令和３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

２０２２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3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. 遠隔環境下におけるプログラミング教育の現状と課題の調査 17
2.2.1. 調査概要 17
2.2.2. IT 系専門学校向け調査アンケートの結果 29
2.2.3. IT 系企業向け調査アンケート結果 68
2.2.4. 考察 78
2.2.5. まとめ 81
2.3. システムと課題付きテキストの開発 82
2.3.1. システム開発の成果 82
2.3.2. 課題付きテキスト開発の成果 102
2.3.3. プログラム評価基準案 104
3. 課題付きテキスト 105

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学生の理解状況を推定する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。

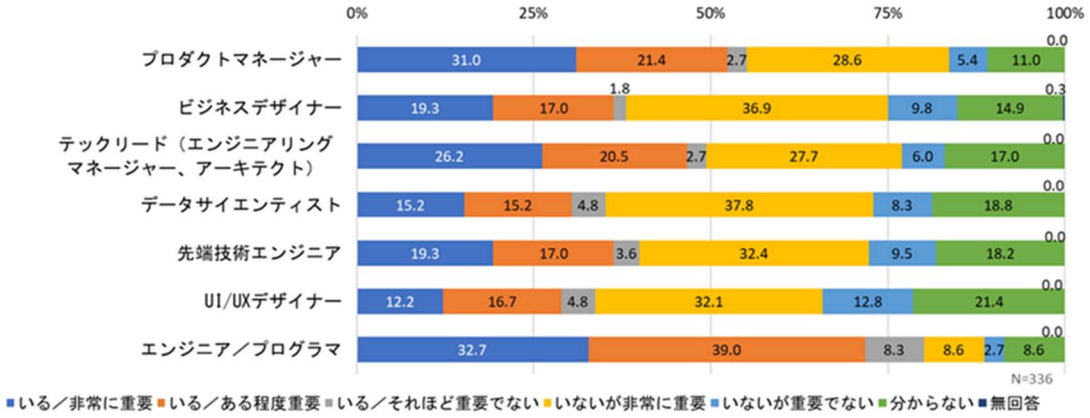


図 1. IT 企業における IT 人材種ごとの重要性の相違

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

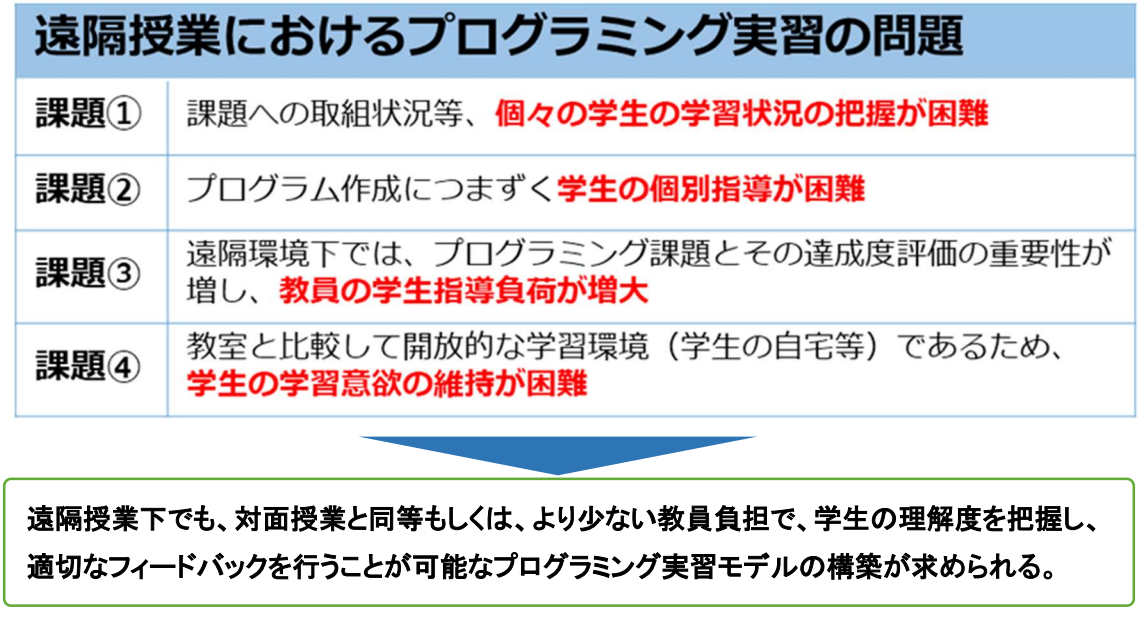
どちらの問題も、これまでは放課後や空き時間の指導（場合によっては個別指導）で補うというのが一般的であったが、翌日以降の授業準備も抱える教員にとっては大きな負担を抱えながらの苦肉の策である。しかし、新型コロナウィルス蔓延による遠隔授業への移行を受けて、実習時間中の指導も放課後フォローも同様に困難となったばかりか、以下のように問題が多面的に拡大してしまった。

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。

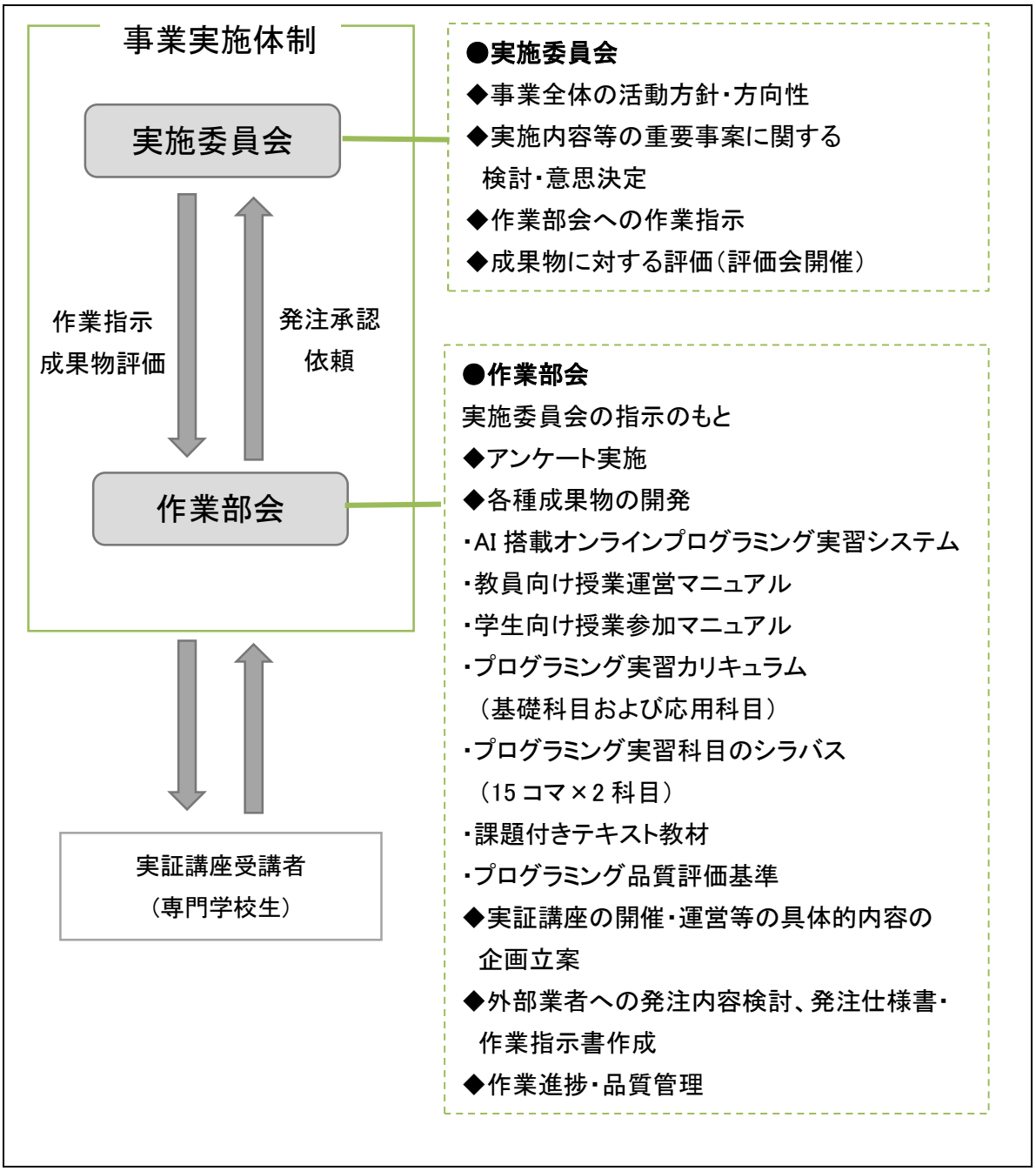
③①の問題により、頻回のテストによる学生の理解度確認の必要性が高まったが、プログラミング技術を問うテストは作成だけでなく採点にも手がかかり、教員負荷が大きい。放課後フォローやプログラム品質に関する指導がますます困難になる。

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。



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	愛媛県
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	愛媛県
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	高知県
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	東京都
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	千葉県
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	福岡県
10	松浦達也	株式会社いよぎんコンピュータサービス	開発・評価	東京都
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	香川県
12	國廣尚良	株式会社アイ・エヌ・エス	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　会長	評価	愛媛県
15	藤本朋成	愛媛県経済労働部産業支援局	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	加茂野怜	学校法人河原学園　システム部職員	開発・実証	愛媛県
3	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
4	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
5	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
6	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
7	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
8	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	2021 年 11 月 11 日（木）17：00～18：00
場　所	TKP 松山市駅前カンファレンスセンター 2F （愛媛県松山市千舟町 4 丁目 3−7 青野ビル)、 オンライン会議 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)、中村亮（学校法人河原学園　河原電子ビジネス専門学校)
議題	１．委員長挨拶 ２．委員　自己紹介 ３．事業計画の概要説明 ４．アンケート項目の方向性決定 ５．意見交換 ６．今後のスケジュール ７．事務連絡
配布資料	資料１　事業概要 資料２　IT 企業向けアンケート文案 資料３　専門学校向けアンケート文案

◆第 2 回実施委員会

日　時	2021 年 12 月 13 日（月）17：00～18：30
場　所	オンライン会議 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)、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)、中村亮（学校法人河原学園河原電子ビジネス専門学校)

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．事務連絡
配布資料	資料１　AI 搭載オンラインプログラミング実習システム開発概要 資料２　課題付きテキスト開発概要

◆第 3 回実施委員会

日　時	2022 年 02 月 14 日（月）17：00～18：30
場　所	オンライン会議 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）、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）、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）、中村亮（学校法人河原学園河原電子ビジネス専門学校）
議題	１．挨拶 ２．アンケート結果報告・開発報告 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　専門学校向けアンケート結果 資料２　企業向けアンケート結果 資料３　集計結果の補足資料 資料４　各開発の進捗報告 資料５　次年度以降反映予定機能一覧 資料６　課題付きテキスト（2 章「変数・データ型・代入・数値計算」）

1.3.5. 作業部会の開催

◆第 1 回作業部会

日　時	2021 年 11 月 19 日（金）16：00～17：00
場　所	TKP 松山市駅前カンファレンスセンター 4F （愛媛県松山市千舟町 4 丁目 3−7 青野ビル）、 オンライン会議 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）、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）、加茂野怜（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　事業概要 資料２　IT 企業向けアンケート文案 資料３　専門学校向けアンケート文案

◆第 2 回作業部会

日　時	2021 年 12 月 8 日（水）16：00～17：30
場　所	専門学校東京テクニカルカレッジ会議室 11F （東京都中野区東中野 4-2-3）、 オンライン会議 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）、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）、加茂野怜（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　AI 搭載オンラインプログラミング実習システム開発概要 資料２　課題付きテキスト開発概要

◆第 3 回作業部会

日　時	2022 年 01 月 27 日（水）17：00～18：30
場　所	オンライン会議 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）、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）、加茂野怜（学校法人河原学園）
議　題	１．挨拶 ２．アンケート結果報告・開発報告 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　専門学校向けアンケート結果 資料２　企業向けアンケート結果 資料３　集計結果の補足資料 資料４　各開発の進捗報告 資料５　課題付きテキスト（2 章「変数・データ型・代入・数値計算」）

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、生徒の理解度を推定する 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：学生の理解度を推定する AI

既存のオンラインジャッジでは、問題の一覧が与えられるだけであり、どの問題をどの順序で解くべきかといったガイドは与えられない。そこで、機械学習における推薦アルゴリズムを応用して、学生の過去の学習履歴から、解けそうな問題・解けなさそうな問題を推定する AI 技術を組み込む。解けそうな問題・解けなさそうな問題の一覧は、学生の理解度を表現した情報であり、学生は解けそうにない問題に挑戦することで、新しいプログラミングスキル・知識を効率よく習得できる。

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

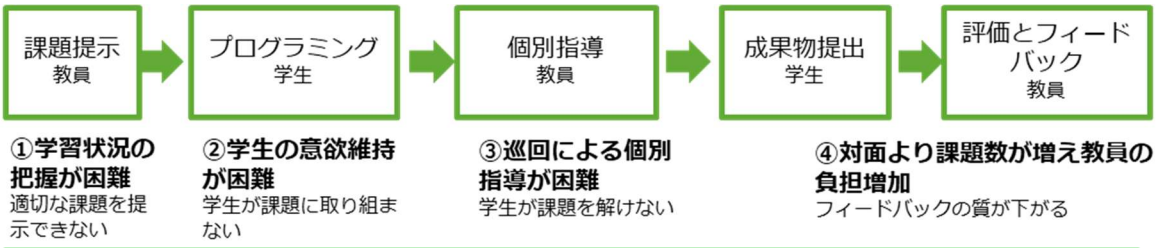
従来のオンラインジャッジは、プログラムを自動採点するために必要なテストケースを教員が用意する必要があり、テストケースの作成は容易ではないという問題があった。採点回数が十分に多ければ、採点コストがテストケースの作成コストを上回り、自動化によるコスト削減効果が出るものの、採点回数が少ないと、かえって採点の手間が増える問題がある。また、自動採点では正解・不正解の 2 値でしか評価できず、プログラムの品質を測定することはできない。そこで、ソフトウェア工学研究におけるテストケース生成技術を応用することで、与えられた模範解答からテストケースを自動生成する AI 技術を組み込み、さらに、ソフトウェアメトリク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



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 と Java の 2 言語に対応
・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学生の理解状況を推定するAI
理解状況に基づいて適切な課題を提示できる

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躓いても、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 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。

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について 1.2 プログラムについて 1.3 プログラミング言語の種類 1.4 Python について
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 コレクション	3.1 コレクションの種類 3.2 リスト 3.2 辞書 3.3 タプル 3.4 プログラム作成課題
4 条件分岐・繰り返し	4.1 条件分岐 4.2 繰り返し 4.3 繰り返しの制御 4.4 プログラム作成課題
5 関数・オブジェクト	5.1 関数 5.2 オブジェクト 5.3 プログラム作成課題
6 モジュール	6.1 モジュール 6.2 パッケージ 6.3 標準ライブラリと外部ライブラリ 6.4 プログラム作成課題
7 Python を使ったデータ分析および可視化	7.1 NumPy を使った分析 7.2 pandas を使った分析 7.3 Matplotlib を使った分析 7.4 seaborn による可視化 7.5 プログラム作成課題
8 Python を用いたシステム開発	8.1 システム開発の流れ 8.2 Python を用いたシステム開発 8.3 プログラム作成課題
9 ソフトウェアの品質保証	9.1 ソフトウェア品質保証の考え方 9.2 設計時においての品質保証 9.3 開発時においての品質保証 9.4 テスト時においての品質保証
10 総合演習	10.1 アルゴリズム演習 10.2 データ分析および可視化演習 10.3 システム開発演習

2.2. 遠隔環境下におけるプログラミング教育の現状と課題の調査

2.2.1. 調査概要

目的

AI 搭載オンラインプログラミング実習システムによって解決すべき、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明確化することを本調査の目的とする。

遠隔授業時のプログラミング教育の現状、使用しているプログラミング実習課題の種類・難易度、学生指導時に参照するプログラム品質の評価基準、技術修得の状況、その他プログラミング教育の課題について、専門学校を対象に調査する。

プログラミング人材の新卒採用者に要求するプログラミングスキルの内容とレベル、ソフトウェア製品に適用するプログラム品質の評価基準について、IT 企業を対象に調査する。

調査の対象・方法・実施期間および回収状況

プログラミング教育のカリキュラムを有する IT 系専門学校 59 校、ソフトウェアの受託開発事業を実施している IT 企業 301 社を調査対象とし、令和 3 年 12 月 17 日にアンケート用紙を郵便により発送し、令和 4 年 2 月 2 日まで回答期間で郵便による返信および同じ内容のウェブアンケートによる回答を受け付けた。

IT 系専門学校からは 43 件（回収率 72.9%）の回答を得ることができたが、IT 系企業からは 53 件（回収率 17.6%）のみの回答であった。

・IT 系専門学校向けアンケート

発送状況		回収状況：令和 4 年 2 月 2 日				
件数	発送日		合計	回収率	郵送	ウェブ
59 件	令和 3 年 12 月 17 日	全件	43 件	72.9%	0 件	43 件
		校数	35 校	59.3%	0 件	35 校

・IT 系企業向けアンケート

発送状況		回収状況：令和 4 年 2 月 2 日			
件数	発送日	合計	回収率	ウェブ	郵送
301 件	令和 3 年 12 月 17 日	53 件	17.6%	42 件	11 件

調査アンケート

専門学校向けアンケート用紙および IT 企業向けアンケート用紙を次ページ以降に掲載。

2021 年度 文部科学省委託事業
【プログラミング実習調査アンケート】回答用紙

※本アンケートは、次の URL より Web からもご回答いただけます。

URL: https://forms.gle/27basb5118CaaBYEA

（0）回答者基礎情報

貴校名			
貴学科名			
ご氏名		役職名	

※ご回答いただいた情報は、本文部科学省委託事業のアンケート集計のみに利用し、その結果を本事業にて作成するカリキュラムと教材に反映いたします。なお、個別のデータが貴社名およびご回答者名とリンクして公表されることは一切ございません。

以下の（1）～（6）に関して、各質問に回答してください。選択肢がある場合は、最も当てはまる選択肢の番号に○をつけてください。

（1）遠隔授業実施時のプログラミング実習の状況

質問 1　現行カリキュラムの修業年限、総授業時間数、プログラム言語の学習に費やす講義授業とプログラミング実習授業の時間数を、それぞれ回答してください。

【注意】 90 分＝2 時間として計算してください。

【注意】 1 つの科目のなかで講義授業と実習授業が混在する場合は、おおよその目安で分割して計算してください。

修業年限（　　）年　　総授業時間数（　　）時間

講義授業（　　）時間　　プログラミング実習授業（　　）時間

質問 1 で、プログラミング実習授業の時間数を 0 時間と回答した場合はアンケートを終了いただいて返信用封筒にて返送してください。
プログラミング実習授業の時間数を 0 時間以外で回答した場合は質問 2 以降にも回答してください。

質問 2　遠隔授業でプログラミング実習を実施したときに、どのような問題が生じましたか（複数回答可）。

1	学生が真摯にプログラミングに取り組んでいるかを把握しにくい
2	学生のつまずいている箇所が把握しにくい
3	プログラミングにつまずいている学生を特定しにくい
4	学生が理解しながらプログラミングを進めているかどうかを把握しにくい
5	プログラミングにつまずいている学生の個別フォローを行いにくい
6	とくに問題は発生していない
7	遠隔授業下ではプログラミング実習を実施していない

（2）プログラミング実習課題の難易度

質問 3　プログラミング実習において、おもに学習している言語はどれですか（複数回答可）。

1	Java
2	C++
3	C#
4	PHP
5	Ruby

6	Python
7	JavaScript
8	TypeScript
9	その他(　　)

質問 4　プログラミング実習において、おもに学習しているフレームワークはどれですか（複数回答可）。

1	Spring Framework(Java)
2	Struts2(Java)
3	TreeFrog Framework(C++)
4	ASP.NET(C#)
5	Laravel(PHP)
6	CakePHP(PHP)

7	Ruby on Rails(Ruby)
8	Django(Python)
9	React(JavaScript、TypeScript)
10	Vue(JavaScript、TypeScript)
11	その他(　　)
12	使用していない

質問 5　プログラミング実習において、おもに学習しているパブリッククラウドはどれですか（複数回答可）。

1	Google Cloud Platform
2	Amazon Web Services
3	Microsoft Azure
4	IBM Cloud

5	FUJITSU Hybrid IT Service FJcloud
6	その他(　　)
7	使用していない

質問 6　プログラミング実習において、最終的な課題のソースコードの量はどれくらいですか。

1	200 行未満
2	200 行以上 500 行未満

3	500 行以上 1000 行未満
4	1000 行以上

質問 7　プログラミング実習において、そこで作成するプログラムには、どのような要素が含まれますか（複数回答可）。

1	オブジェクト指向
2	データベース
3	著名なフレームワーク
4	アマゾン等が公開する Web API

質問 8　プログラミング実習において、プログラミング課題に取り組む学生の作業単位は何名ですか。

1	1 名
2	2～4 名
3	5～7 名
4	8 名～10 名
5	11 名以上

（3）プログラミング実習における指導のあり方

質問 9　指導担当の教員によって、指導の仕方に個人差はありますか。

1	個人差はない
2	個人差はある

質問 10　プログラミング教育において、プログラム品質に関する指導はどのように実施していますか（複数回答可）。

1	ソフトウェア工学を主題として扱う科目がカリキュラムに存在する
2	ソフトウェアの品質特性モデルに立脚したプログラム評価を指導している
3	プログラム品質としては、プログラム仕様への合致のみ指導している
4	ソフトウェアパターンの利用をプログラム品質と結びつけて指導している
5	その他（　　）

質問 11　プログラミング実習の指導（改善指導・改修指示）においては、どのような点を重視していますか。下記の項目ごとに、最も当てはまる選択肢に○をつけてください。

		【回答選択肢】 1　重視していない 2　あまり重視していない 3　やや重視している 4　重視している
プログラミングについて	プログラム実行時に正しい結果が得られているか	1　2　3　4
	バグになりそうなコードがないか	1　2　3　4
	コンパイルエラーにはならないが、不適切なアクセス修飾子を使用していないか	1　2　3　4
	多態性が適切に利用されているか	1　2　3　4
	抽象クラスや抽象メソッドが適切に利用されているか	1　2　3　4
	ソースコードの可読性、保守性が高いか	1　2　3　4
	拡張性が高いか(オブジェクト指向に沿ったプログラムか)	1　2　3　4
	テスト品質は適切か	1　2　3　4
	リソースを無駄に消費しないか	1　2　3　4
	記法や用語が一貫しているか	1　2　3　4
	アルゴリズムの使い分けは適切か	1　2　3　4
	アルゴリズムの内容を正確に記述できているか	1　2　3　4
プログラミング以外について	出席率が高いかどうか	1　2　3　4
	真摯な授業態度かどうか	1　2　3　4
	（作業単位が複数の場合）チーム内のコミュニケーションが良好か	1　2　3　4
	（作業単位が複数の場合）チーム内のマネジメントが良好か	1　2　3　4
その他（　　）		

