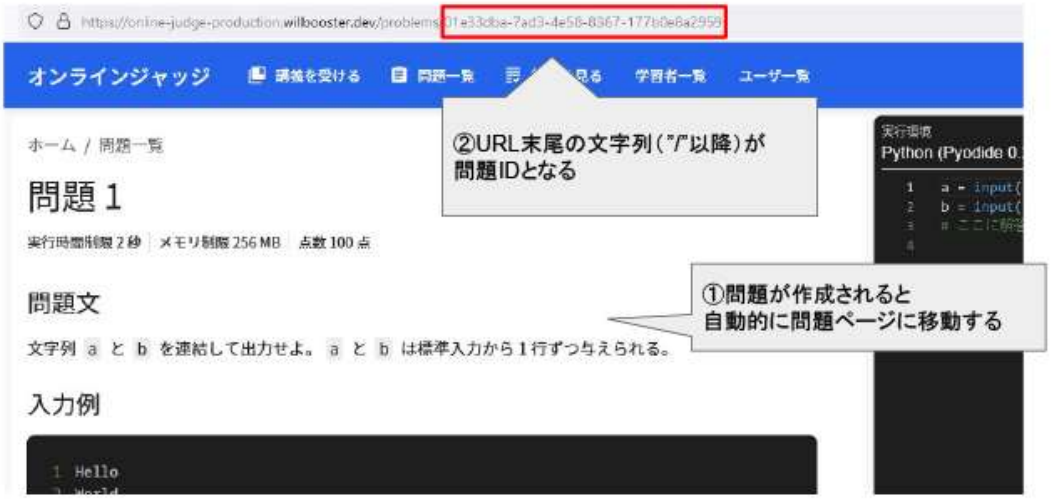
問題作成(2/5)

問題作成(3/5)

問題作成(4/5)



問題作成(5/5)



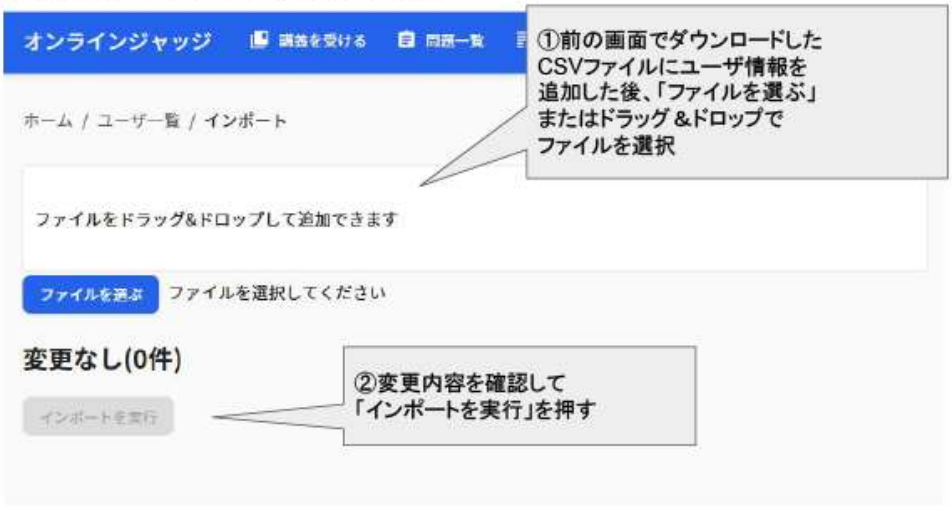
科目作成(1/3)

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)