（4）プログラミング実習における学生の成果物に対する評価のあり方

質問 12　プログラミング実習における学生の成果物に対して、教員による評価の状況を回答してください。

1	もともとプログラミング実習における成果物を評価する意図はない
2	評価できる教員がいないため、まったく評価できていない
3	評価できる教員はいるが、評価をする時間が足りないため、十分には評価できていない
4	十分に評価できている
5	その他（　　）

質問 12 で 1 または 2 と回答した場合は質問 15 に進んでください。
質問 12 で 3 または 4 と回答した場合は続けて質問 13 以降も回答してください。

質問 13　プログラミング実習における学生の成果物に対して、教員による評価の方法を回答してください（複数回答可）。

1	実習後に学生にプログラムを提出させ、教員環境でソースコードの中身を確認している
2	実習後に学生にプログラムを提出させ、教員環境で実行結果を確認している
3	実習中に学生開発環境でプログラムのソースコードの中身を目視で確認している
4	実習中に学生開発環境でプログラムを実行する画面と、実行結果を確認している
5	その他（　　）

質問 14　プログラミング実習における学生の成果物に対して、どのような点を重視して評価していますか。下記の項目ごとに、最も当てはまる選択肢に〇をつけてください。

【回答選択肢】
1　重視していない
2　あまり重視していない
3　やや重視している
4　重視している

プログラミングについて	プログラム実行時に正しい結果が得られているか	1　2　3　4
	バグになりそうなコードがないか	1　2　3　4
	コンパイルエラーにはならないが、不適切なアクセス修飾子を使用していないか	1　2　3　4
	多態性が適切に利用されているか	1　2　3　4
	抽象クラスや抽象メソッドが適切に利用されているか	1　2　3　4
	ソースコードの可読性、保守性が高いか	1　2　3　4
	拡張性が高いか(オブジェクト指向に沿ったプログラムか)	1　2　3　4
	テスト品質は適切か	1　2　3　4
	リソースを無駄に消費しないか	1　2　3　4
	記法や用語が一貫しているか	1　2　3　4
	アルゴリズムの使い分けは適切か	1　2　3　4
	アルゴリズムの内容を正確に記述できているか	1　2　3　4
プログラミング以外について	出席率が高いかどうか	1　2　3　4
	真摯な授業態度かどうか	1　2　3　4
	（作業単位が複数の場合）チーム内のコミュニケーションが良好か	1　2　3　4
	（作業単位が複数の場合）チーム内のマネジメントが良好か	1　2　3　4
その他（　　）		

質問 15　学生が作成したプログラムの評価を行う際に評価が難しいと考える点があれば記述してください。

（5）プログラミング関連資格取得状況

質問 16　プログラミングスキルの達成度をはかるために、学生にどのような資格を取得させていますか。ただし、受講学生の 50%以上が取得できている資格に限ります（複数回答可）。

1	基本情報技術者試験
2	Oracle Certified Java Programmer, Bronze SE 認定資格
3	Oracle Certified Java Programmer, Silver SE 11 認定資格
4	Android 技術者認定試験
5	Python3 エンジニア認定基礎試験
6	Python3 エンジニア認定データ分析試験
7	PHP 技術者認定試験　初級試験
8	PHP 技術者認定試験　準上級試験
9	その他（　　）
10	とくにない

（6）その他

質問 17　遠隔授業時のプログラミング実習に関する課題や問題があれば記述してください。

ご回答いただき、誠にありがとうございます。ご多忙の中、恐れ入りますが、令和 4 年 1 月 21 日（金）までに本回答用紙（全 7 ページ）を返信用封筒にてご返信いただきますようお願いいたします。

ご回答いただきました結果を、本事業のカリキュラムと教材の開発に活用させていただきます。

今後とも、何卒よろしくお願いいたします。

2021 年度 文部科学省委託事業
【人材ニーズ・品質ニーズ調査アンケート】回答用紙

※本アンケートは、次の URL より Web からもご回答いただけます。

URL：https://forms.gle/XWhzp6Zpv7dK84w27

（０）回答者基礎情報

貴社・貴団体名	
業種	
ご氏名	
部署名	
役職名	

※ご回答いただいた情報は、本文部科学省委託事業のアンケート集計のみに利用し、その結果を本事業にて作成するカリキュラムと教材に反映いたします。なお、個別のデータが貴社名およびご回答者名とリンクして公表されることは一切ございません。

以下(1)～(7)で、下記の項目ごとに、最も当てはまる選択肢に〇をつけてください。

（１） 新卒採用に関して、「文系大学生／理工系大学生／IT 系専門学校生」に対する選考基準に、プログラミングスキルを含めていますか。

学生区分	【回答選択肢】 選考基準に 1 含めていない 2 ほとんど含めていない 3 多少含めている 4 含めている
文系大学生	1 2 3 4
理工系大学生	1 2 3 4
IT 系専門学校生※	1 2 3 4

（※質問（１）の「IT 系専門学校生」で、1 以外を回答した場合は質問（２）にも、お答えください。）

（２） IT 系専門学校のプログラミング教育に、どのようなスキルの習得を期待していますか。

番号	IT 系専門学校のプログラミング教育に期待するスキル	【回答選択肢】 1 期待しない 2 あまり期待しない 3 やや期待する 4 期待している
1	複数のプログラミング言語を活用できる	1 2 3 4
2	自らアルゴリズムを考えることができる	1 2 3 4

3	正しいアルゴリズムでプログラムを構築することができる	1	2	3	4
4	一つのプログラム課題に対して、多数の解法を考案することができる	1	2	3	4
5	実行処理速度も考慮したプログラムを構築することができる	1	2	3	4
6	セキュリティを考慮したプログラムを構築することができる	1	2	3	4
7	自身が構築したプログラムのエラーについて、自己解決できる	1	2	3	4
8	可読性の高いプログラムコードを記述することができる	1	2	3	4
9	他者が記述したプログラムコードを正しく読み解くことができる	1	2	3	4
10	UML 図(クラス図、ER 図、シーケンス図等)の読み書きができる	1	2	3	4

（３）自社のシステム・ソフトウェア開発に適用している主な開発モデルはどれですか（該当する番号に○、複数回答可）。

1	ウォーターフォール型開発
2	アジャイル型開発
3	プロトタイプ型開発
4	スパイラル型開発
5	リーンソフトウェア開発
6	スクラム開発

7	DevOps
8	SRE
9	その他(　　)

（４）自社のシステム・ソフトウェア開発で使用している主なプログラム言語はどれですか（該当する番号に○、複数回答可）。

1	Java
2	C＋＋
3	C#
4	PHP
5	Ruby
6	Python
7	JavaScript
8	TypeScript
9	その他(　　)

（５）自社のシステム・ソフトウェア開発で使用している主なフレームワークはどれですか（該当する番号に○、複数回答可）。

1	Spring Framework(Java)
2	Struts2(Java)
3	TreeFrog Framework(C++)
4	ASP.NET(C#)

7	Ruby on Rails(Ruby)
8	Django(Python)
9	React(JavaScript、TypeScript)
10	Vue(JavaScript、TypeScript)

5	Laravel(PHP)		11	その他(　　)
6	CakePHP(PHP)		12	使用していない

（６）自社のシステム・ソフトウェア開発で使用している主なパブリッククラウドサービスはどれですか（該当する番号に○、複数回答可）。

1	Google Cloud Platform
2	Amazon Web Services
3	Microsoft Azure
4	IBM Cloud
5	FUJITSU Hybrid IT Service FJcloud
6	その他(　　)
7	使用していない

（７）システム・ソフトウェア開発に適用しているプログラム品質の基準において、以下の品質を重要視していますか。

番号	自社のソフトウェア製品について、製造過程で留意するプログラム品質	【回答選択肢】 1　重要ではない 2　あまり重要ではない 3　やや重要 4　重要
1	正当性が高いこと （入力条件を満たしたプログラムを実行すると、そのプログラムが確実に完了もしくは出力条件を満たした結果が得られる）	1　2　3　4
2	データ完全性が高いこと （必要なデータが全て揃っていて欠損や不整合がないことを保証する）	1　2　3　4
3	バグが無いこと	1　2　3　4
4	ソースコードの可読性が高いなど、保守性に優れていること （バグなどの不具合要因の修正や、性能や使い易さといった特性を改善でき、製作当初に想定していなかった機能の追加や変更を少ない手間やコストで実施・改変が行える）	1　2　3　4
5	拡張性が高いこと （付加的な機能を追加し、それらの性能をあとから向上させる事が可能である）	1　2　3　4
6	仕様書、設計書等のドキュメントの精度が高いこと	1　2　3　4
7	仕様書、設計書等の必要なドキュメントが網羅されていること	1　2　3　4
8	テスト仕様書の項目数が揃っていること	1　2　3　4
9	リソースを無駄に消費しないプログラムであること	1　2　3　4
10	記法や用語が一貫していること	1　2　3　4
11	プログラムの実行速度、アルゴリズムの精度	1　2　3　4
12	プログラムの持続性（ほかの環境や将来の環境における互換性）	1　2　3　4

（８）今後採用したいと思う新卒のエンジニア人材に求めたいその他の知識・スキル（自由記述）

ご回答いただき、誠にありがとうございます。ご多忙の中、恐れ入りますが、令和 4 年1月 21 日(金)までに本回答用紙(全4ページ)を返信用封筒にてご返信いただきますようお願いいたします。

ご回答いただきました結果を、本事業のカリキュラムと教材の開発に活用させていただきます。

今後とも、何卒よろしくお願いいたします。

2.2.2. IT 系専門学校向け調査アンケートの結果

IT 系専門学校に実施した調査アンケートの結果を以下にまとめる。

質問 1

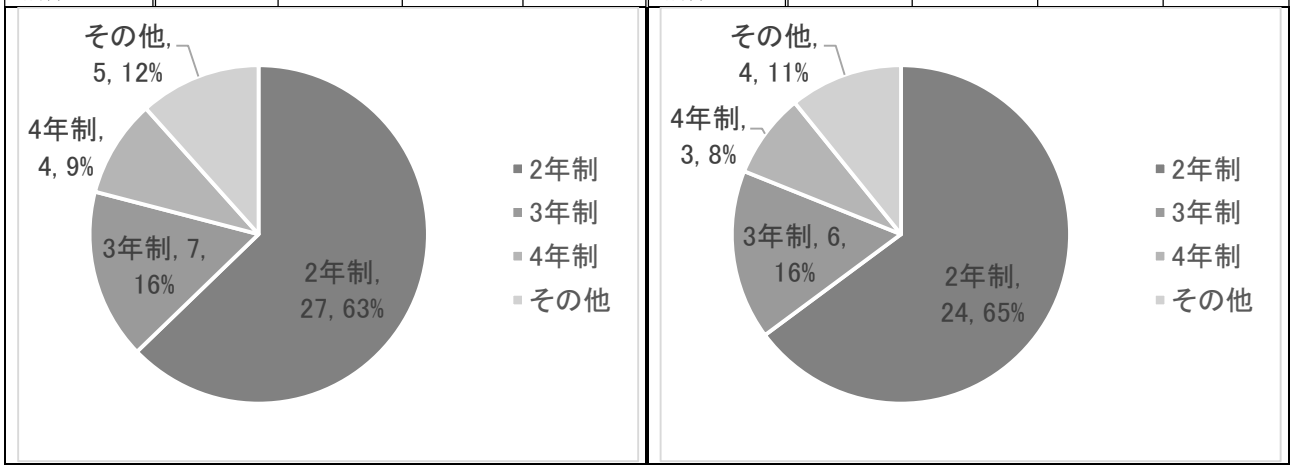
質問 1 では、回答対象の学科を特定するために、「現行カリキュラムの修業年限」「現行カリキュラムの総授業時間数」「現行カリキュラムでプログラミング言語の学習に費やす講義の授業時間数」「現行カリキュラムでプログラミング実習の授業時間数」について確認している。

修業年限は、2 年制が最も多く 27 件（43 件中 62.8%）、3 年制が 7 件（43 件中 16.3%）、4 年制が 4 件（43 件中 9.3%）、その他には、「プログラム科目を有しない学科」、「修業年限が 5 年や 7 年の学科」、「1 回答に複数の修業年限が混ざっている」などの回答が計 5 件（43 件中 11.6%）であった。

また、「プログラミング実習の授業時間数」が「0 時間」の回答が 6 件（2 年制 3 件、3 年制 1 件、4 年制 1 件、その他 1 件）あり、この 6 件を除く 37 件を有効回答データとしている。

・現行カリキュラムの修業年限

全回答の内訳					有効回答データの内訳※				
修業年限	2 年制	3 年制	4 年制	その他	修 業 年限	2 年制	3 年制	4 年制	その他
件数	27 件	7 件	4 件	5 件	件数	24 件	6 件	3 件	4 件
構成比	62.8%	16.3%	9.3%	11.6%	構成比	64.9%	16.2%	8.1%	10.8%



・現行カリキュラムの総授業時間数・プログラミング講義時間数・プログラミング実習時間数

	総授業時間数			講義時間数			実習時間数		
	平均	最小	最大	平均	最小	最大	平均	最小	最大
2 年制	1918	1700	2444	388	0	1088	375	0	1200
3 年制	2702	2400	3066	373	0	690	566	0	1200
4 年制	3963	3960	3966	870	0	1600	864	0	2256

※講義時間数が 0 の回答が 4 件（内 3 件は同一校であるため、講義時間数が 0 の回答は 2 校）

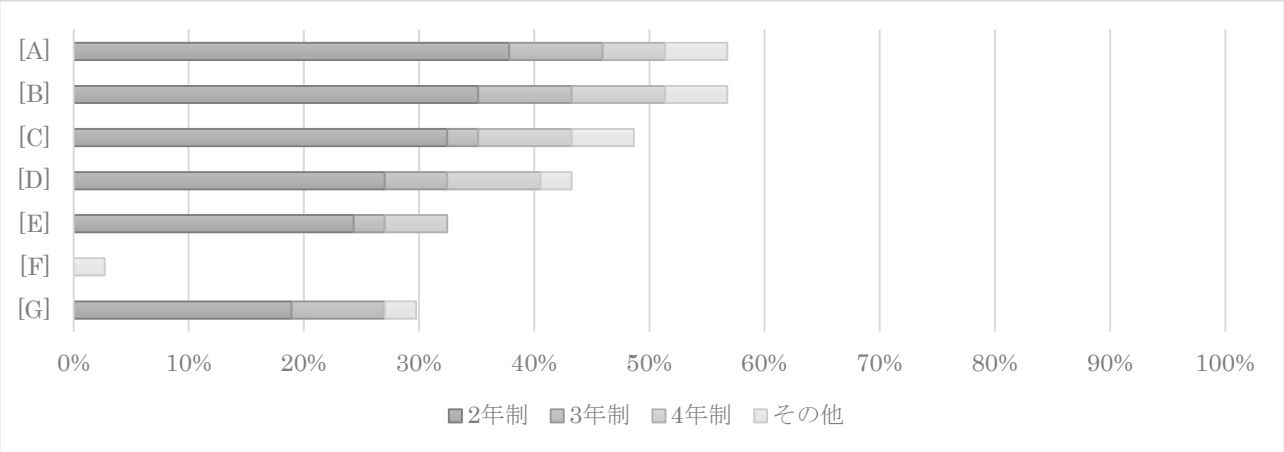
※実習時間数が 0 の回答は 6 件（内 3 件は同一校であるため、実習時間数が 0 の回答は 4 校）

質問 2

質問 2 では、「遠隔授業でプログラミング実習を実施したときに生じた問題」について複数回答可で確認している。

「学生のつまずいている個所が把握しにくい」と「プログラミングにつまずいている学生の個別フォローを行いにくい」の回答が全体の 56.8%で過半数以上であり、かつ、全ての修業年限においても過半数を上回る回答が得られている。

・遠隔授業でプログラミング実習を実施したときに生じた問題の回答率



・遠隔授業でプログラミング実習を実施したときに生じた問題の修業年限別回答率

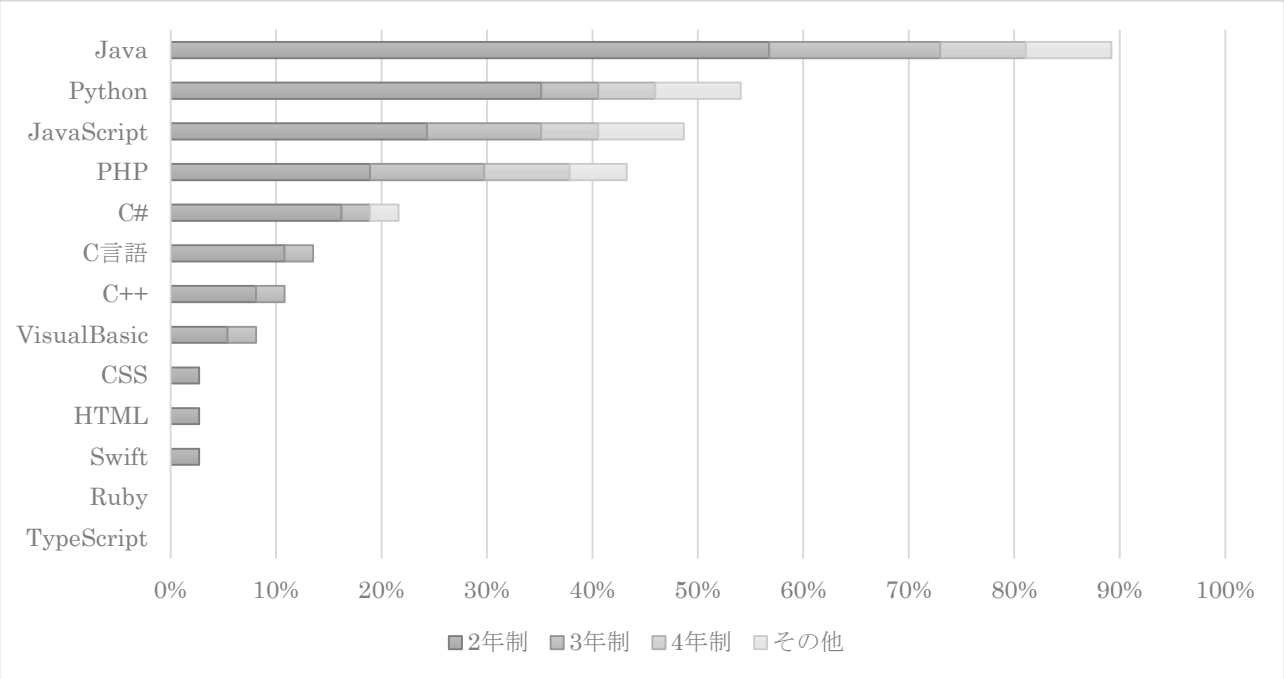
		全体 (37 件)	2 年制 (24 件)	3 年制 (6 件)	4 年制 (3 件)	その他 (4 件)
[A]	学生のつまずいている個所が把握しにくい	56.8%	58.3%	50.0%	66.7%	50.0%
[B]	プログラミングにつまずいている学生の個別フォローを行いにくい	56.8%	54.2%	50.0%	100.0%	50.0%
[C]	学生が真摯にプログラミングに取り組んでいるかを把握しにくい	48.6%	50.0%	16.7%	100.0%	50.0%
[D]	学生が理解しながらプログラミングを進めているかどうかを把握しにくい	43.2%	41.7%	33.3%	100.0%	25.0%
[E]	プログラミングにつまずいている学生を特定しにくい	32.4%	37.5%	16.7%	66.7%	0.0%
[F]	とくに問題は発生していない	2.7%	0.0%	0.0%	0.0%	25.0%
[G]	遠隔授業下ではプログラミング実習を実施していない	29.7%	29.2%	50.0%	0.0%	25.0%

質問３

質問 3 では、「プログラミング実習において、おもに学習している言語」について複数選択可で確認している。

「Java」が全体の 89.2%で最も多くの回答を得ており、全ての修業年限においても 75%以上の回答が得られている。次いで、近年人気の高い「Python」が全体の 54.1%で過半数以上の回答で、続いて、Web システムの開発で使われる「JavaScript」が全体の 48.6%、「PHP」が全体の 43.2%で、これらに 4 つの言語に回答が集中していることが確認できる。

・プログラミング実習において、おもに学習している言語



・プログラミング実習において、おもに学習している言語の修業年限別回答率

	全体 (37 件)	2 年制 (24 件)	3 年制 (6 件)	4 年制 (3 件)	その他 (4 件)
Java	89.2%	87.5%	100.0%	100.0%	75.0%
Python	54.1%	54.2%	33.3%	66.7%	75.0%
JavaScript	48.6%	37.5%	66.7%	66.7%	75.0%
PHP	43.2%	29.2%	66.7%	100.0%	50.0%
C#	21.6%	25.0%	16.7%	0.0%	25.0%
C 言語	13.5%	16.7%	16.7%	0.0%	0.0%
C++	10.8%	12.5%	16.7%	0.0%	0.0%
VisualBasic	8.1%	8.3%	16.7%	0.0%	0.0%
CSS	2.7%	4.2%	0.0%	0.0%	0.0%
HTML	2.7%	4.2%	0.0%	0.0%	0.0%
Swift	2.7%	4.2%	0.0%	0.0%	0.0%
Ruby	0.0%	0.0%	0.0%	0.0%	0.0%
TypeScript	0.0%	0.0%	0.0%	0.0%	0.0%

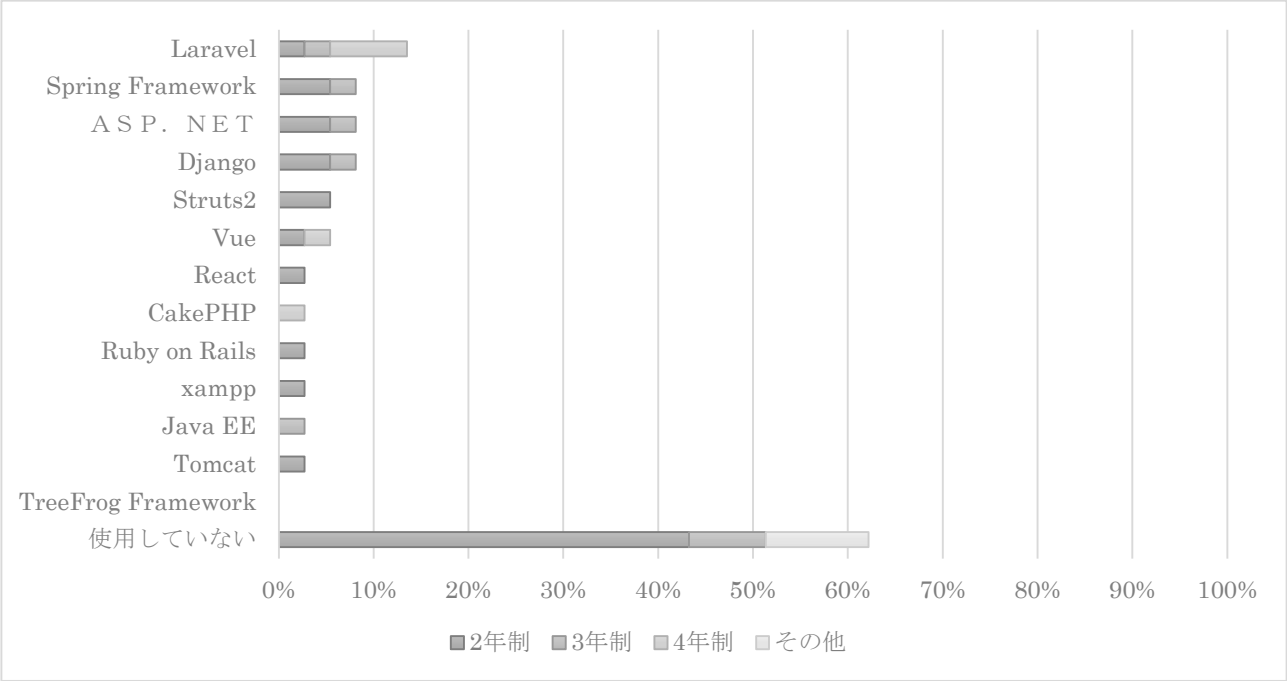
質問４

質問 4 では、「プログラミング実習において、おもに学習しているフレームワーク」について複数選択可で確認している。

最も多く学習されているフレームワークは「Laravel」ではあるが、全体の 13.5%と低い回答率であり、他のフレームワークについては全体の 10%を下回る回答であった。

一方で、「フレームワークを使用していない」が全体の 62.2%の回答で、修業年限別では 2 年制学科が全体の 66.7%、3 年制学科が全体の 50.0%で「フレームワークを使用していない」と回答していることから、専門学校のプログラミング教育において修業年限の短い学科ほどフレームワークを使ったプログラミング教育が進んでいないことが確認できる。

・プログラミング実習において、おもに学習しているフレームワーク



・プログラミング実習において、おもに学習しているフレームワークの修業年限別回答率

	全体 (37 件)	2 年制 (24 件)	3 年制 (6 件)	4 年制 (3 件)	その他 (4 件)
Laravel	13.5%	4.2%	16.7%	100.0%	0.0%
Spring Framework	8.1%	8.3%	16.7%	0.0%	0.0%
ＡＳＰ．ＮＥＴ	8.1%	8.3%	16.7%	0.0%	0.0%
Django	8.1%	8.3%	16.7%	0.0%	0.0%
Struts2	5.4%	8.3%	0.0%	0.0%	0.0%
Vue	5.4%	4.2%	0.0%	33.3%	0.0%
React	2.7%	4.2%	0.0%	0.0%	0.0%
CakePHP	2.7%	0.0%	0.0%	33.3%	0.0%
Ruby on Rails	2.7%	4.2%	0.0%	0.0%	0.0%
xampp	2.7%	4.2%	0.0%	0.0%	0.0%

Java EE	2.7%	0.0%	16.7%	0.0%	0.0%
Tomcat	2.7%	4.2%	0.0%	0.0%	0.0%
TreeFrog Framework	0.0%	0.0%	0.0%	0.0%	0.0%
使用していない	62.2%	66.7%	50.0%	0.0%	100.0%

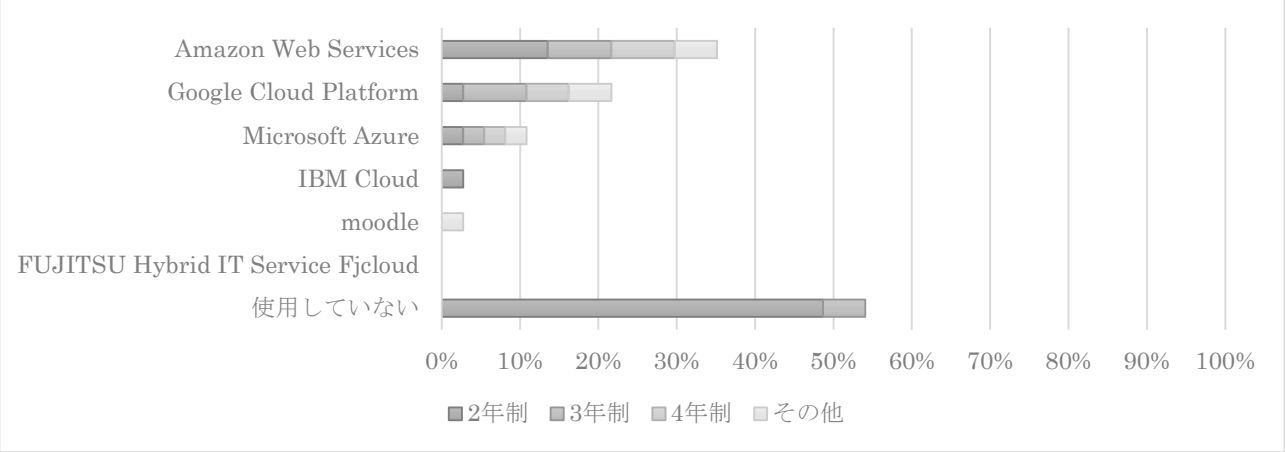
質問５

質問 5 では、「プログラミング実習において、おもに学習しているパブリッククラウド」について複数選択可で確認している。

最も多く学習されているパブリッククラウドは、「AmazonWebServices」が全体の 35.1%で、長い修業年限ほど回答率が高くなっていることが確認できる。

一方で、「パブリッククラウドは使用していない」の回答が全体の 54.1%で、「フレームワークを使用した学習」と同じく、修業年限の短い学科ほどフレームワークを使用したプログラミング教育が進んでいないことが確認できる。

・プログラミング実習において、おもに学習しているパブリッククラウド



・プログラミング実習において、おもに学習しているパブリッククラウドの修業年限別回答率

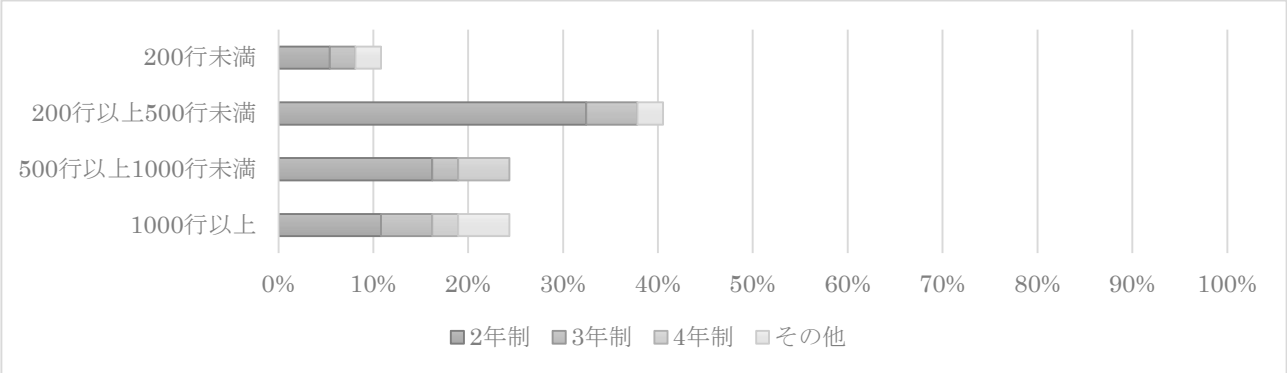
	全体 (37 件)	2 年制 (24 件)	3 年制 (6 件)	4 年制 (3 件)	その他 (4 件)
Amazon Web Services	35.1%	20.8%	50.0%	100.0%	50.0%
Google Cloud Platform	21.6%	4.2%	50.0%	66.7%	50.0%
Microsoft Azure	10.8%	4.2%	16.7%	33.3%	25.0%
IBM Cloud	2.7%	4.2%	0.0%	0.0%	0.0%
moodle	2.7%	0.0%	0.0%	0.0%	25.0%
FUJITSU Hybrid IT Service Fjcloud	0.0%	0.0%	0.0%	0.0%	0.0%
使用していない	54.1%	75.0%	33.3%	0.0%	0.0%

質問６

質問 6 では、「プログラミング実習における最終的な課題のソースコードの量」を確認している。

「200 行以上 500 行未満」が全体の 40.5%で最も高い回答率で、長い修業年限ほど「500 行以上 1000 行未満」「1000 行以上」と回答率が上がり、プログラミング実習での最終的な課題のソースコード量が増えていることが確認できる。

・プログラミング実習における最終的な課題のソースコードの量



・プログラミング実習における最終的な課題のソースコードの量の修業年限別回答率

	全体 (37 件)	2 年制 (24 件)	3 年制 (6 件)	4 年制 (3 件)	その他 (4 件)
200 行未満	10.8%	8.3%	16.7%	0.0%	25.0%
200 行以上 500 行未満	40.5%	50.0%	33.3%	0.0%	25.0%
500 行以上 1000 行未満	24.3%	25.0%	16.7%	66.7%	0.0%
1000 行以上	24.3%	16.7%	33.3%	33.3%	50.0%

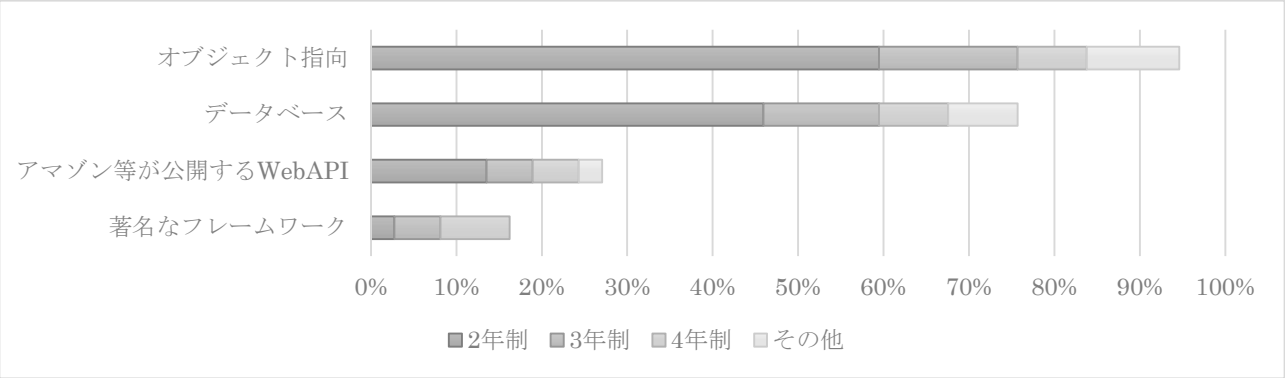
質問 7

質問 7 では、「プログラミング実習における、作成するプログラムの要素」を複数回答可で確認している。

「オブジェクト指向」が全体の 94.6%で、最も高い回答率が得られており、次いで、「データベース」が全体の 75.7%で高い回答率が得られている。

一方で、「アマゾン等が公開する WebAPI」が全体の 27.0%、「著名なフレームワーク」が全体の 16.2%と低い回答率であり、先の質問 4 と質問 5 の結果と同じである。

・プログラミング実習における、作成するプログラムの要素



・プログラミング実習における、作成するプログラムの要素の修業年限別回答率

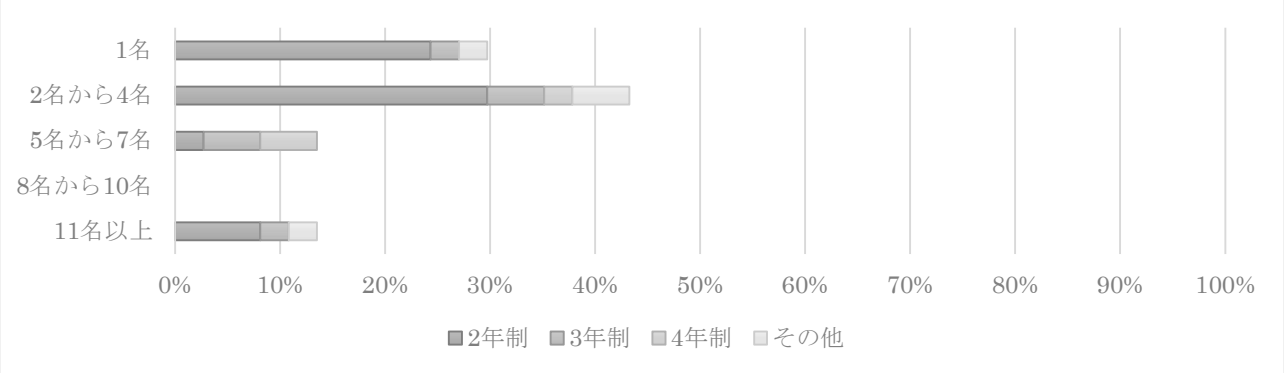
	全体 (37 件)	2 年制 (24 件)	3 年制 (6 件)	4 年制 (3 件)	その他 (4 件)
オブジェクト指向	94.6%	91.7%	100.0%	100.0%	100.0%
データベース	75.7%	70.8%	83.3%	100.0%	75.0%
アマゾン等が公開する WebAPI	27.0%	20.8%	33.3%	66.7%	25.0%
著名なフレームワーク	16.2%	4.2%	33.3%	100.0%	0.0%

質問８

質問 8 では、「プログラミング実習におけるプログラミング課題に取り組む学生の作業単位人数」を確認している。

「2名から4名」が全体の43.2%で最も多い回答を集めており、次いで、「1名」が全体の29.7%で、小規模な作業単位人数に回答が集まっていることが確認できる。

・プログラミング実習におけるプログラミング課題に取り組む学生の作業単位人数



・プログラミング実習におけるプログラミング課題に取り組む学生の作業単位人数の修業年限別回答率

	全体 (37 件)	2 年制 (24 件)	3 年制 (6 件)	4 年制 (3 件)	その他 (4 件)
1 名	29.7%	37.5%	16.7%	0.0%	25.0%
2 名から 4 名	43.2%	45.8%	33.3%	33.3%	50.0%
5 名から 7 名	13.5%	4.2%	33.3%	66.7%	0.0%
8 名から 10 名	0.0%	0.0%	0.0%	0.0%	0.0%
11 名以上	13.5%	12.5%	16.7%	0.0%	25.0%

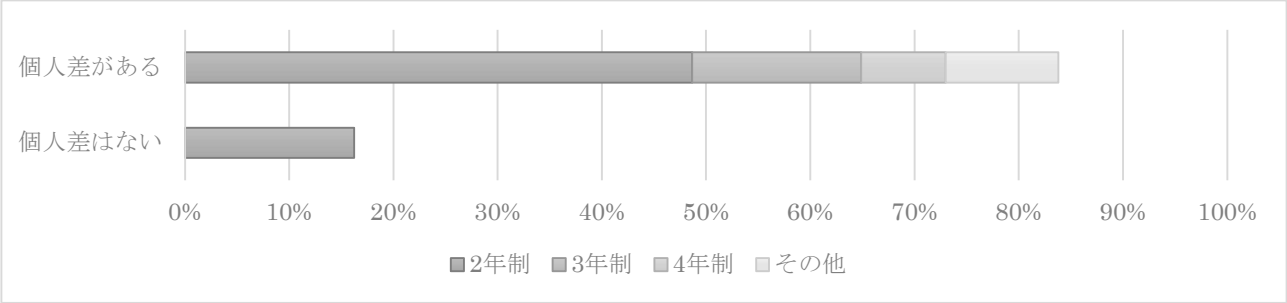
質問９

質問 9 では、「指導担当の教員による指導の個人差」について確認している。

「個人差がある」が全体の 83.8%で回答しており、全ての修業年限においても 75%以上で「個人差がある」に回答している。

一方で、「個人差はない」の回答には、2 年制学科のみに 25.0%の回答があり、短い修業年限の範囲であれば、課題の難易度の程度によって、指導担当教員の知識量や得意な分野などによる指導の個人差があらわれにくくなることも考えられる。

・指導担当の教員による指導の個人差



・指導担当の教員による指導の個人差の修業年限別回答率

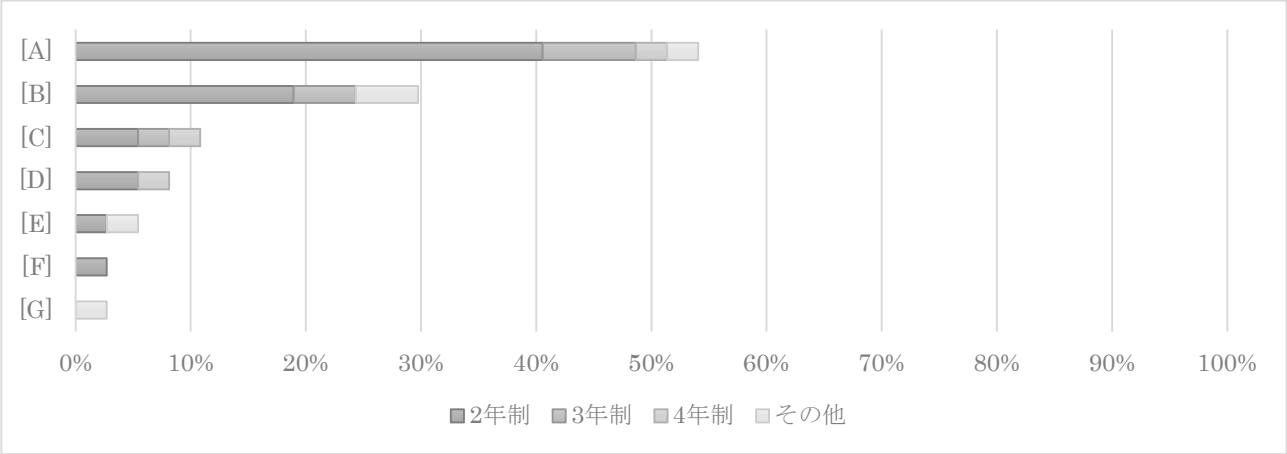
	全体 (37 件)	2 年制 (24 件)	3 年制 (6 件)	4 年制 (3 件)	その他 (4 件)
個人差がある	83.8%	75.0%	100.0%	100.0%	100.0%
個人差はない	16.2%	25.0%	0.0%	0.0%	0.0%

質問１０

質問 10 では、「プログラミング教育におけるプログラム品質に関する指導」について複数回答可で確認している。

最も多い回答は、「プログラム仕様への合致のみ指導している」が全体の 54.1%の回答で、他の回答が全体の 30%未満であることから、プログラム品質に関する指導は限定的になっていること考えられる。

・プログラミング教育におけるプログラム品質に関する指導



・プログラミング教育におけるプログラム品質に関する指導の修業年限別回答率

		全体 (37 件)	2 年制 (24 件)	3 年制 (6 件)	4 年制 (3 件)	その他 (4 件)
[A]	プログラム品質としては、プログラム仕様への合致のみ指導している	54.1%	62.5%	50.0%	33.3%	25.0%
[B]	ソフトウェア工学を主題として扱う科目がカリキュラムに存在する	29.7%	29.2%	33.3%	0.0%	50.0%
[C]	ソフトウェアパターンの利用をプログラム品質と結び付けて指導している	10.8%	8.3%	16.7%	33.3%	0.0%
[D]	ソフトウェアの品質特性モデルに立脚したプログラム評価を指導している	8.1%	8.3%	0.0%	33.3%	0.0%
[E]	コードレビューの実施	5.4%	4.2%	0.0%	0.0%	25.0%
[F]	特別講義形式　産学連携（IVIA との）	2.7%	4.2%	0.0%	0.0%	0.0%
[G]	実施していない	2.7%	0.0%	0.0%	0.0%	25.0%

質問１１－１

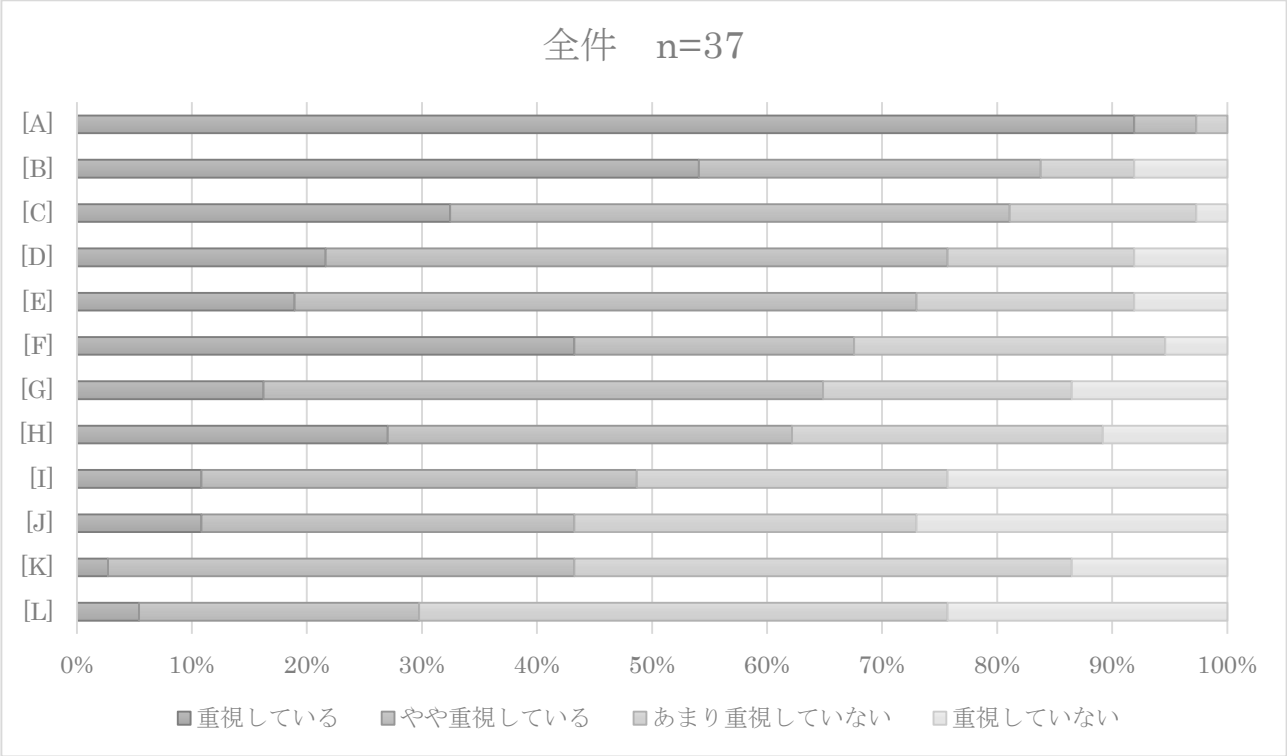
質問 11－1 では、12 の確認事項で「プログラミング実習の指導（改善指導・改修指示）において重視している点」について確認している。

「プログラム実行時に正しい結果が得られているか」に対してのみ、「重視している」の回答率が90%を超えており、続けて、「ソースコードの可読性や保守性が高いか」「バグになりそうなコードがないか」「アルゴリズムの内容を正確に記述できているか」「記法や用語が一貫しているか」「コンパイルエラーにならないが、不適切なアクセス修飾子を使用していないか」「拡張性が高いか（オブジェクト指向に沿ったプログラムか)」「アルゴリズムの使い分けは適切か」が「重視している」と「やや重視している」の回答率を合計した順番であり、ここまでの 8 項目が「重視している」と「やや重視している」の回答率の合計が過半数を超えている項目である。