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

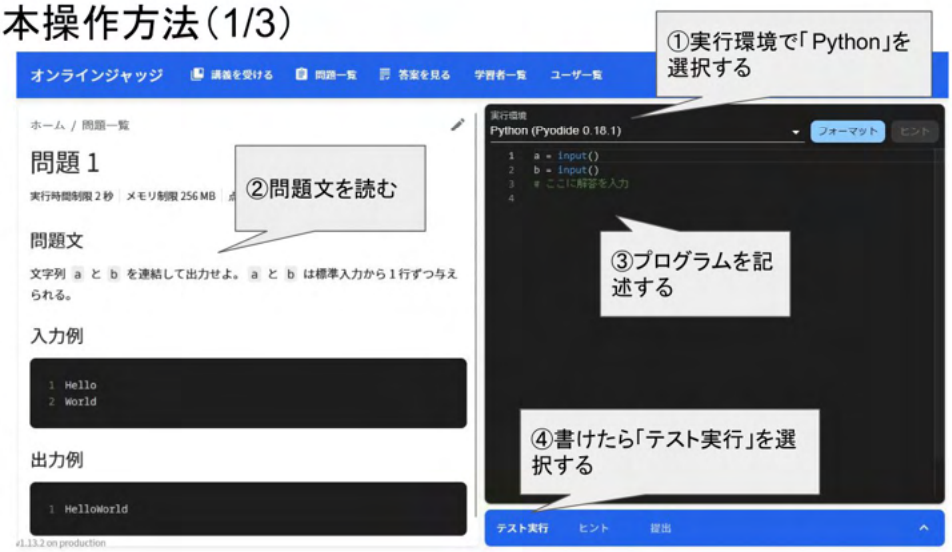
```
HelloWorld
```

正解プログラム

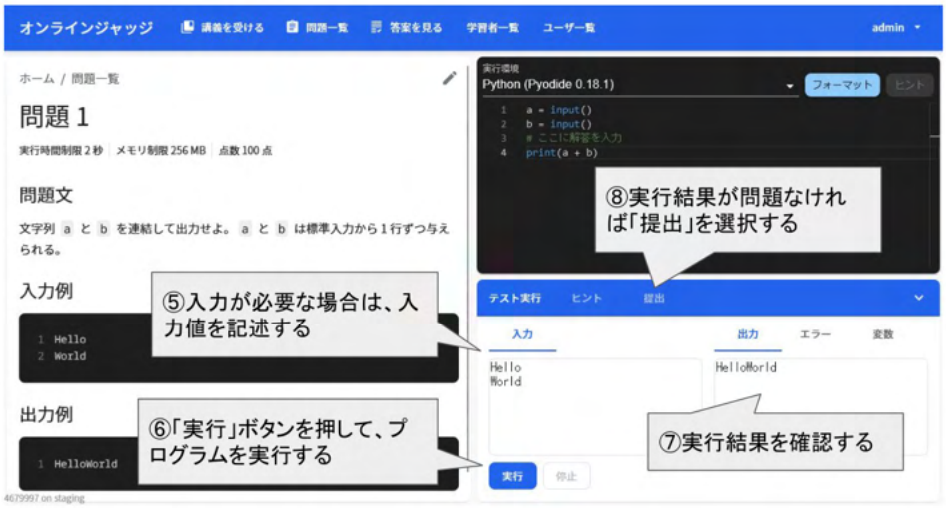
```
a = input()
b = input()
print(a + b)
```

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

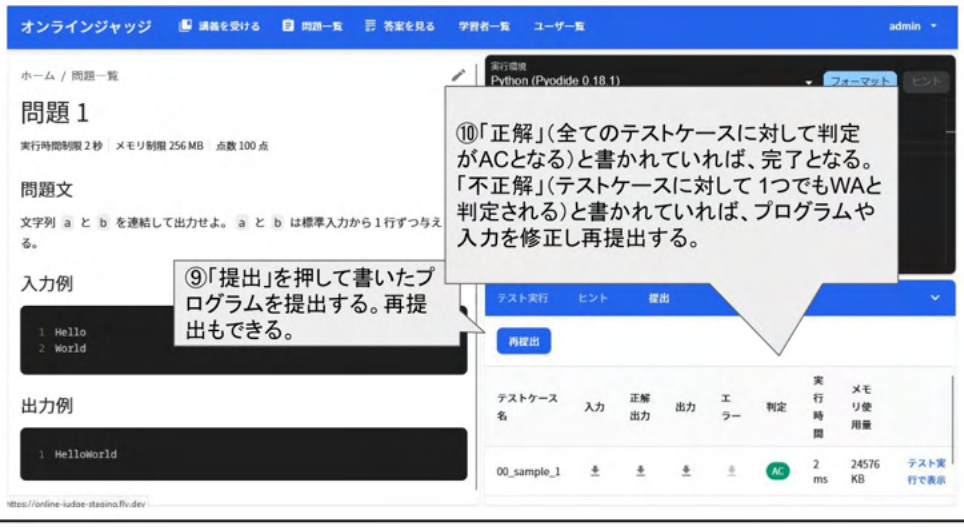
基本操作方法(1/3)