一方で、「抽象クラスや抽象メソッドが適切に利用されているか」「多態性が適切に利用されているか」「テスト品質は適切か」「リソースを無駄に消費していないか」の 4 項目が「重視している」と「やや重視している」の回答率の合計が過半数を下回っている。

この「重視している」と「やや重視している」の回答率の合計で得られる順番は、プログラミング実習の指導時期に関係しているように思われ、早い段階から指導できる項目が指導において重視されている傾向にあり、指導時期の遅い段階でないと指導できない項目やプログラム設計と関係する項目ほど指導において重視されにくい項目になっているように感じる。

・プログラミング実習の指導（改善指導・改修指示）において重視している点

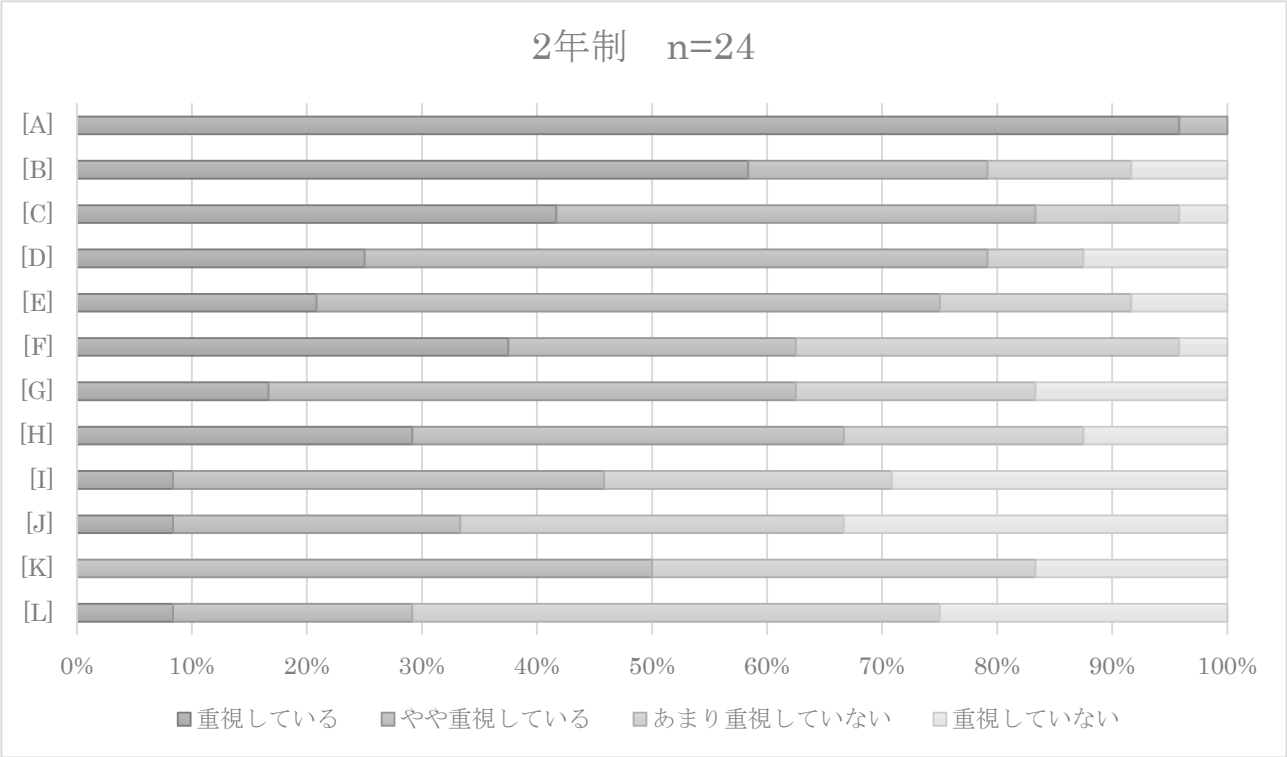


・プログラミング実習の指導（改善指導・改修指示）において重視している点

		重視している	やや重視している	あまり重視していない	重視していない
[A]	プログラム実行時に正しい結果が得られているか	91.9%	5.4%	2.7%	0.0%
[B]	ソースコードの可読性や保守性が高いか	54.1%	29.7%	8.1%	8.1%
[C]	バグになりそうなコードがないか	32.4%	48.6%	16.2%	2.7%
[D]	アルゴリズムの内容を正確に記述できているか	21.6%	54.1%	16.2%	8.1%
[E]	記法や用語が一貫しているか	18.9%	54.1%	18.9%	8.1%
[F]	コンパイルエラーにならないが、不適切なアクセス修飾子を使用していないか	43.2%	24.3%	27.0%	5.4%
[G]	拡張性が高いか（オブジェクト指向に沿ったプログラムか）	16.2%	48.6%	21.6%	13.5%
[H]	アルゴリズムの使い分けは適切か	27.0%	35.1%	27.0%	10.8%
[I]	抽象クラスや抽象メソッドが適切に利用されているか	10.8%	37.8%	27.0%	24.3%
[J]	多態性が適切に利用されているか	10.8%	32.4%	29.7%	27.0%
[K]	テスト品質は適切か	2.7%	40.5%	43.2%	13.5%
[L]	リソースを無駄に消費していないか	5.4%	24.3%	45.9%	24.3%

以下は、2 年制学科のみを抽出した結果を示しているが、全体と近い結果にはなっているものの、プログラミング実習の指導時期や難易度による変化がみられる。

・プログラミング実習の指導（改善指導・改修指示）において重視している点



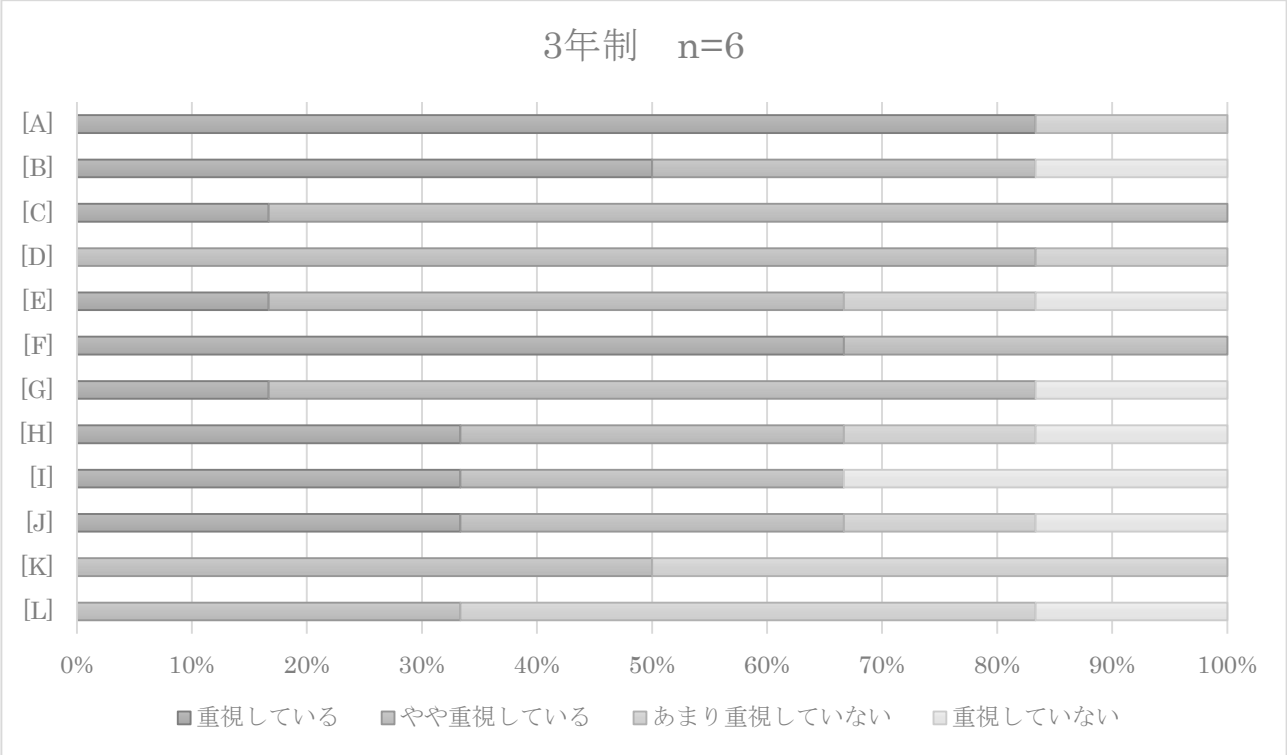
・プログラミング実習の指導（改善指導・改修指示）において重視している点

		重視している	やや重視している	あまり重視していない	重視していない
[A]	プログラム実行時に正しい結果が得られているか	95.8%	4.2%	0.0%	0.0%
[B]	ソースコードの可読性や保守性が高いか	58.3%	20.8%	12.5%	8.3%
[C]	バグになりそうなコードがないか	41.7%	41.7%	12.5%	4.2%
[D]	アルゴリズムの内容を正確に記述できているか	25.0%	54.2%	8.3%	12.5%
[E]	記法や用語が一貫しているか	20.8%	54.2%	16.7%	8.3%
[F]	コンパイルエラーにならないが、不適切なアクセス修飾子を使用していないか	37.5%	25.0%	33.3%	4.2%
[G]	拡張性が高いか（オブジェクト指向に沿ったプログラムか）	16.7%	45.8%	20.8%	16.7%
[H]	アルゴリズムの使い分けは適切か	29.2%	37.5%	20.8%	12.5%
[I]	抽象クラスや抽象メソッドが適切に利用されているか	8.3%	37.5%	25.0%	29.2%
[J]	多態性が適切に利用されているか	8.3%	25.0%	33.3%	33.3%

[K]	テスト品質は適切か	0.0%	50.0%	33.3%	16.7%
[L]	リソースを無駄に消費していないか	8.3%	20.8%	45.8%	25.0%

以下は、3 年制学科のみを抽出した結果を示しているが、回答件数が少ないため、傾向が見えにくい結果になっている。

・プログラミング実習の指導（改善指導・改修指示）において重視している点



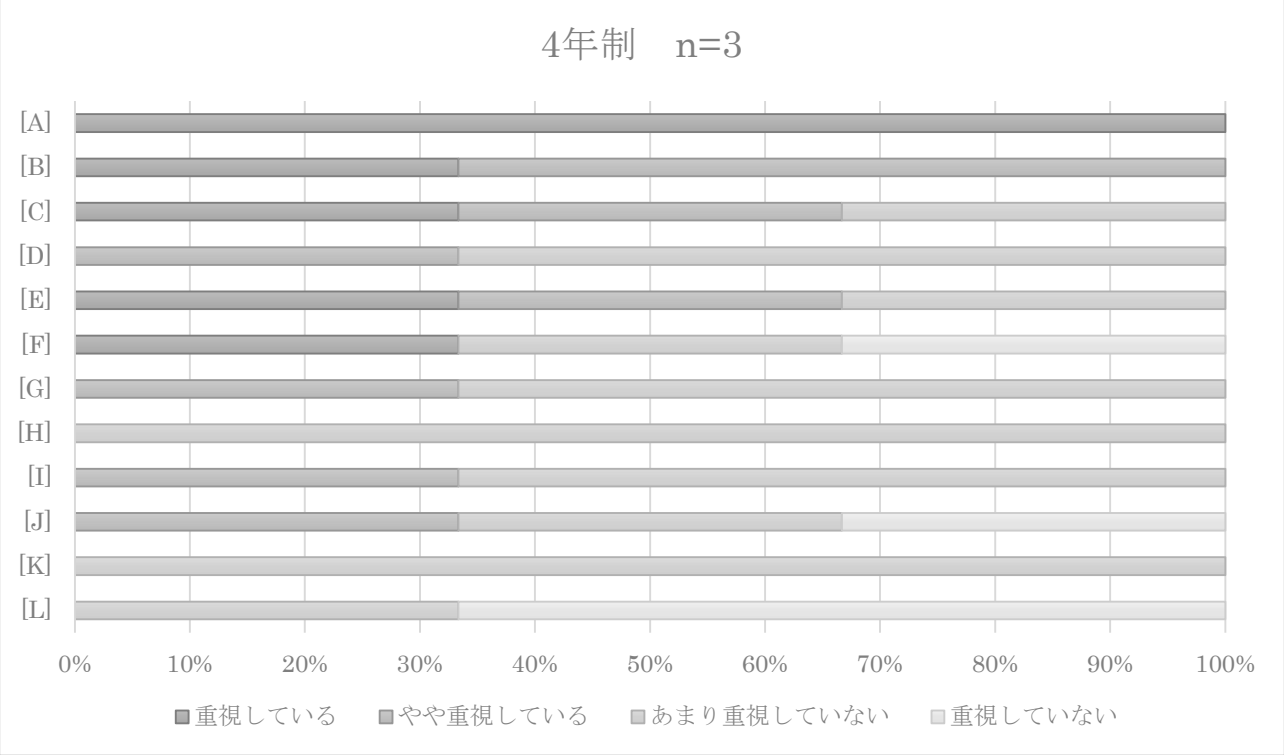
・プログラミング実習の指導（改善指導・改修指示）において重視している点

		重視している	やや重視している	あまり重視していない	重視していない
[A]	プログラム実行時に正しい結果が得られているか	83.3%	0.0%	16.7%	0.0%
[B]	ソースコードの可読性や保守性が高いか	50.0%	33.3%	0.0%	16.7%
[C]	バグになりそうなコードがないか	16.7%	83.3%	0.0%	0.0%
[D]	アルゴリズムの内容を正確に記述できているか	0.0%	83.3%	16.7%	0.0%
[E]	記法や用語が一貫しているか	16.7%	50.0%	16.7%	16.7%
[F]	コンパイルエラーにならないが、不適切なアクセス修飾子を使用していないか	66.7%	33.3%	0.0%	0.0%
[G]	拡張性が高いか（オブジェクト指向に沿ったプログラムか）	16.7%	66.7%	0.0%	16.7%

[H]	アルゴリズムの使い分けは適切か	33.3%	33.3%	16.7%	16.7%
[I]	抽象クラスや抽象メソッドが適切に利用されているか	33.3%	33.3%	0.0%	33.3%
[J]	多態性が適切に利用されているか	33.3%	33.3%	16.7%	16.7%
[K]	テスト品質は適切か	0.0%	50.0%	50.0%	0.0%
[L]	リソースを無駄に消費していないか	0.0%	33.3%	50.0%	16.7%

以下は、4 年制学科のみを抽出した結果を示しているが、回答件数が少ないため、傾向が見えにくい結果になっている。

・プログラミング実習の指導（改善指導・改修指示）において重視している点



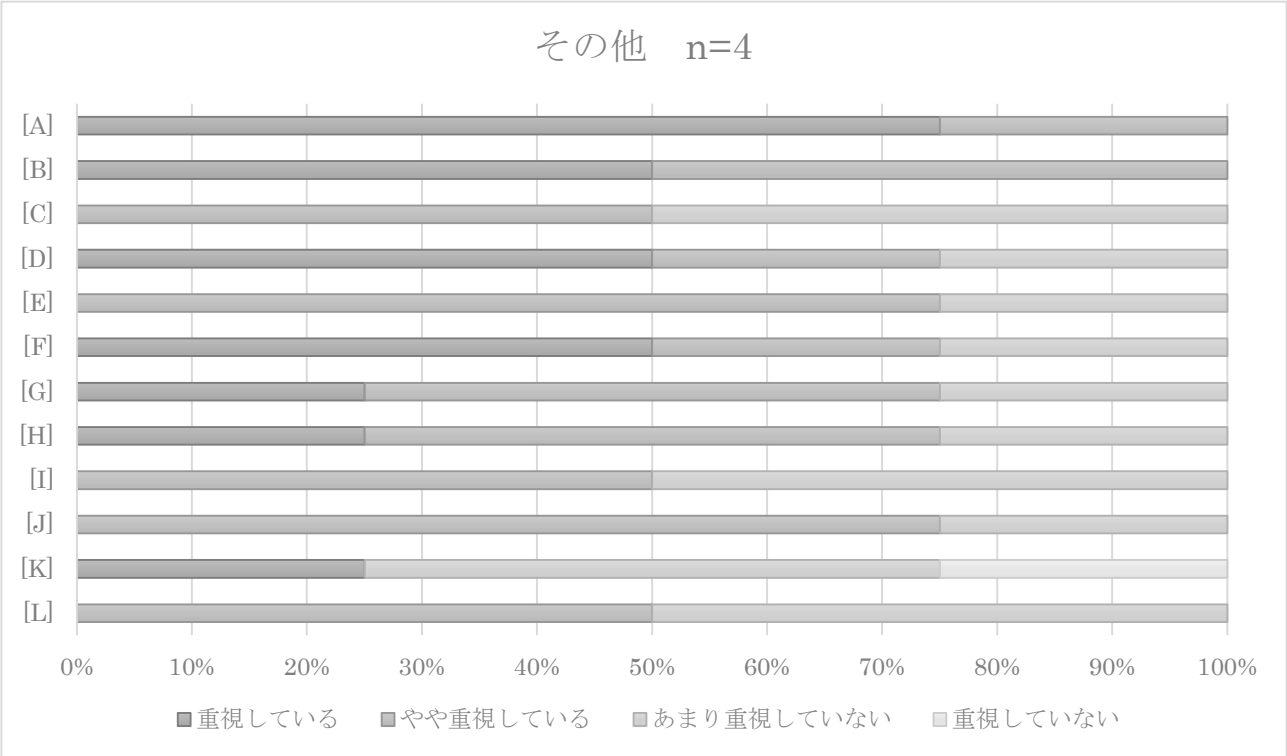
・プログラミング実習の指導（改善指導・改修指示）において重視している点

		重視している	やや重視している	あまり重視していない	重視していない
[A]	プログラム実行時に正しい結果が得られているか	100.0%	0.0%	0.0%	0.0%
[B]	ソースコードの可読性や保守性が高いか	33.3%	66.7%	0.0%	0.0%
[C]	バグになりそうなコードがないか	33.3%	33.3%	33.3%	0.0%
[D]	アルゴリズムの内容を正確に記述できているか	0.0%	33.3%	66.7%	0.0%
[E]	記法や用語が一貫しているか	33.3%	33.3%	33.3%	0.0%
[F]	コンパイルエラーにならないが、不適切なアクセス修	33.3%	0.0%	33.3%	33.3%

	飾子を使用していないか				
[G]	拡張性が高いか（オブジェクト指向に沿ったプログラムか)	0.0%	33.3%	66.7%	0.0%
[H]	アルゴリズムの使い分けは適切か	0.0%	0.0%	100.0%	0.0%
[I]	抽象クラスや抽象メソッドが適切に利用されているか	0.0%	33.3%	66.7%	0.0%
[J]	多態性が適切に利用されているか	0.0%	33.3%	33.3%	33.3%
[K]	テスト品質は適切か	0.0%	0.0%	100.0%	0.0%
[L]	[リソースを無駄に消費していないか]	0.0%	0.0%	33.3%	66.7%

以下は、その他（2 年制、3 年制、4 年制以外）のみを抽出した結果を示しているが、回答件数が少ないため、傾向が見えにくい結果になっている。

・プログラミング実習の指導（改善指導・改修指示）において重視している点



・プログラミング実習の指導（改善指導・改修指示）において重視している点

		重視している	やや重視している	あまり重視していない	重視していない
[A]	プログラム実行時に正しい結果が得られているか	75.0%	25.0%	0.0%	0.0%
[B]	ソースコードの可読性や保守性が高いか	50.0%	50.0%	0.0%	0.0%

[C]	バグになりそうなコードがないか	0.0%	50.0%	50.0%	0.0%
[D]	アルゴリズムの内容を正確に記述できているか	50.0%	25.0%	25.0%	0.0%
[E]	記法や用語が一貫しているか	0.0%	75.0%	25.0%	0.0%
[F]	コンパイルエラーにならないが、不適切なアクセス修飾子を使用していないか	50.0%	25.0%	25.0%	0.0%
[G]	拡張性が高いか（オブジェクト指向に沿ったプログラムか)	25.0%	50.0%	25.0%	0.0%
[H]	アルゴリズムの使い分けは適切か	25.0%	50.0%	25.0%	0.0%
[I]	抽象クラスや抽象メソッドが適切に利用されているか	0.0%	50.0%	50.0%	0.0%
[J]	多態性が適切に利用されているか	0.0%	75.0%	25.0%	0.0%
[K]	テスト品質は適切か	25.0%	0.0%	50.0%	25.0%
[L]	リソースを無駄に消費していないか	0.0%	50.0%	50.0%	0.0%

質問１１－２

質問 11－2 では、以下の 4 項目で「プログラミング実習の指導（改善指導・改修指示）において、プログラミング以外で重視している点」を確認している。

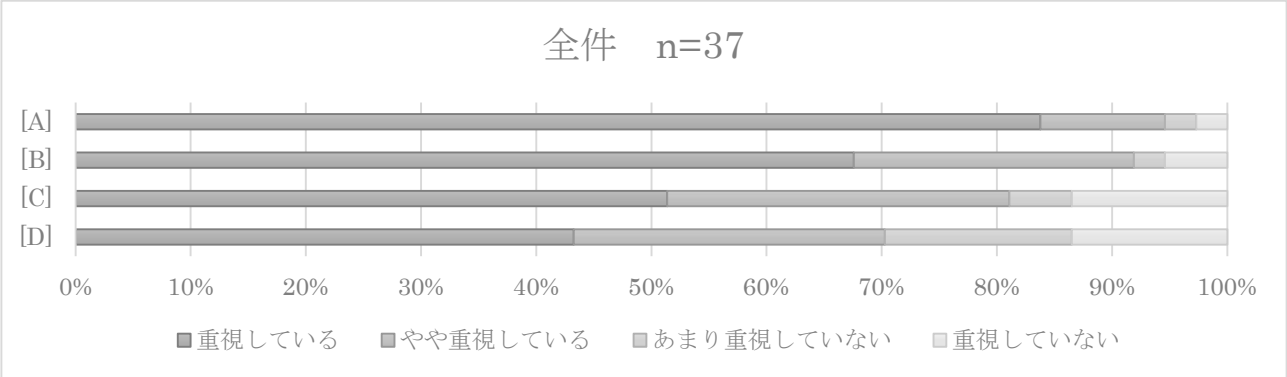
プログラミング以外で「出席率」と「授業態度」に対する指導において、「重視している」と「やや重視している」の回答率の合計が 90%を超えており、非常に強く重視していることが確認できる。

また、「チーム内のコミュニケーション」に対する指導においては、「重視している」と「やや重視している」の回答率の合計が 80%を超えており、強く重視していることが確認できる。

さらに、「チーム内のマネジメント」に対する指導においても、「重視している」と「やや重視している」の回答率の合計が 70%を超えており、やや強く重視していることが確認できる。

この結果は、質問 11-1 で確認している項目の中でも、上位に入る結果になっていることから、プログラミング実習の指導の中で、ヒューマンリテラシー面の指導が強く重視されていることが確認できる。

・プログラミング実習の指導（改善指導・改修指示）において、プログラミング以外で重視している点

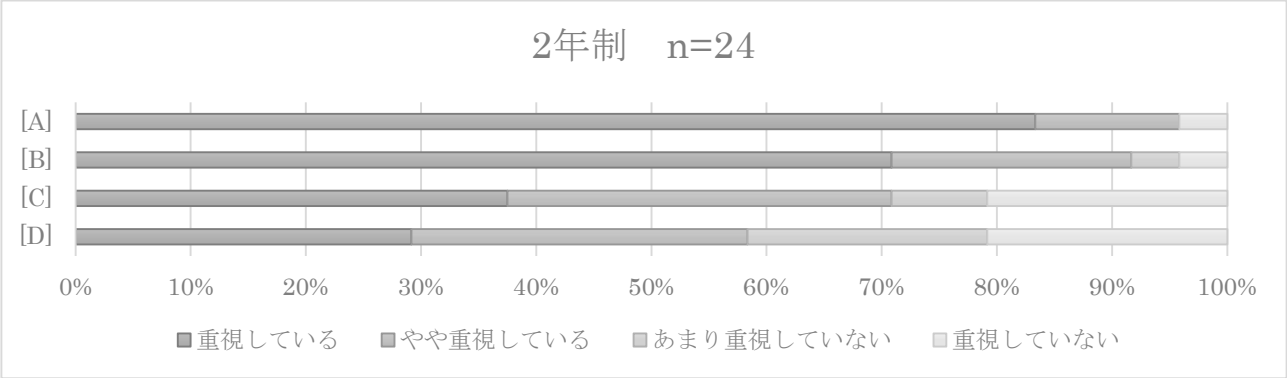


・プログラミング実習の指導（改善指導・改修指示）において、プログラミング以外で重視している点

		重視している	やや重視している	あまり重視していない	重視していない
[A]	出席率が高いかどうか	83.8%	10.8%	2.7%	2.7%
[B]	真摯な授業態度かどうか	67.6%	24.3%	2.7%	5.4%
[C]	（作業単位が複数の場合） チーム内のコミュニケーションが良好か	51.4%	29.7%	5.4%	13.5%
[D]	（作業単位が複数の場合） チーム内のマネジメントが良好か	43.2%	27.0%	16.2%	13.5%

以下は、2 年制学科のみを抽出した結果を示しているが、全体とほぼ同じ結果にはなっていることが確認できる。

・プログラミング実習の指導（改善指導・改修指示）において、プログラミング以外で重視している点

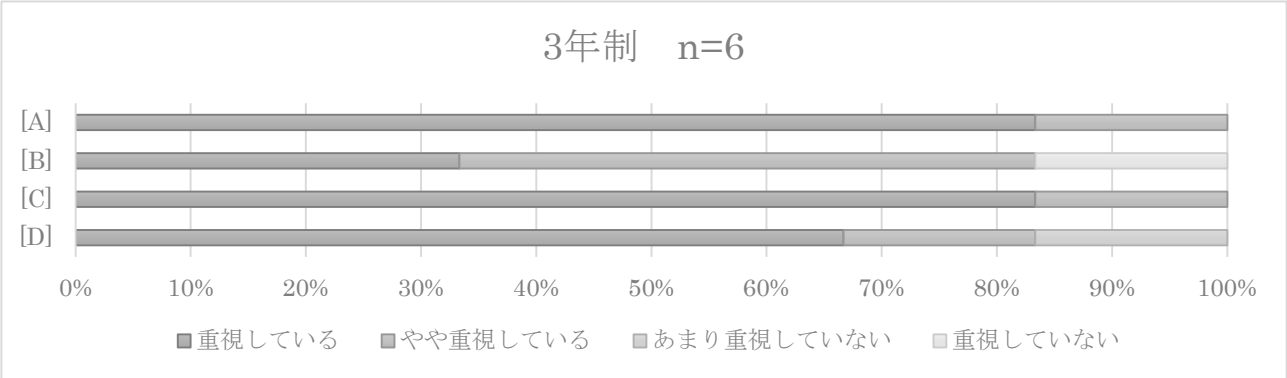


・プログラミング実習の指導（改善指導・改修指示）において、プログラミング以外で重視している点

		重視している	やや重視している	あまり重視していない	重視していない
[A]	出席率が高いかどうか	83.3%	12.5%	0.0%	4.2%
[B]	真摯な授業態度かどうか	70.8%	20.8%	4.2%	4.2%
[C]	（作業単位が複数の場合） チーム内のコミュニケーションが良好か	37.5%	33.3%	8.3%	20.8%
[D]	（作業単位が複数の場合） チーム内のマネジメントが良好か	29.2%	29.2%	20.8%	20.8%

以下は、3 年制学科のみを抽出した結果を示しているが、回答件数が少ないため、傾向が見えにくい結果になっている。

・プログラミング実習の指導（改善指導・改修指示）において、プログラミング以外で重視している点

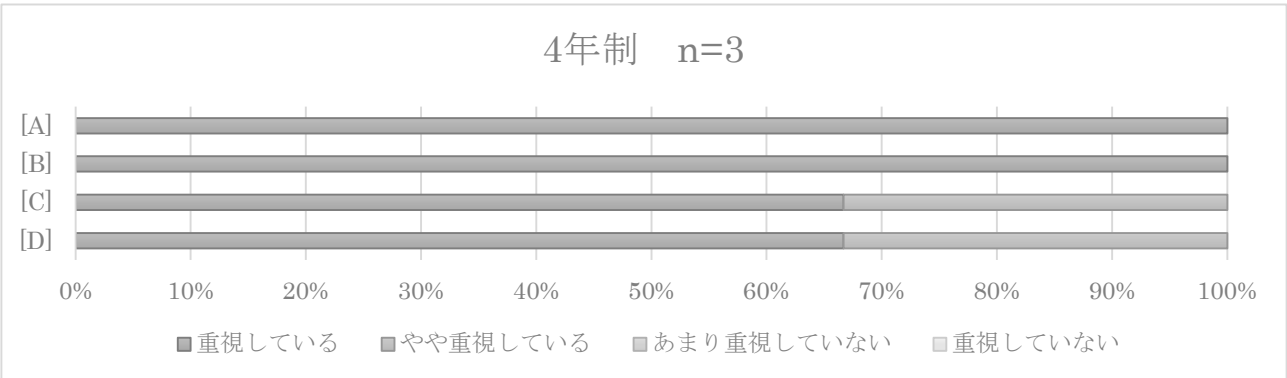


・プログラミング実習の指導（改善指導・改修指示）において、プログラミング以外で重視している点

		重視している	やや重視している	あまり重視していない	重視していない
[A]	出席率が高いかどうか	83.3%	16.7%	0.0%	0.0%
[B]	真摯な授業態度かどうか	33.3%	50.0%	0.0%	16.7%
[C]	（作業単位が複数の場合） チーム内のコミュニケーションが良好か	83.3%	16.7%	0.0%	0.0%
[D]	（作業単位が複数の場合） チーム内のマネジメントが良好か	66.7%	16.7%	16.7%	0.0%

以下は、4 年制学科のみを抽出した結果を示しているが、回答件数が少ないため、傾向が見えにくい結果になっている。

・プログラミング実習の指導（改善指導・改修指示）において、プログラミング以外で重視している点

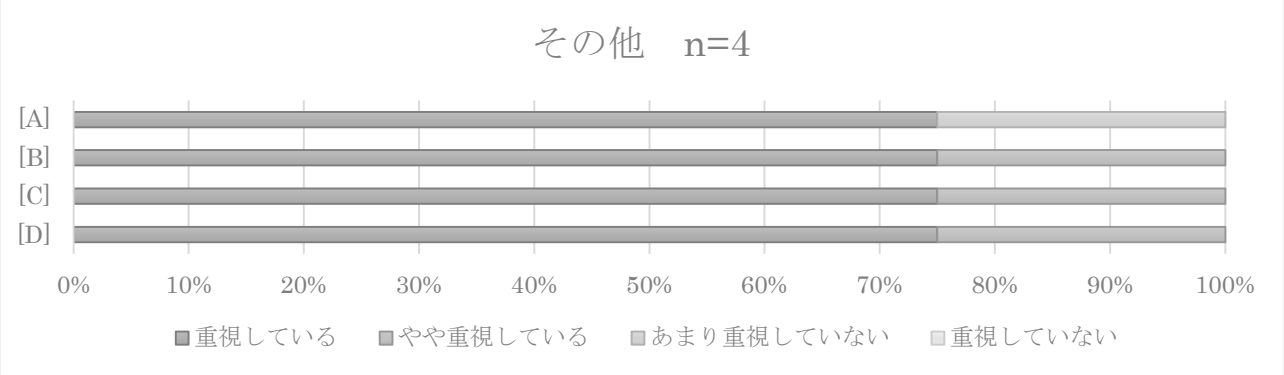


・プログラミング実習の指導（改善指導・改修指示）において、プログラミング以外で重視している点

		重視している	やや重視している	あまり重視していない	重視していない
[A]	出席率が高いかどうか	100.0%	0.0%	0.0%	0.0%
[B]	真摯な授業態度かどうか	100.0%	0.0%	0.0%	0.0%
[C]	（作業単位が複数の場合） チーム内のコミュニケーションが良好か	66.7%	33.3%	0.0%	0.0%
[D]	（作業単位が複数の場合） チーム内のマネジメントが良好か	100.0%	0.0%	0.0%	0.0%

以下は、その他（2 年制、3 年制、4 年制以外）のみを抽出した結果を示しているが、回答件数が少ないため、傾向が見えにくい結果になっている。

・プログラミング実習の指導（改善指導・改修指示）において、プログラミング以外で重視している点



・プログラミング実習の指導（改善指導・改修指示）において、プログラミング以外で重視している点

		重視している	やや重視している	あまり重視していない	重視していない
[A]	出席率が高いかどうか	75.0%	0.0%	25.0%	0.0%

[B]	真摯な授業態度かどうか	75.0%	25.0%	0.0%	0.0%
[C]	（作業単位が複数の場合） チーム内のコミュニケーションが良好か	75.0%	25.0%	0.0%	0.0%
[D]	（作業単位が複数の場合） チーム内のマネジメントが良好か	75.0%	25.0%	0.0%	0.0%

質問１１－３

質問 11－3 では、「プログラミング実習の指導（改善指導・改修指示）において、質問 11－1、質問 11－2 以外の項目で重視している点」を自由記述で確認している。

2 年制	与えられた課題を期限内に提出しているか。
2 年制	速度（指定時間内にどれだけ達成できているか）
2 年制	動く事および口頭試問において、適切かつ論理的に回答が出来ているか。また、結果をイメージしてから作成しているのかなどエンジニアとして開発前段階の姿勢を指示および指導している
2 年制	・工程管理（進捗管理）・プログラミング力がまだ身についていない学生でも、ドキュメント作成、テストなどプログラミング技術だけでない部分でサポートができているか
2 年制	要件定義から内部設計書までの記述内容との整合性
2 年制	実行エラー発生時の解析用ログの埋め込みとログの解析スキル
3 年制	課題の提出回数、提出タイミング

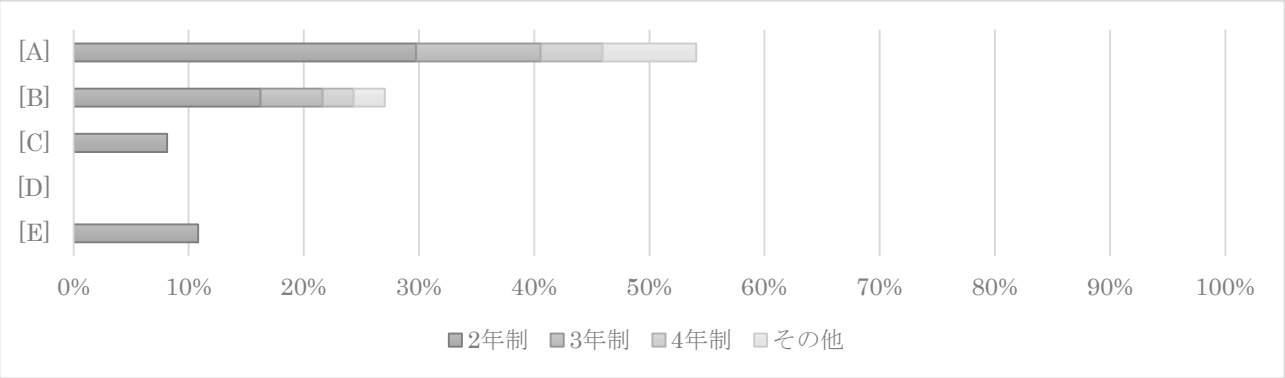
質問１２

質問 12 では、「プログラミング実習における学生の成果物に対する教員による評価の状況」について確認している。

全体の 27.0%が「十分に評価できている」と回答している一方で、全体の 54.7%が「評価できる教員はいるが、評価をする時間が足りていないため、十分には評価できていない」に回答している。

また、「評価できる教員はいるが、評価をする時間が足りていないため、十分には評価できていない」は、長い修業年限ほど回答率が高くなっていることが確認できる。

・プログラミング実習における学生の成果物に対する教員による評価の状況



・プログラミング実習における学生の成果物に対する教員による評価の状況

		全体 (37 件)	2 年制 (24 件)	3 年制 (6 件)	4 年制 (3 件)	その他 (4 件)
[A]	評価できる教員はいるが、評価をする時間が足りていないため、十分には評価できていない	54.1%	45.8%	66.7%	66.7%	75.0%
[B]	十分に評価できている	27.0%	25.0%	33.3%	33.3%	25.0%
[C]	もともとプログラミング実習における成果物を評価する意図はない	8.1%	12.5%	0.0%	0.0%	0.0%
[D]	評価できる教員がいないため、まったく評価できていない	0.0%	0.0%	0.0%	0.0%	0.0%
[E]	その他	10.8%	16.7%	0.0%	0.0%	0.0%

・[E]その他　の回答

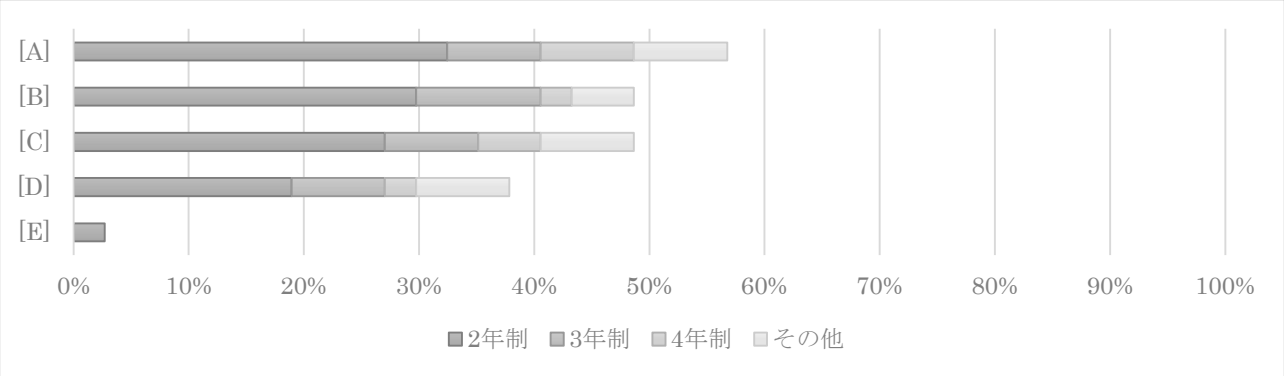
ソフトウェア会社へ勤務している卒業生に評価を一部依頼している。
年３～４回行う作品発表会での発表内容を評価としている
プログラミングの得意な学生に対する評価は十分だが、不得意な学生の評価は、細かいレベルでどこまで出来て、どこまで出来ていないかが十分に評価できてない
未回答

質問１３

質問 13 では、「プログラミング実習における学生の成果物に対する教員の評価方法」について複数回答可で確認している。

「実習後に学生にプログラムを提出させ、教員環境でソースコードの中身を確認している」が全体の 56.8%で過半数を超える回答率があり、次いで、「実習後に学生にプログラムを提出させ、教員環境で実行結果を確認している」と「実習中に学生開発環境でプログラムを実行する画面と、実行結果を確認している」が 48.6%で、「実習後にプログラムを提出させて確認する」評価方法が若干多いことが確認できる。

・プログラミング実習における学生の成果物に対する教員の評価方法



・プログラミング実習における学生の成果物に対する教員の評価方法

		全体 (37 件)	2 年制 (24 件)	3 年制 (6 件)	4 年制 (3 件)	その他 (4 件)
[A]	実習後に学生にプログラムを提出させ、教員環境でソースコードの中身を確認している	56.8%	50.0%	50.0%	100.0%	75.0%
[B]	実習後に学生にプログラムを提出させ、教員環境で実行結果を確認している	48.6%	45.8%	66.7%	33.3%	50.0%
[C]	実習中に学生開発環境でプログラムを実行する画面と、実行結果を確認している	48.6%	41.7%	50.0%	66.7%	75.0%
[D]	実習中に学生開発環境でプログラムのソースコードの中身を目視で確認している	37.8%	29.2%	50.0%	33.3%	75.0%
[E]	上記に加えてソースレビューを実施	2.7%	4.2%	0.0%	0.0%	0.0%

質問１４－１

質問 14－1 では、以下の 12 項目で「プログラミング実習における学生の成果物に対する評価で重視している点」について確認している。

「プログラム実行時に正しい結果が得られているか」に対してのみ、「重視している」の回答率が 90%を超えており、評価において非常に強く重視していることが確認できる。

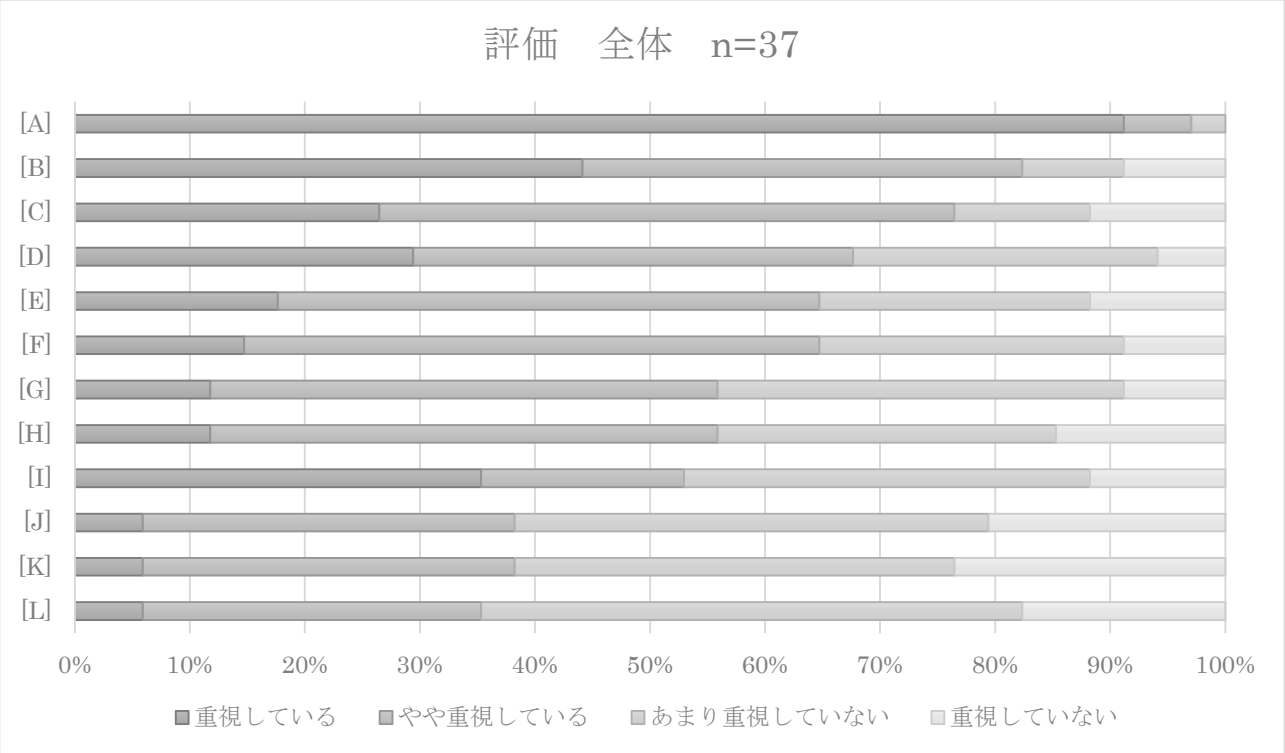
続けて、「ソースコードの可読性や保守性が高いか」が「重視している」と「やや重視している」の回答率の合計が 80%を超え、「アルゴリズムの内容を正確に記述できているか」が 70%を超えており、評価において強く重視していることが確認できる。

さらに、「バグになりそうなコードがないか」「アルゴリズムの使い分けは適切か」「記法や用語が一貫しているか」が 60%を超え、「拡張性が高いか（オブジェクト指向に沿ったプログラムか)」「テスト品質は適切か」「コンパイルエラーにならないが、不適切なアクセス修飾子を使用していないか」が「重視している」と「やや重視している」の回答率の合計が過半数を超えている項目である。

一方で、「リソースを無駄に消費していないか」「抽象クラスや抽象メソッドが適切に利用されているか」「多態性が適切に利用されているか」の 3 項目が「重視している」と「やや重視している」の回答率の合計が過半数を下回っている。