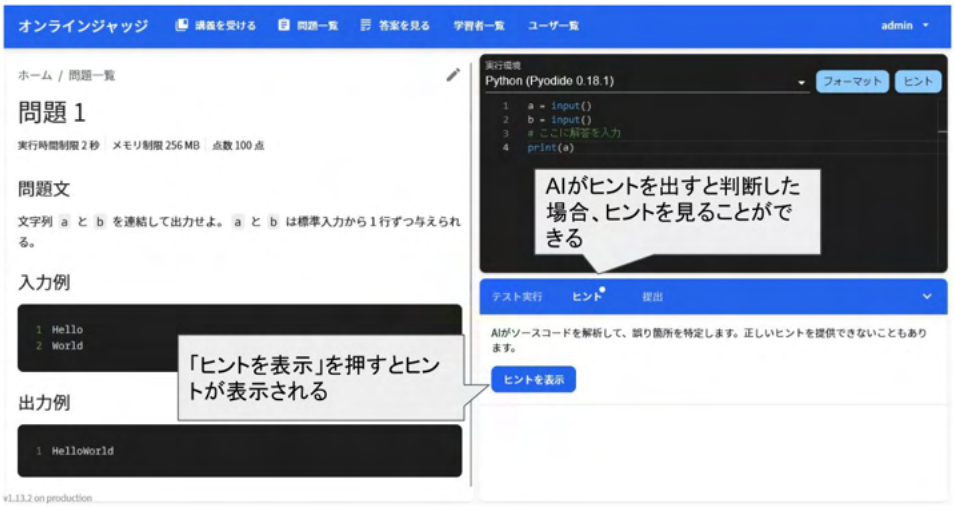
基本操作方法(2/3)



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】

図 1 で画面のスクリーンショットを示す。本画面は学生が問題を解く画面であり、画面左下の「類似コード」欄に AI が推薦した類似コードの URL が表示される。文書同士の類似度を計算する手法を用いることで、問題の模範解答と OSS から収集したソースコード間の類似度を計算する。この結果得られた類似度の高いソースコードのうち、適切なソースコードを教員があらかじめ選択して問題に設定しておくことで、類似コードとして表示される。図 1 の例では、CSV ファイルの読み込みを行う問題の模範解答と類似したソースコードの URL が表示されている。提示された URL が指すソースコードを図 2 で示す。問題の解答に必要となる CSV ファイルの読み込みに関する類似コードが推薦されていることが確認できる。

開発初期の段階で、既存のオンラインジャッジ上の類題を推薦する 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 では OSS 等から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。OSS からデータ収集を行って推薦対象とした結果、適切に推薦できる問題の種類が増加した。