この結果は、質問 11－1 と同様に、学生の成果物に対する評価においても、指導時期や指導の難易度に関係しているように思われる。

・プログラミング実習における学生の成果物に対する評価で重視している点

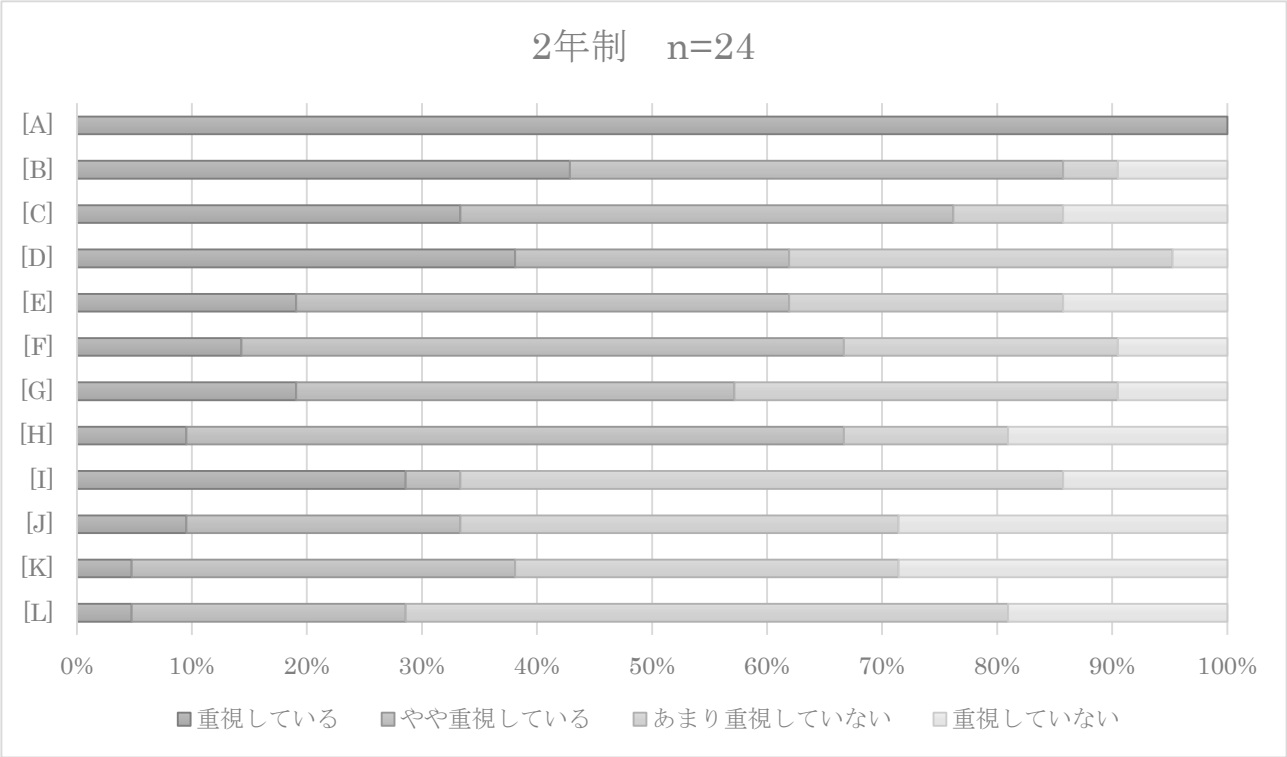


・プログラミング実習における学生の成果物に対する評価で重視している点

		重視している	やや重視している	あまり重視していない	重視していない
[A]	プログラム実行時に正しい結果が得られているか	91.2%	5.9%	2.9%	0.0%
[B]	ソースコードの可読性や保守性が高いか	44.1%	38.2%	8.8%	8.8%
[C]	アルゴリズムの内容を正確に記述できているか	26.5%	50.0%	11.8%	11.8%
[D]	バグになりそうなコードがないか	29.4%	38.2%	26.5%	5.9%
[E]	アルゴリズムの使い分けは適切か	17.6%	47.1%	23.5%	11.8%
[F]	記法や用語が一貫しているか	14.7%	50.0%	26.5%	8.8%
[G]	拡張性が高いか（オブジェクト指向に沿ったプログラムか)	11.8%	44.1%	35.3%	8.8%
[H]	テスト品質は適切か	11.8%	44.1%	29.4%	14.7%
[I]	コンパイルエラーにならないが、不適切なアクセス修飾子を使用していないか	35.3%	17.6%	35.3%	11.8%
[J]	リソースを無駄に消費していないか	5.9%	32.4%	41.2%	20.6%
[K]	抽象クラスや抽象メソッドが適切に利用されているか	5.9%	32.4%	38.2%	23.5%
[L]	多態性が適切に利用されているか	5.9%	29.4%	47.1%	17.6%

以下は、2 年制学科のみを抽出した結果を示しているが、全体と近い結果にはなっており、指導時期や難易度によって若干の変化がみられる。

・プログラミング実習における学生の成果物に対する評価で重視している点



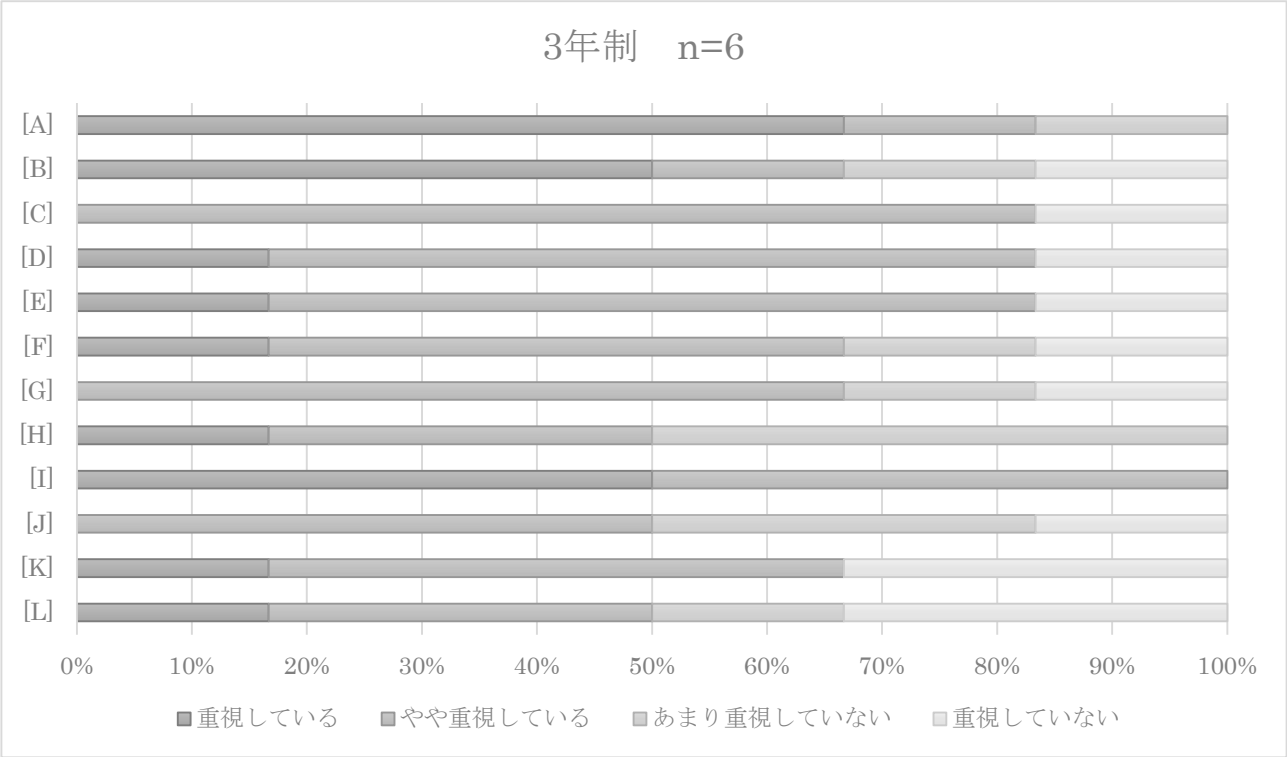
・プログラミング実習における学生の成果物に対する評価で重視している点

		重視している	やや重視している	あまり重視していない	重視していない
[A]	プログラム実行時に正しい結果が得られているか	100.0%	0.0%	0.0%	0.0%
[B]	ソースコードの可読性や保守性が高いか	42.9%	42.9%	4.8%	9.5%
[C]	アルゴリズムの内容を正確に記述できているか	33.3%	42.9%	9.5%	14.3%
[D]	バグになりそうなコードがないか	38.1%	23.8%	33.3%	4.8%
[E]	アルゴリズムの使い分けは適切か	19.0%	42.9%	23.8%	14.3%
[F]	記法や用語が一貫しているか	14.3%	52.4%	23.8%	9.5%
[G]	拡張性が高いか（オブジェクト指向に沿ったプログラムか)	19.0%	38.1%	33.3%	9.5%
[H]	テスト品質は適切か	9.5%	57.1%	14.3%	19.0%
[I]	コンパイルエラーにならないが、不適切なアクセス修飾子を使用していないか	28.6%	4.8%	52.4%	14.3%
[J]	リソースを無駄に消費していないか	9.5%	23.8%	38.1%	28.6%

[K]	抽象クラスや抽象メソッドが適切に利用されているか	4.8%	33.3%	33.3%	28.6%
[L]	多態性が適切に利用されているか	4.8%	23.8%	52.4%	19.0%

以下は、3 年制学科のみを抽出した結果を示しているが、回答件数が少ないため、傾向が見えにくい結果になっている。

・プログラミング実習における学生の成果物に対する評価で重視している点



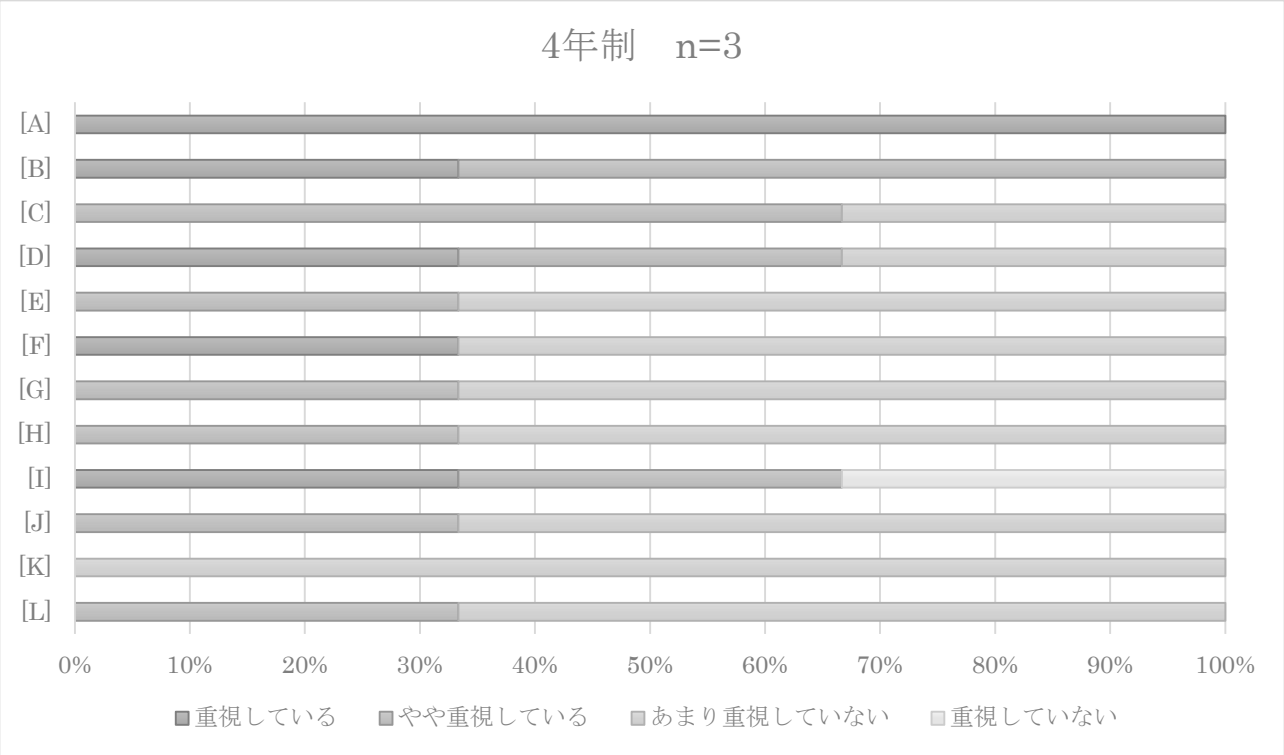
・プログラミング実習における学生の成果物に対する評価で重視している点

		重視している	やや重視している	あまり重視していない	重視していない
[A]	プログラム実行時に正しい結果が得られているか	66.7%	16.7%	16.7%	0.0%
[B]	ソースコードの可読性や保守性が高いか	16.7%	66.7%	0.0%	16.7%
[C]	アルゴリズムの内容を正確に記述できているか	50.0%	50.0%	0.0%	0.0%
[D]	バグになりそうなコードがないか	16.7%	33.3%	16.7%	33.3%
[E]	アルゴリズムの使い分けは適切か	16.7%	50.0%	0.0%	33.3%
[F]	記法や用語が一貫しているか	50.0%	16.7%	16.7%	16.7%
[G]	拡張性が高いか（オブジェクト指向に沿ったプログラムか）	0.0%	66.7%	16.7%	16.7%

[H]	テスト品質は適切か	16.7%	33.3%	50.0%	0.0%
[I]	コンパイルエラーにならないが、不適切なアクセス修飾子を使用していないか	0.0%	50.0%	33.3%	16.7%
[J]	リソースを無駄に消費していないか	16.7%	50.0%	16.7%	16.7%
[K]	抽象クラスや抽象メソッドが適切に利用されているか	16.7%	66.7%	0.0%	16.7%
[L]	多態性が適切に利用されているか	0.0%	83.3%	0.0%	16.7%

以下は、4 年制学科のみを抽出した結果を示しているが、回答件数が少ないため、傾向が見えにくい結果になっている。

・プログラミング実習における学生の成果物に対する評価で重視している点



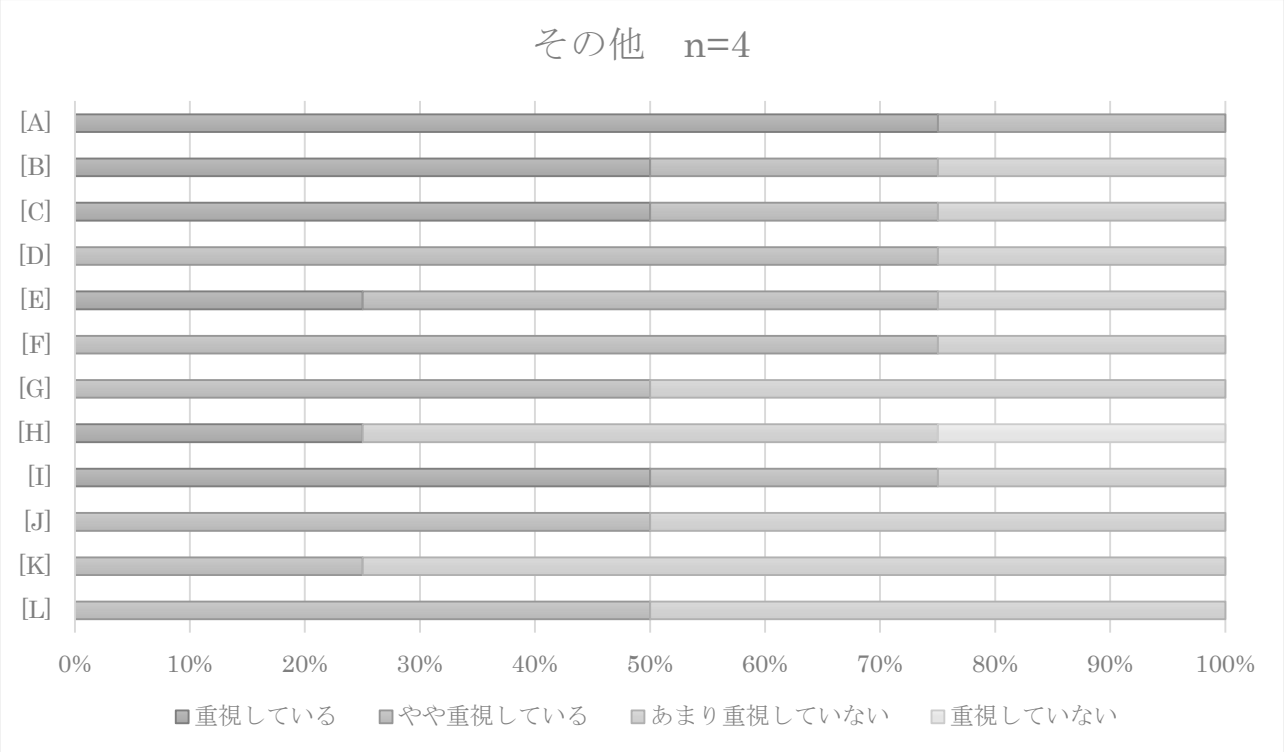
・プログラミング実習における学生の成果物に対する評価で重視している点

		重視している	やや重視している	あまり重視していない	重視していない
[A]	プログラム実行時に正しい結果が得られているか	100.0%	0.0%	0.0%	0.0%
[B]	ソースコードの可読性や保守性が高いか	33.3%	33.3%	33.3%	0.0%
[C]	アルゴリズムの内容を正確に記述できているか	33.3%	33.3%	0.0%	33.3%
[D]	バグになりそうなコードがないか	0.0%	33.3%	66.7%	0.0%
[E]	アルゴリズムの使い分けは適切か	0.0%	0.0%	100.0%	0.0%

[F]	記法や用語が一貫しているか	33.3%	66.7%	0.0%	0.0%
[G]	拡張性が高いか（オブジェクト指向に沿ったプログラムか)	0.0%	33.3%	66.7%	0.0%
[H]	テスト品質は適切か	0.0%	33.3%	66.7%	0.0%
[I]	コンパイルエラーにならないが、不適切なアクセス修飾子を使用していないか	0.0%	33.3%	66.7%	0.0%
[J]	リソースを無駄に消費していないか	33.3%	0.0%	66.7%	0.0%
[K]	抽象クラスや抽象メソッドが適切に利用されているか	0.0%	33.3%	66.7%	0.0%
[L]	多態性が適切に利用されているか	0.0%	66.7%	33.3%	0.0%

以下は、その他（2 年制、3 年制、4 年制以外）のみを抽出した結果を示しているが、回答件数が少ないため、傾向が見えにくい結果になっている。

・プログラミング実習における学生の成果物に対する評価で重視している点



・プログラミング実習における学生の成果物に対する評価で重視して点

		重視している	やや重視している	あまり重視していない	重視していない
[A]	プログラム実行時に正しい結果が得られているか	75.0%	25.0%	0.0%	0.0%

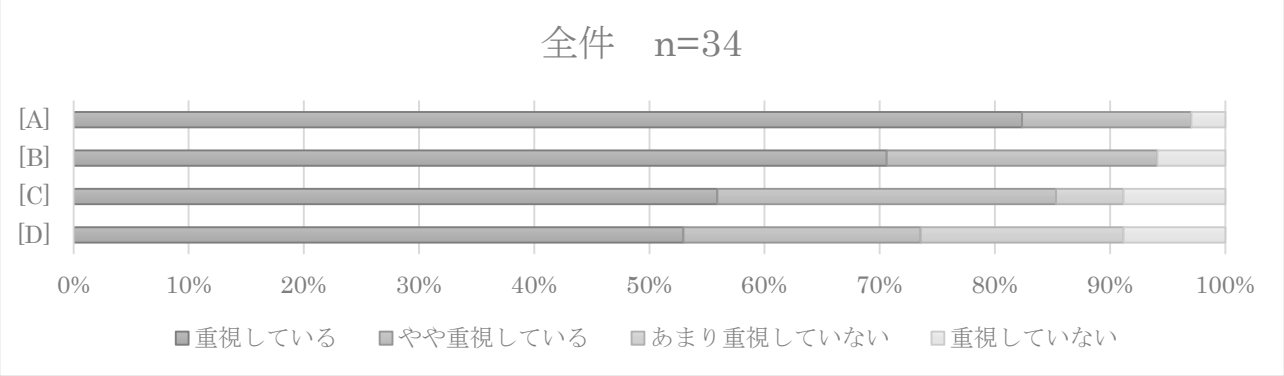
[B]	ソースコードの可読性や保守性が高いか	50.0%	25.0%	25.0%	0.0%
[C]	アルゴリズムの内容を正確に記述できているか	50.0%	25.0%	25.0%	0.0%
[D]	バグになりそうなコードがないか	0.0%	75.0%	25.0%	0.0%
[E]	アルゴリズムの使い分けは適切か	25.0%	50.0%	25.0%	0.0%
[F]	記法や用語が一貫しているか	0.0%	75.0%	25.0%	0.0%
[G]	拡張性が高いか（オブジェクト指向に沿ったプログラムか)	0.0%	50.0%	50.0%	0.0%
[H]	テスト品質は適切か	25.0%	0.0%	50.0%	25.0%
[I]	コンパイルエラーにならないが、不適切なアクセス修飾子を使用していないか	50.0%	25.0%	25.0%	0.0%
[J]	リソースを無駄に消費していないか	0.0%	50.0%	50.0%	0.0%
[K]	抽象クラスや抽象メソッドが適切に利用されているか	0.0%	25.0%	75.0%	0.0%
[L]	多態性が適切に利用されているか	0.0%	50.0%	50.0%	0.0%

質問１４－２

質問 14－2 では、以下の 4 項目で「プログラミング実習における学生の成果物に対する評価で、プログラミング以外で重視して評価している点」について確認している。

プログラミング以外で「出席率」と「授業態度」に対する指導において、「重視している」と「やや重視している」の回答率の合計が 90%を超えており、「チーム内のコミュニケーション」に対する指導においては、「重視している」と「やや重視している」の回答率の合計が 80%を超えており、「チーム内のマネジメント」に対する指導においても、「重視している」と「やや重視している」の回答率の合計が 70%を超えていることから、質問 11－2 と同様に、評価においてもヒューマンリテラシー面が強く重視されていることが確認できる。

・プログラミング実習における学生の成果物に対する評価で、プログラミング以外で重視して評価している点

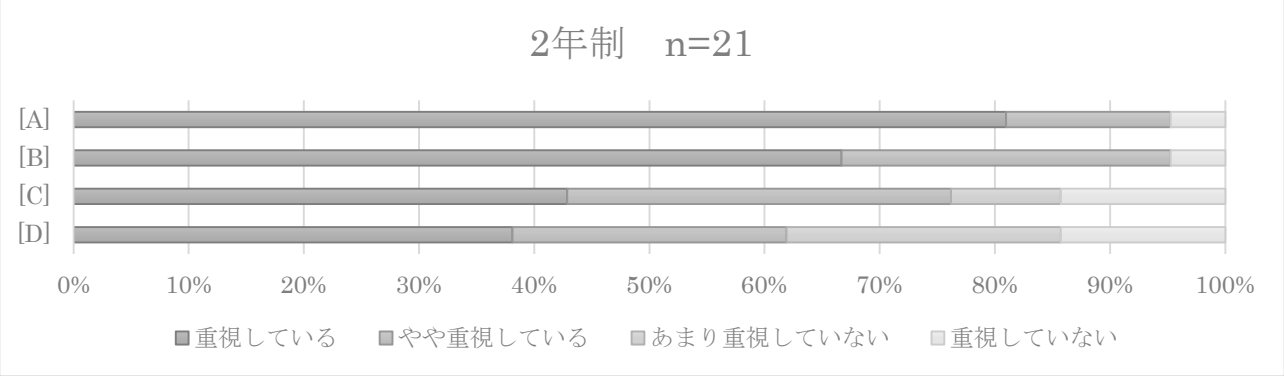


・プログラミング実習における学生の成果物に対する評価で、プログラミング以外で重視して評価している点

		重視している	やや重視している	あまり重視していない	重視していない
[A]	出席率が高いかどうか	82.4%	14.7%	0.0%	2.9%
[B]	真摯な授業態度かどうか	70.6%	23.5%	0.0%	5.9%
[C]	（作業単位が複数の場合） チーム内のコミュニケーションが良好か	55.9%	29.4%	5.9%	8.8%
[D]	（作業単位が複数の場合） チーム内のマネジメントが良好か	52.9%	20.6%	17.6%	8.8%