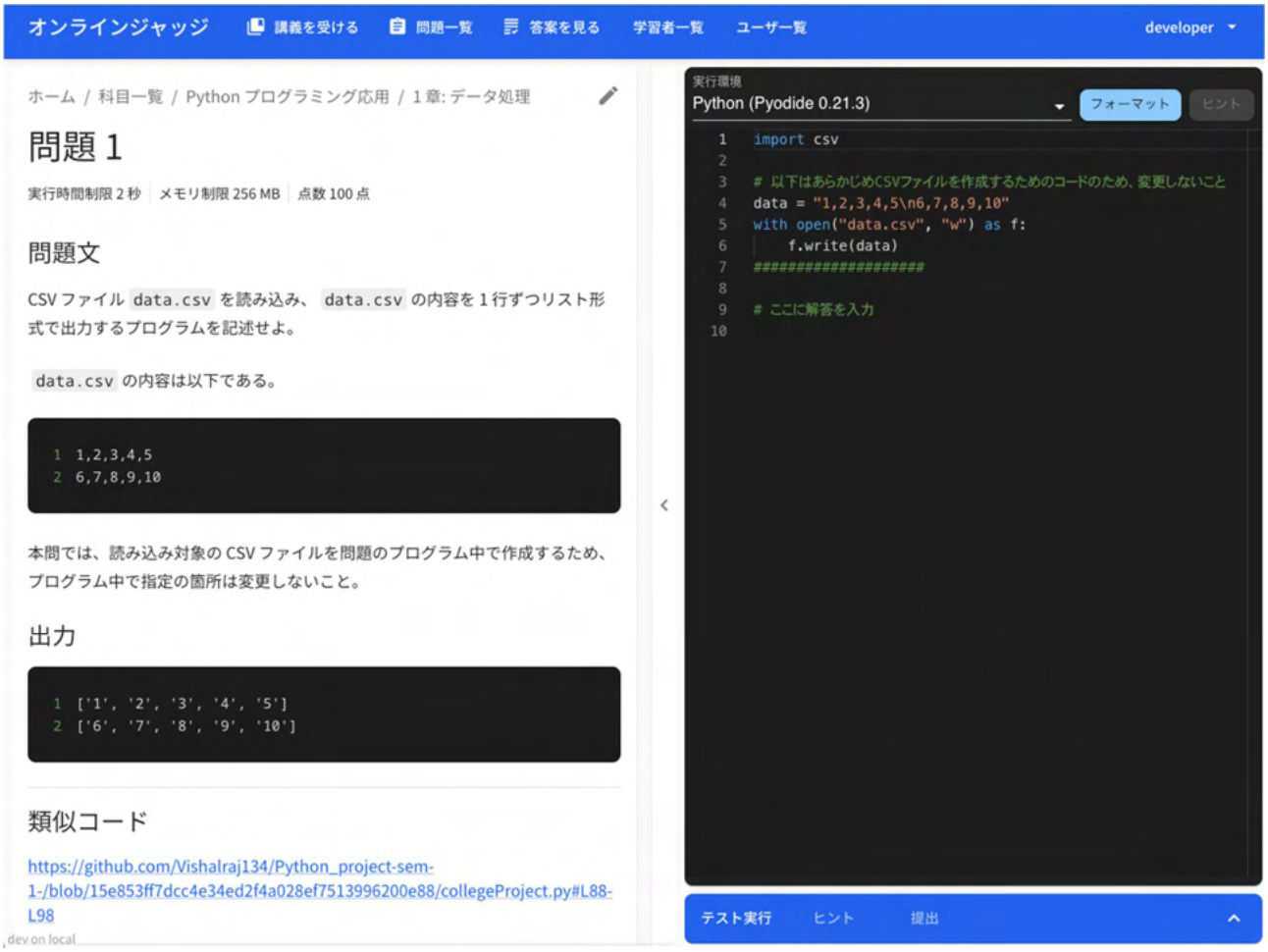


図 1. 課題に応じて模範となるコードを提示する AI（画面左下）

```
88  def get_batch():
89      with open(f'C:/PythonProgrammingProject_main-folder/Batch.csv','r') as f:
90          reader=csv.reader(f)
91          rows=[row for row in reader]
92          column=[]
93          for i in range(len(rows)):
94              if i==0:
95                  pass
96              else:
97                  column+=[rows[i][0]]
98      return column
```

図 2. 課題に応じて模範となるコードを提示する AI により提示されたソースコード

【不正解の原因・誤り箇所を推定する AI】

図 3 で画面のスクリーンショットを示す。本画面は学習者が問題を解く画面であり、画面右下の「ヒント」欄にて AI の動作結果が表示される。学習者が画面右上のコードエディタで誤ったプログラ

ムを記述したとき、正解プログラムとの差分をもとに AI が誤り箇所を特定して、画面右下のヒント欄に誤り箇所と修正方法を提示する。なお、誤り箇所や修正方法の情報は、誤り箇所のみ、誤り箇所と修正方法といったように、学習者が自ら選択して段階的に閲覧できるようになっている。

本 AI では、正解プログラムと不正解プログラムの差分をもとに誤り箇所を特定するが、本質的には同一であるとみなせる差分は吸収できるようにしている。AI の開発初期段階では、正解プログラムと不正解プログラムの差異が非常に小さい場合のみ誤り箇所が推定できる状態であった。そこで、実証講座における学生の投稿プログラムや講師の模範解答などのデータをもとに、差異を吸収すべきケースを特定したうえで、差異を吸収する仕組みを増やすことで、正解プログラムと不正解プログラムの差異がある程度あっても誤り箇所を特定できるようにした。

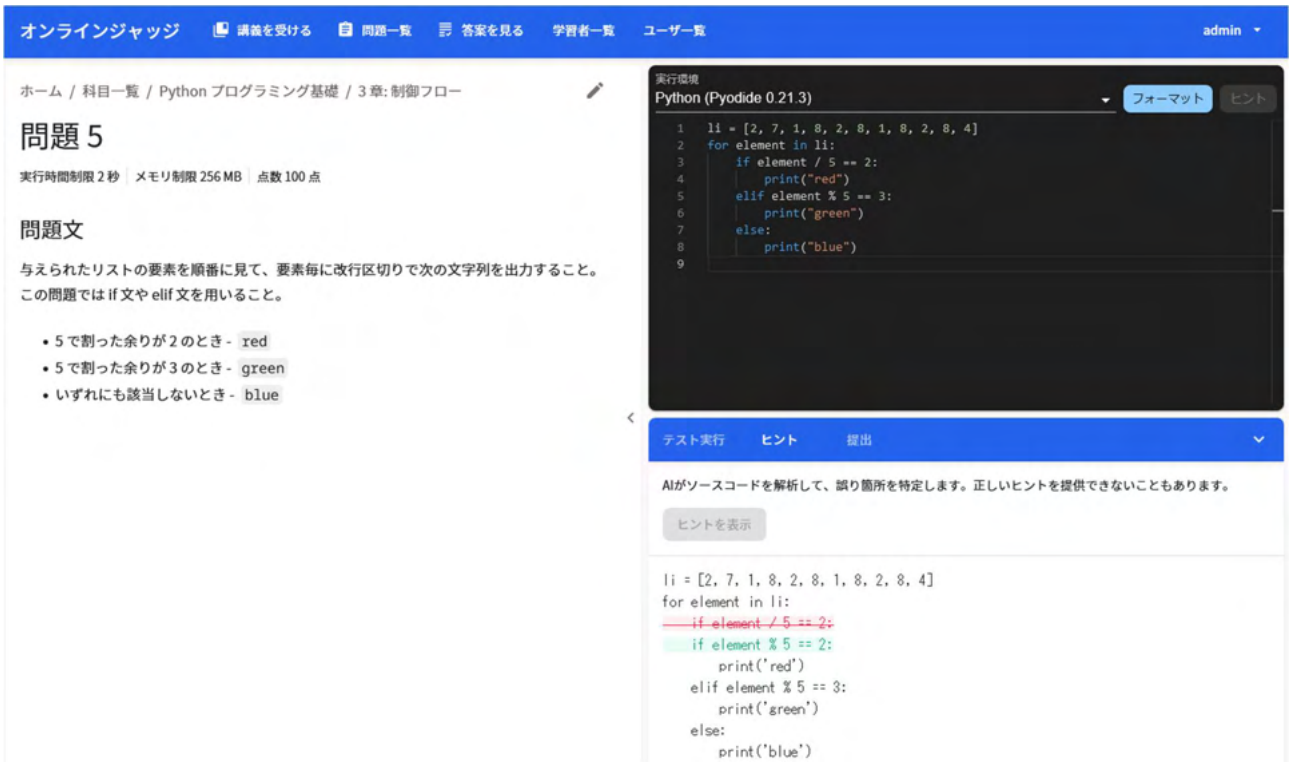


図 3. 不正解の原因・誤り箇所を推定する AI（画面右下）

【学生の個性に応じて動機づける AI】

図 4 で画面のスクリーンショットを示す。本画面はホーム画面であり、画面右上の欄に AI により提示された動機づけのためのフィードバックが表示される。あらかじめ学生に質問したアンケートの回答をもとに AI が学生の個性を分析し、それまでに収集した各学生の個性情報および学習履歴を用いることで、より学習意欲が増すフィードバックを推測して提示する。図 4 では、ボットを混ぜてランキングを活性化させることで拮抗した競争を行える得点ランキングをフィードバックとして提示している。