以下は、2 年制学科のみを抽出した結果を示しているが、全体とほぼ同じ結果にはなっていることが確認できる。

・プログラミング実習における学生の成果物に対する評価で、プログラミング以外で重視して評価している点

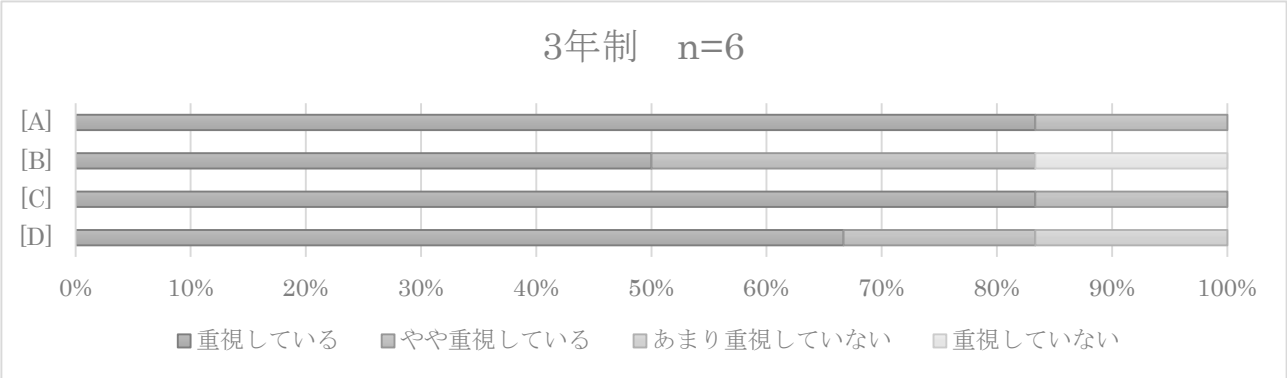


・プログラミング実習における学生の成果物に対する評価で、プログラミング以外で重視して評価している点

		重視している	やや重視している	あまり重視していない	重視していない
[A]	出席率が高いかどうか	81.0%	14.3%	0.0%	4.8%
[B]	真摯な授業態度かどうか	66.7%	28.6%	0.0%	4.8%
[C]	（作業単位が複数の場合） チーム内のコミュニケーションが良好か	42.9%	33.3%	9.5%	14.3%
[D]	（作業単位が複数の場合） チーム内のマネジメントが良好か	38.1%	23.8%	23.8%	14.3%

以下は、3 年制学科のみを抽出した結果を示しているが、回答件数が少ないため、傾向が見えにくい結果になっている。

・プログラミング実習における学生の成果物に対する評価で、プログラミング以外で重視して評価している点

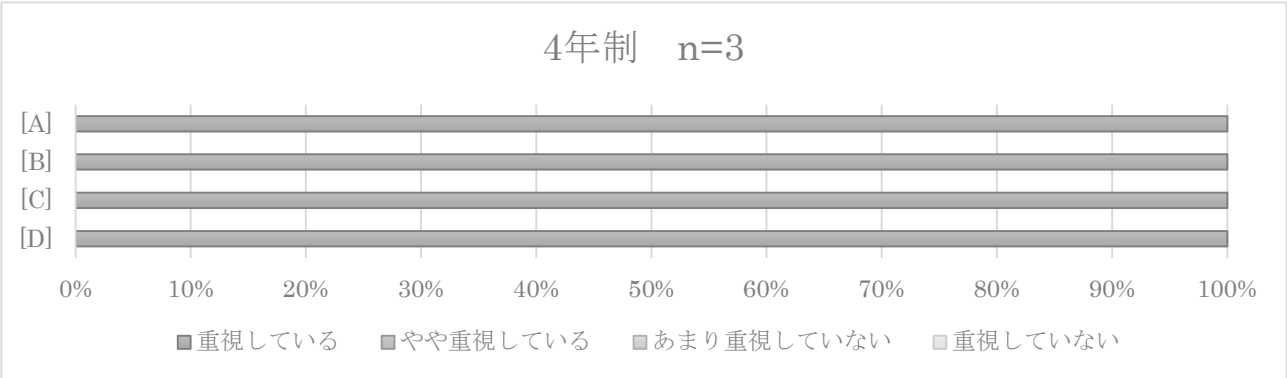


・プログラミング実習における学生の成果物に対する評価で、プログラミング以外で重視して評価している点

		重視している	やや重視している	あまり重視していない	重視していない
[A]	出席率が高いかどうか	83.3%	16.7%	0.0%	0.0%
[B]	真摯な授業態度かどうか	50.0%	33.3%	0.0%	16.7%
[C]	（作業単位が複数の場合） チーム内のコミュニケーションが良好か	83.3%	16.7%	0.0%	0.0%
[D]	（作業単位が複数の場合） チーム内のマネジメントが良好か	66.7%	16.7%	16.7%	0.0%

以下は、4 年制学科のみを抽出した結果を示しているが、回答件数が少ないため、傾向が見えにくい結果になっている。

・プログラミング実習における学生の成果物に対する評価で、プログラミング以外で重視して評価している点

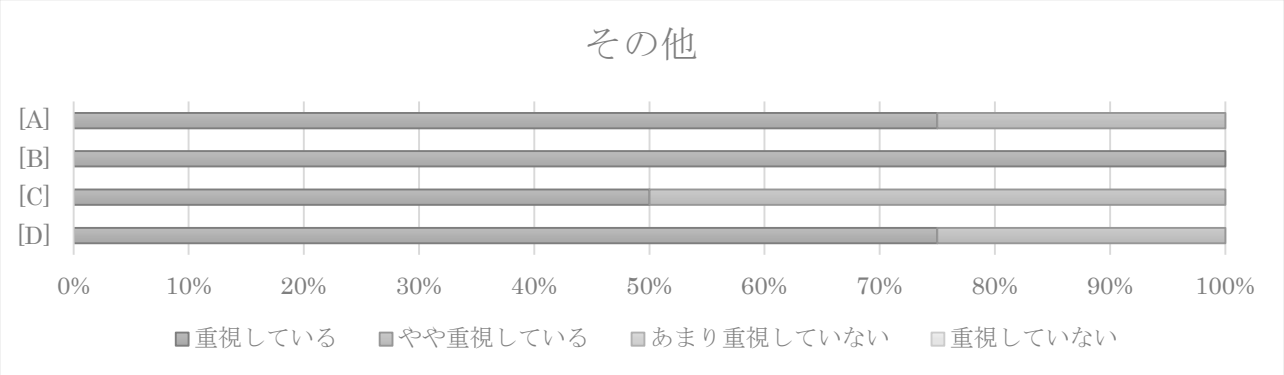


・プログラミング実習における学生の成果物に対する評価で、プログラミング以外で重視して評価している点

		重視している	やや重視している	あまり重視していない	重視していない
[A]	出席率が高いかどうか	100.0%	0.0%	0.0%	0.0%
[B]	真摯な授業態度かどうか	100.0%	0.0%	0.0%	0.0%
[C]	（作業単位が複数の場合） チーム内のコミュニケーションが良好か	100.0%	0.0%	0.0%	0.0%
[D]	（作業単位が複数の場合） チーム内のマネジメントが良好か	100.0%	0.0%	0.0%	0.0%

以下は、その他（2 年制、3 年制、4 年制以外）のみを抽出した結果を示しているが、回答件数が少ないため、傾向が見えにくい結果になっている。

・プログラミング実習における学生の成果物に対する評価で、プログラミング以外で重視して評価している点



・プログラミング実習における学生の成果物に対する評価で、プログラミング以外で重視して評価している点

		重視している	やや重視している	あまり重視していない	重視していない
[A]	出席率が高いかどうか	75.0%	25.0%	0.0%	0.0%
[B]	真摯な授業態度かどうか	100.0%	0.0%	0.0%	0.0%
[C]	（作業単位が複数の場合） チーム内のコミュニケーションが良好か	50.0%	50.0%	0.0%	0.0%
[D]	（作業単位が複数の場合） チーム内のマネジメントが良好か	75.0%	25.0%	0.0%	0.0%

質問１４－３

質問 14−3 では、「プログラミング実習における学生の成果物に対して、質問 14−1、質問 14−2 以外の項目で重視している点」を記述形式で確認している。

・プログラミング実習における学生の成果物に対して、質問 14−1、質問 14−2 以外の項目で重視している点

2 年制	提出物の期限が守られているか。
2 年制	コメントアウト
2 年制	ソースレビューにおけるコミュニケーション能力
2 年制	動く事かどうかを重視しており、また、口頭試問においてどのような考え方で制作したのか、結果をイメージしてからの制作なのか、考えをしっかりと持って取り組んでいるのかなど制作前段階を重視している
2 年制	要件定義から内部設計書の記述内容との整合性
3 年制	課題の提出回数、タイミング

質問１５

質問 15 では、「学生が作成したプログラムの評価を行う際に評価が難しいと考える点」を自由記述方式で確認している。

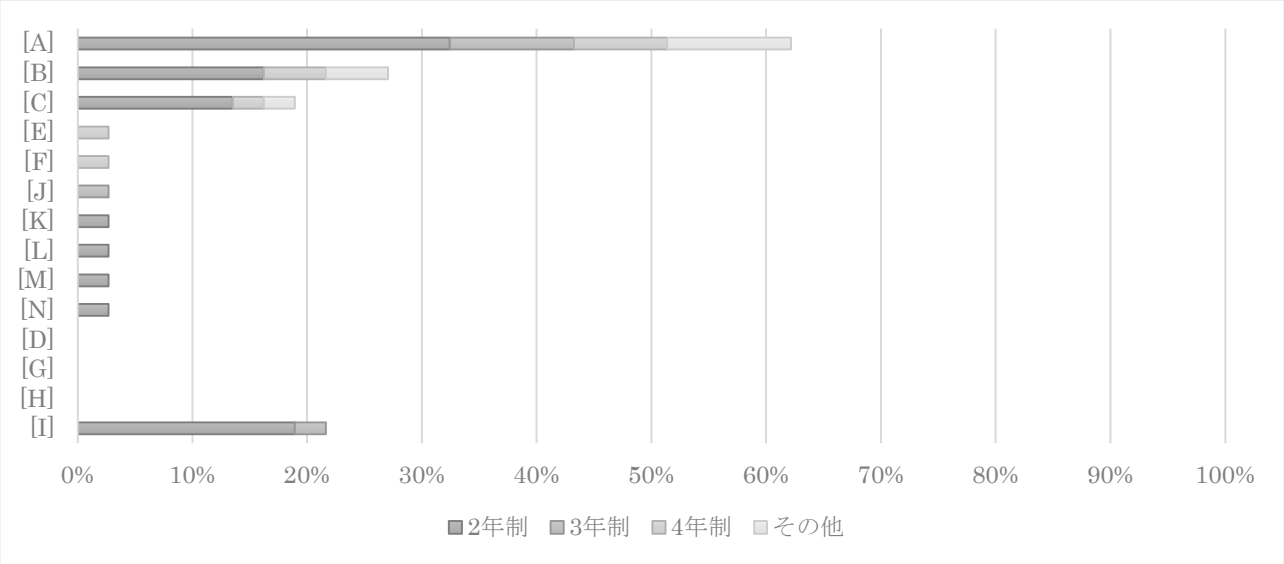
2 年制	ブラックボックスで評価をするかホワイトボックスで評価するかで評価付けが変わってくる。
2 年制	ある程度の長さのプログラムでは、全員分見るのに時間がかかる。
2 年制	一人一人のソースコードの確認、レビューにかなりの時間を要してしまう
2 年制	授業内容以外の技術を自主的に勉強して、プログラムに取り込んでいる場合、評価が難しい。
2 年制	コードレビューと考えを確認しているが、近年ではサンプルソースの乱立やアセットなどがあり、自分の考えなのか引用なのかが難しい
2 年制	同じ課題であっても、学生ごとにプログラムは異なるため、評価基準の一貫性の維持
2 年制	評価対象がチームでの成果発表会となるため、個人評価が定期試験の点数のみとなってしまうこと。
2 年制	評価対象のプログラムの数の多さ
2 年制	コンパイルエラーがあるコードが提出される場合があること
2 年制	チーム開発のプログラミングでは、メンバー同士の相談を行うため、自分の力で作成したか他人の力を大きく借りたかが、判断しづらい
2 年制	確認ポイントが多く、結果、教員の作業量が大きくなる。
3 年制	コード規約などの標準化をしていないため、多種多様な書き方が存在し、内容までの確認が出来ない。また、ソースを確認する時間が無い。
4 年制	本当に本人の力となっていて、授業目標を満たした実力を身に付けたかどうか判断することが難しい。特にグループワークは教員側マネジメントの問題もあり、作業負担が一部の学生へ偏ってしまう傾向にある。個人を本当に正しく評価できているか疑問である。
4 年制	内部のすべてを追えない
その他	学生たちが完成することで精いっぱいになっているので、その先の要求が難しい
その他	正解は一つではないところ

質問１６

質問16では、「プログラミングスキルの達成度をはかるために、学生にどのような資格を取得させ、受講学生の50%以上が取得できている資格」について複数回答可で確認している。

「基本情報技術者試験」が全体の 62.2%で最も高い回答率であるが、プログラミングスキルを直接的に判定できるベンダー資格は少数の回答率であった。

・プログラミングスキルの達成度をはかるために、学生にどのような資格を取得させ、受講学生の50%以上が取得できている資格



・プログラミングスキルの達成度をはかるために、学生にどのような資格を取得させ、受講学生の50%以上が取得できている資格

		全体 (37 件)	2 年制 (24 件)	3 年制 (6 件)	4 年制 (3 件)	その他 (4 件)
[A]	基本情報技術者試験	62.2%	50.0%	66.7%	100.0%	100.0%
[B]	Oracle Certified Java Programmer, Bronze SE 認定資格	27.0%	25.0%	0.0%	66.7%	50.0%
[C]	Oracle Certified Java Programmer, Silver SE 11 認定資格	18.9%	20.8%	0.0%	33.3%	25.0%
[D]	Python3 エンジニア認定基礎試験	0.0%	0.0%	0.0%	0.0%	0.0%
[E]	Python3 エンジニア認定データ分析試験	2.7%	0.0%	0.0%	33.3%	0.0%
[F]	サーティファイ ＪＡＶＡ検定	2.7%	0.0%	0.0%	33.3%	0.0%
[G]	サーティファイ C 言語プログラミング能力検定２級	0.0%	0.0%	0.0%	0.0%	0.0%
[H]	情報検定情報システム試験プログラミングスキル	0.0%	0.0%	0.0%	0.0%	0.0%
[I]	情報処理能力認定試験 3 級	21.6%	29.2%	16.7%	0.0%	0.0%
[J]	実践プログラミング技術者試験	2.7%	0.0%	16.7%	0.0%	0.0%
[K]	Android 技術者認定試験	2.7%	4.2%	0.0%	0.0%	0.0%
[L]	PHP 技術者認定試験 初級試験	2.7%	4.2%	0.0%	0.0%	0.0%

質問１７

質問 17 では、「遠隔授業時のプログラミング実習に関する課題や問題」について自由記述方式で確認している。

2 年制	学生の声にならない声を聞く体制が必要だが、不十分である。
2 年制	学生の家庭により通信環境・ハードウェア環境が異なるので、作業効率の面で差が出やすい。
2 年制	課題を行わせているときに各学生の進捗がどうなっているか？
2 年制	学生一人一人の状況が把握しづらい。
2 年制	学生の取り組み状況が見えづらい
2 年制	プログラミング実習に本格的に遠隔授業を取り入れていないが、数日～１か月程度やってみたが、授業資料の作成など準備が間に合っていなかった。
2 年制	モチベーションによってやらない事があるが、それはエンジニアとしての資質を見る機会と考えている。しかし、通常の学校評価ではなかなか厳しい評価になるため難しい状況である。
2 年制	授業時間内での学生ソース確認の円滑さ
2 年制	質問をしてくれる学生の個別指導は充実するが、声を挙げない学生へのケアは全くできなくなるので理解度の格差が大きくなってしまう。
2 年制	実施していないので無し。
2 年制	対面より遠隔の方が画面共有がやりやすく、課題ではなく利点になります。
2 年制	自宅に PC を持たない、または PC の性能が低い学生が数名いる場合、一律で開発環境を作り、指導することが難しい
2 年制	学生ごとの個別指導に手間がかかる。特に、授業時間内でのリアルタイムな対応は難しい。
3 年制	遠隔授業においてプログラミング実習を行っていない為、無回答
3 年制	グループ作業を行う場合は、オンライン発信用途は別の環境がなければ、各グループ間での意思疎通が図りづらい
3 年制	学生の能力差が激しく、ライブ配信での授業はさらにやりにくい。
4 年制	学生が真摯に取り組んでいるか把握することが難しい。また、エラーの指摘について口頭だけでは難しく、最終的に学生が自身の誤りに気付く機会が減っているように感じる。
4 年制	進捗状況の確認や作業状態の確認が難しい
その他	PC スキルが低いが置いてけぼりになってしまう

2.2.3. IT 系企業向け調査アンケート結果

IT 系企業に実施した調査アンケートの結果を以下にまとめる。

質問 1

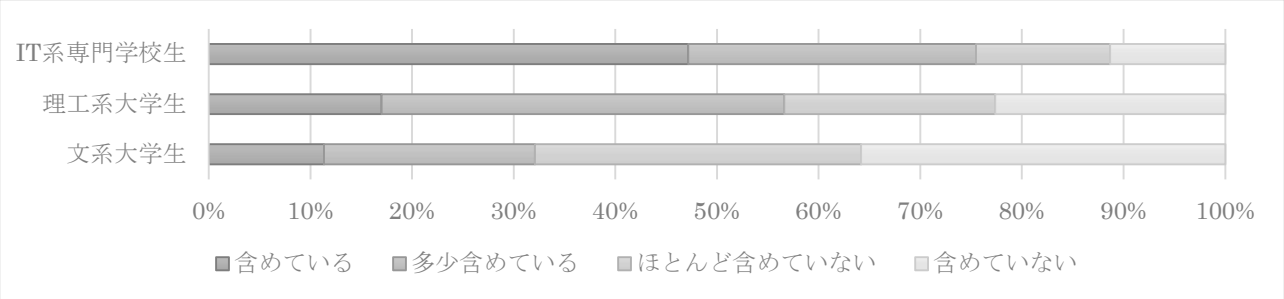
質問 1 では、「新卒採用の選考基準にプログラミングスキルを含めているか」について、選考対象を「IT 系専門学校生」「理工系大学生」「文系大学生」に分けて確認している。

選考基準にプログラミングスキルを「含めている」と「多少含めている」を合計した回答率を見ると、IT 系専門学校生が 75.5%、理工系大学生が 56.6%、文系大学生が 32.1%と差がみられる。

また、「含めている」のみ回答率を見ても、IT 系専門学校生が 47.2%、理工系大学生が 17.0%、文系大学生が 11.3%と同じ順列での差がみられる。

これらのことから、IT 系企業は選考対象によって選考基準を変化させており、中でも IT 系専門学校生に対してプログラミングスキルを選考基準に含めている IT 系企業が多いことが確認できる。

・新卒採用におけるプログラミングスキルの選考



・新卒採用におけるプログラミングスキルの選考

	含めている	多少含めている	ほとんど 含めていない	含めていない
IT 系専門学校生	47.2%	28.3%	13.2%	11.3%
理工系大学生	17.0%	39.6%	20.8%	22.6%
文系大学生	11.3%	20.8%	32.1%	35.8%

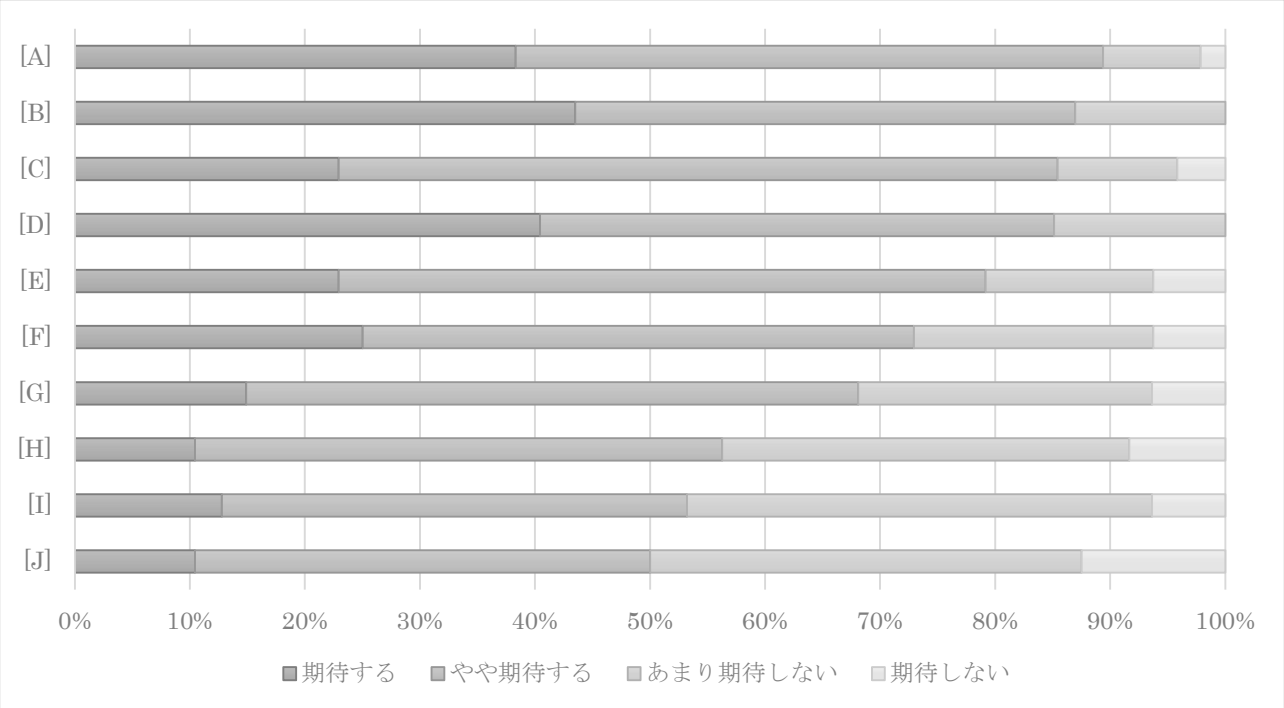
※以下の質問では、IT 系専門学校生に対する選考基準にプログラミングスキルを含めていないと回答した 6 件を省いた 46 件の回答で集計している

質問 2

質問 2 では、「IT 系専門学校のプログラミング教育に対するスキル習得への期待」について確認している。

全ての項目において「期待する」と「やや期待する」の合計が過半数を超えているが、「自らのアルゴリズムを考えることができる」「自身が構築したプログラムのエラーについて、自己解決できる」「他者が記述したプログラムコードを正しく読み解くことができる」「正しいアルゴリズムでプログラムを構築することができる」の 4 項目が「期待する」と「やや期待する」の合計が 80%を超えており、「プログラムを構築できる基礎的な能力」に期待していることが確認できる。

・IT 系専門学校のプログラミング教育に対するスキル習得への期待



・IT 系専門学校のプログラミング教育に対するスキル習得への期待

		期待する	やや 期待する	あまり 期待しない	期待しない
[A]	自らのアルゴリズムを考えることができる	38.3%	51.1%	8.5%	2.1%
[B]	自身が構築したプログラムのエラーについて、自己解決できる	43.5%	43.5%	13.0%	0.0%
[C]	他者が記述したプログラムコードを正しく読み解くことができる	22.9%	62.5%	10.4%	4.2%
[D]	正しいアルゴリズムでプログラムを構築することができる	40.4%	44.7%	14.9%	0.0%
[E]	一つのプログラム課題に対して、多数の解法を考案することができる	22.9%	56.3%	14.6%	6.3%
[F]	可読性の高いプログラムコードを記述	25.0%	47.9%	20.8%	6.3%