実証講座前の段階では、AI がフィードバックの推薦に使用するためのデータが蓄積されていなかったため、学生の個性に応じたフィードバックの出し分けができなかった。実証講座にて実際の学生のアンケート結果および学習履歴データを蓄積することで、学生の個性に応じたフィードバックを提示

できるようになった。また、フィードバックとして、AI 作成当初は「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」の 3 種類を作成していたが、後から「対人ランキング」を追加することでより幅広い個性に応じて最適なフィードバックを提示できるようにした。

図 4. 学生の個性に応じて動機づける AI（画面右上）

【採点を完全に自動化する AI】

プログラムの自動採点に必要なテストケースを自動生成するために、教員が用意した模範解答を真の仕様と見立てて、模範解答からテストケースを自動生成する AI を開発した。開発初期に試作した自動生成 AI では、活用したサードパーティ製のツールの都合上、模範解答を Java や C/C++で記述する必要があった。本事業では Python によるプログラミング教育を主な対象としているため、模範解答を Java や C/C++で記述することは、採点にかかる工数を増大させる要因となってしまう。そこで、利用可能なツールを調査したところ、Python で模範解答を記述可能なテストケース生成アルゴリズムを実装したツールを発見したため、当該ツールを活用して本事業に適したテストケースの自動生成 AI を開発した。

採点を完全に自動化する AI によるテストケースの生成例を示す。本事業で開発した基礎コマの 5 コマ目の教材「5 コレクション」の問題 1 を対象とする。図 5 が問題 1 の模範解答の Python プログラムである。AI が処理できるように、solve と命名したテストケースの生成対象となる関数を作成する必要がある。solve 関数の引数がテストケースの入力に該当して、戻り値が出力に該当する。solve 関数の仮引数に対して、各引数の取り得る値の範囲を確認するプログラムを書くことで、AI がより短時間で質の高いテストケースを生成できるようになる。図 6 で、AI が図 10 の模範解答から生成した JSONL 形式のテストケースの一部を抜粋する。各テストケースの入力は 5 つの値から構成されており、最初の 4 つの値は整数値で最後の値は文字列である。整数値に着目すると、全ての値が等しいパターンから、

全ての値が異なるパターンまで、まんべんなくテストケースが生成されていることがわかる。同様に、最後の値の文字列も多様な値が入力として採用されていることが分かる。自動採点による誤判定を防ぐためには、多様な入力を用意することが重要であり、AI は十分に多様な入力の生成に成功していることが読み取れる。

```python
def solve(A: int, B: int, C: int, D: int, s: str) -> str:
    if A <= 0:
        raise Exception()
    if B <= 0:
        raise Exception()
    if C <= 0:
        raise Exception()
    if D <= 0:
        raise Exception()
    if A > B:
        raise Exception()
    if C > D:
        raise Exception()
    s1 = s[A - 1 : B]
    s2 = s[C - 1 : D]
    return s1 + s2


if __name__ == "__main__":
    s = str(input())
    A, B, C, D = (int(x) for x in input().split())
    print(solve(A, B, C, D, s))
```

図 5. 採点を完全に自動化する AI における模範解答プログラムの例


```
{"inputs": [1, 1, 1, 1, ",-="], "output": ",,"}
{"inputs": [1, 6, 1, 1, "_=tG)"], "output": "_=tG)_"}
{"inputs": [1, 5, 1, 4, "N!IWL<<.TbBI&"], "output": "N!IWLN!IW"}
{"inputs": [3, 3, 1, 1, "__main__"], "output": "m_"}
{"inputs": [1, 1, 1, 32, "%`9&\f;Gy1@'"], "output": "%%`9&\f;Gy1@'"}
{"inputs": [1, 1, 1, 6, "__main__"], "output": "___main"}
{"inputs": [1, 5, 1, 14, ".d8'|2Wr}R~BR\n"], "output": ".d8'|.d8'|2Wr}R~BR\n"}
{"inputs": [8, 17, 1, 1, "HX<SdiJF~~Q pZ]x:V"], "output": "F~~Q pZ]x:H"}
{"inputs": [1, 3, 1, 1, "}\"F=q0="], "output": "}\"F}"}
{"inputs": [1, 8, 3, 21, "EQy;ma\rM{$6O"], "output": "EQy;ma\rMy;ma\rM{$6O"}
```


図 6. 採点を完全に自動化する AI における出力結果の例

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・ソースコードメトリクス・不正解の原因・誤り箇所を推定する AI の 3 つの観点に基づいて評価値を算出する予定である。その経緯および具体的な計算式について、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。コーディング規約ではソースコードメトリクスが活用されることがある。 旧来よりソフトウェア工学研究において、Cyclomatic Complexity や Halstead の複雑度などのソースコードメトリクスの測定値は欠陥数と相関があることが確認されている。近年では Cognitive Complexity など、ソースコードの理解容易性に関するメトリクスの有用性を確認する実証的な研究が進んでいる。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、その修正内容の分量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。

以上を踏まえて、本評価基準案として　減点制と部分点を採用する。まず、減点制を採用する理由として、コードチェッカーはコーディング規約等に違反する箇所を発見するツールであり、好ましいコード箇所を発見するツールではないことが挙げられる。また、ソースコードメトリクスは、測定値と欠陥数に相関性が確認されてはいるものの、測定値が良くなれば必ず品質が上がるといった強い相関があるわけではなく、測定値が非常に悪い場合に品質が下がり欠陥数が増えるといったことが研究において確認されていることが挙げられる。実務においては、コーディング規約等への違反がある、あるいはソースコードメトリクスの測定値が目立って悪い場合に、ソースコードを修正するという運用が一般的である。既に問題がないのにも関わらず測定値を少しでも向上させるといった不必要な活動を防ぐことも理由に挙げられる。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、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。減点制により、正解時の点数のバリエーションを増やすことができるが、不正解に対しては影響を及ぼさない。一方、部分点を導入することで、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。

減点制と部分点を実現するため、コーディング規約への違反がある、あるいはソースコードメトリクスの測定値が模範解答の測定値よりも悪い場合に、違反数や測定値の乖離の程度によって減点することとする。ただし、単純に減点された度合いだけを報告すると、例えば「あなたのソースコードの品質は-25 点です。」といったような負の値の報告となり、動機づけの観点から好ましくない。そこで、100 点から減点していく形式を取り、「あなたのソースコードの品質は 100 点満点中 75 点です。」といった報告をできるようにする。また、正解に近い不正解の場合は、最も近い正解の点数を計算して、その正解への近さを加味して減点した点数を部分点として付与することとする。

具体的な評価基準の評価値の計算式の案は以下の通りである。学習者が投稿したソースコード中のトップレベルのステートメント群もしくは関数の中で最大の Halstead の複雑度を Hl、教育者が用意した模範解答のソースコード中のトップレベルのステートメント群もしくは関数の中で最大の Halstead の複雑度を He、学習者が投稿したソースコード中のトップレベルのステートメント群もしくは関数の中で最大の Cognitive Complexity を Cl として、教育者が用意した模範解答のソースコード中のトップレベルのステートメント群もしくは関数の中で最大の Cognitive Complexity を Ce とする。また、部分点の割合は、正解の場合は 1、1 行の修正で正解できる場合は 0.75、2 行以上の修正で正解できる場合は 0.5 とする。

評価値 = (100 - min(10 * PEP8 への違反数, 50)
- min(max((Hl - He) / He, 0), 1) * 25
- min(max((Cl - Ce) / Ce, 0), 1) * 25) * 部分点の割合