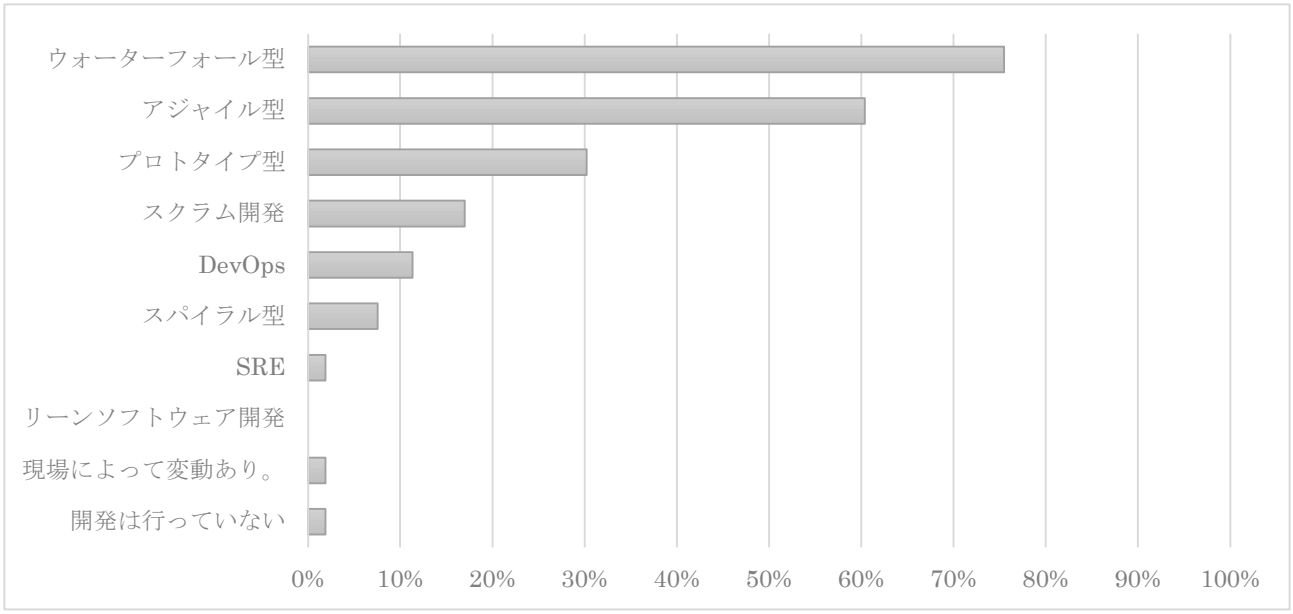
	することができる				
[G]	実行処理速度も考慮したプログラムを構築することができる	14.9%	53.2%	25.5%	6.4%
[H]	UML 図（クラス図、ER 図、シーケンス図等）の読み書きができる	10.4%	45.8%	35.4%	8.3%
[I]	セキュリティを考慮したプログラムを構築することができる	12.8%	40.4%	40.4%	6.4%
[J]	複数のプログラミング言語を活用できる	10.4%	39.6%	37.5%	12.5%

質問３

質問 3 では、「自社のシステム・ソフトウェア開発で使用している主な開発モデル」を複数選択可で確認している。

「ウォーターフォール型」の回答が 75.5%で最も多く、次いで「アジャイル型」の回答が 60.4%で、この２つの開発モデルが過半数の回答であり、一般的（ソフトウェア工学的には古くから使われている）な「ウォーターフォール型」が最も多く、近年人気の高い「アジャイル型」が主流になりつつあることが確認できる。

・自社のシステム・ソフトウェア開発で使用している主な開発モデル



・自社のシステム・ソフトウェア開発で使用している主な開発モデル

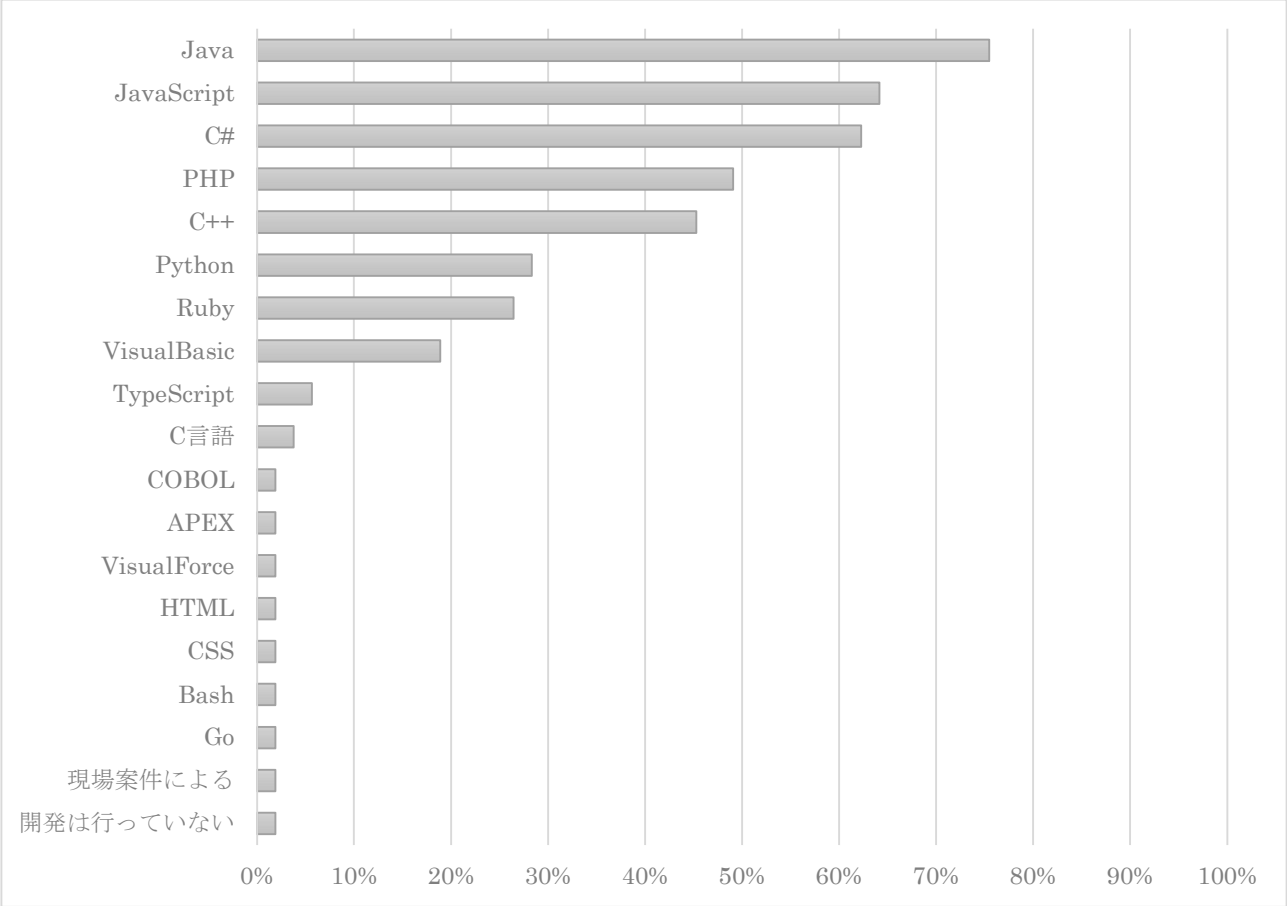
	回答率		回答率
ウォーターフォール型	75.5%	スパイラル型	7.5%
アジャイル型	60.4%	SRE	1.9%
プロトタイプ型	30.2%	リーンソフトウェア開発	0.0%
スクラム開発	17.0%	現場によって変動あり。	1.9%
DevOps	11.3%	開発は行っていない	1.9%

質問４

質問 4 では、「自社のシステム・ソフトウェア開発で使用している主なプログラム言語」を複数選択可で確認している。

75.5%の「Java」が最も多く使われており、次いで、64.2%の「JavaScript」と 62.3%の「C#」が過半数以上の回答で多く使われていることが確認でき、続く、49.1%の「PHP」と 45.3%の「C++」の計 5 つの言語に回答が集まっていることが確認できる。

・自社のシステム・ソフトウェア開発で使用している主なプログラム言語



・自社のシステム・ソフトウェア開発で使用している主なプログラム言語

	回答率		回答率
Java	75.5%	COBOL	1.9%
JavaScript	64.2%	APEX	1.9%
C#	62.3%	VisualForce	1.9%
PHP	49.1%	HTML	1.9%
C++	45.3%	CSS	1.9%
Python	28.3%	Bash	1.9%
Ruby	26.4%	Go	1.9%
VisualBasic	18.9%	現場案件による	1.9%
TypeScript	5.7%	開発は行っていない	1.9%
C 言語	3.8%		

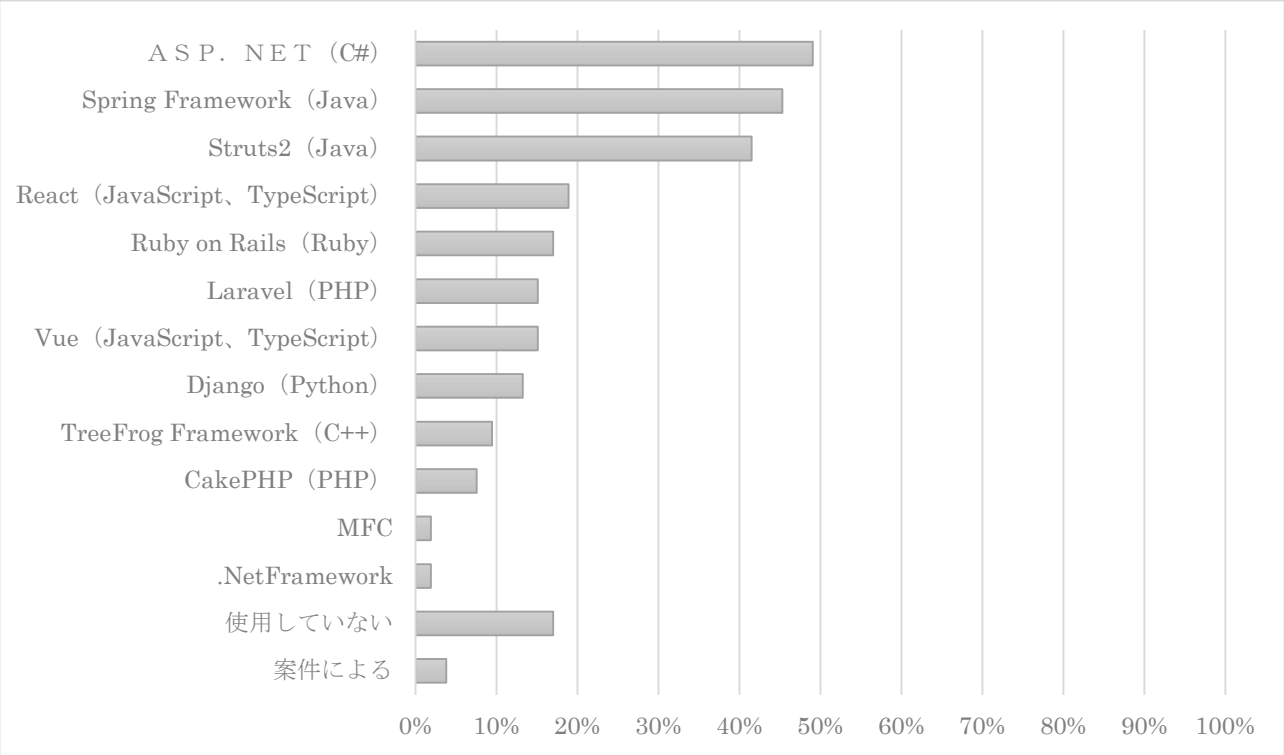
質問５

質問 5 では、「自社のシステム・ソフトウェア開発で使用している主なフレームワーク」を複数選択可で確認している。

「ＡＳＰ．ＮＥＴ（C#)」が 49.1%、「Spring Framework（Java)」が 45.3%、「Struts2（Java)」が 41.5%で、この 3 つのフレームワークに回答が集まっていることが確認でき、質問 4 の回答にある使用言語に関係するフレームワークが多く使われていることが確認できる。

一方で、フレームワークを「使用していない」の回答は 17.0%で、フレームワークを使用した開発が主流になっていることが確認できる。

・自社のシステム・ソフトウェア開発で使用している主なフレームワーク



・自社のシステム・ソフトウェア開発で使用している主なフレームワーク

	回答率		回答率
ＡＳＰ．ＮＥＴ（C#)	49.1%	TreeFrog Framework（C++）	9.4%
Spring Framework（Java）	45.3%	CakePHP（PHP）	7.5%
Struts2（Java）	41.5%	MFC	1.9%
React（JavaScript、TypeScript）	18.9%	.NetFramework	1.9%
Ruby on Rails（Ruby）	17.0%	使用していない	17.0%
Laravel（PHP）	15.1%	無回答	9.4%
Vue（JavaScript、TypeScript）	15.1%	案件による	3.8%
Django（Python）	13.2%		

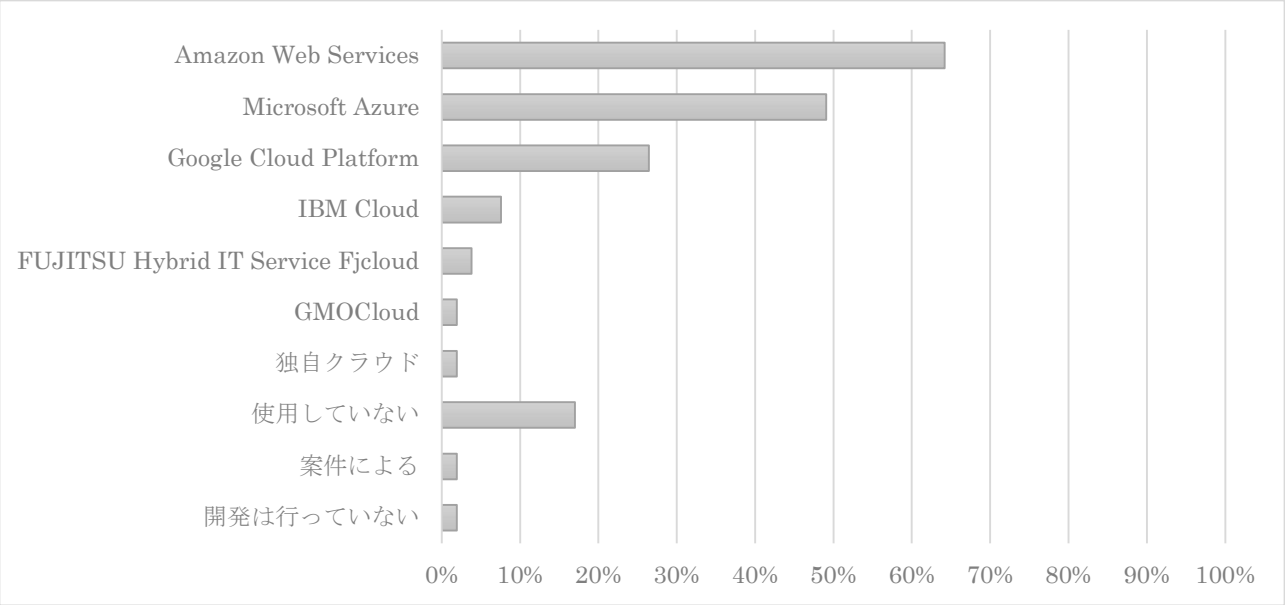
質問６

質問 6 では、「自社のシステム・ソフトウェア開発で使用している主なパブリッククラウドサービス」を複数選択可で確認している。

「Amazon Web Services」が 64.2%、「Microsoft Azure」が 49.1%で、２つのパブリッククラウドサービスに回答が集まっており、質問４の使用言語と質問５の使用しているフレームワークの回答に関係するパブリッククラウドサービスが利用されていることが確認できる。

また、パブリッククラウドサービスを「使用していない」の回答が 17.0%で、フレームワークと同様にパブリッククラウドサービスの利用が開発の主流になっていることが確認できる。

・自社のシステム・ソフトウェア開発で使用している主なパブリッククラウドサービス



・自社のシステム・ソフトウェア開発で使用している主なパブリッククラウドサービス

	回答率		回答率
Amazon Web Services	64.2%	GMOCloud	1.9%
Microsoft Azure	49.1%	独自クラウド	1.9%
Google Cloud Platform	26.4%	使用していない	17.0%
IBM Cloud	7.5%	案件による	1.9%
FUJITSU Hybrid IT Service Fjcloud	3.8%	開発は行っていない	1.9%

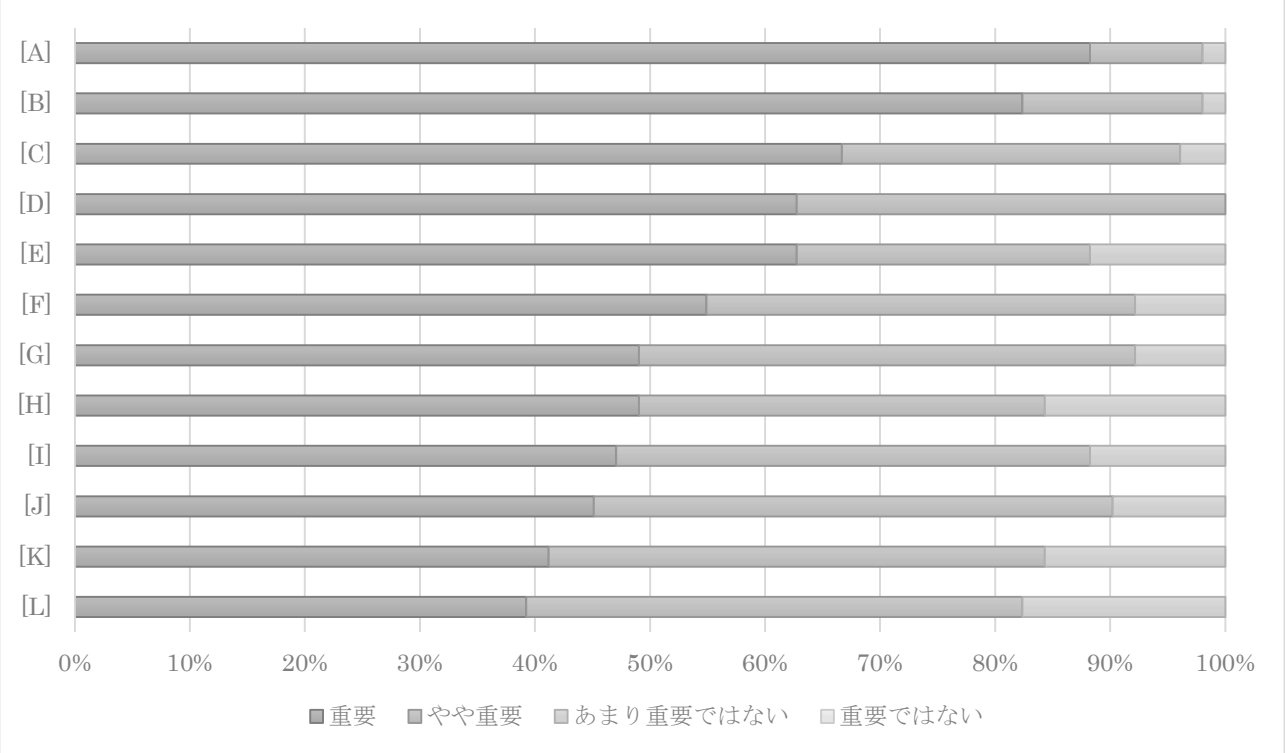
質問 7

質問 7 では、「自社のソフトウェア製品における、製造過程で留意するプログラム品質」について確認している。

全確認事項に対して全体の 80%以上が「重要」または「やや重要」と回答していることから、全体的にプログラム品質に対して重要性を示していることが確認できる。

また、「重要」の回答だけに着目すると、「正当性が高いこと」が 88.2%、「データ完全性が高いこと」が 82.4%で、この２つの確認事項についてプログラム品質の重要性が強く示されていることが確認でき、次いで、「バグが無いこと」が 66.7%、「ソースコードの可読性が高いなどの保守性に優れていること」が 62.7%、「テスト仕様書の項目数が揃っていること」が 62.7%、「プログラムの実行速度、アルゴリズムの精度」が 54.9%で、「重要」の回答が過半数を超えたこれら 6 つの確認事項についてプログラム品質の重要性が示されていることが確認できる。

・自社のソフトウェア製品における、製造過程で留意するプログラム品質



・自社のソフトウェア製品における、製造過程で留意するプログラム品質

		重要	やや重要	あまり重要ではない	重要ではない
[A]	正当性が高いこと	88.2%	9.8%	2.0%	0.0%
[B]	データ完全性が高いこと	82.4%	15.7%	2.0%	0.0%
[C]	バグが無いこと	66.7%	29.4%	3.9%	0.0%
[D]	ソースコードの可読性が高いなどの保守性に優れていること	62.7%	37.3%	0.0%	0.0%

[E]	テスト仕様書の項目数が揃っていること	62.7%	25.5%	11.8%	0.0%
[F]	プログラムの実行速度、アルゴリズムの精度	54.9%	37.3%	7.8%	0.0%
[G]	記法や用語が一貫していること	49.0%	43.1%	7.8%	0.0%
[H]	仕様書、設計書等の必要なドキュメントが網羅されていること	49.0%	35.3%	15.7%	0.0%
[I]	仕様書、設計書等のドキュメントの精度が高いこと	47.1%	41.2%	11.8%	0.0%
[J]	拡張性が高いこと	45.1%	45.1%	9.8%	0.0%
[K]	リソースを無駄に消費しないプログラムであること	41.2%	43.1%	15.7%	0.0%
[L]	プログラムの持続性	39.2%	43.1%	17.6%	0.0%

[A] 入力条件を満たしたプログラムを実行すると、そのプログラムが確実に完了もしくは出力条件を満たした結果が得られる

[B] 必要なデータが全て揃っていて欠損や不整合がないことを保証する

[D] バグなどの不具合要因の修正や、性能や使い易さといった特性を改善でき、製作当初に想定していなかった機能の追加や変更を少ない手間やコストで実施・改変が行える

[J] 付加的な機能を追加し、それらの性能をあとから向上させる事が可能である

[L] ほかの環境や将来の環境における互換性

質問８

質問 8 では、「今後採用したいと思う新卒のエンジニア人材に求めたいその他の知識・スキル」について自由記述で確認している。

全体的に「ヒューマンリテラシー」や「社会人基礎力」などに該当する意見が多いが、IT 分野の知識・スキルでは「データベース」「SQL」「設計書」「アルゴリズム」「AWS、GCP を始めとした Saas」「インフラ領域」「IT パスポート」などの単語が意見の中に含まれている。

ヒューマンスキルが高く、会話のレスポンスが早いこと。
コミュニケーションスキル
やる気がある、自分の考えを伝えることができる（言葉や図表等問わず）、人の話を聞くことができる、必要なことを報告できる（報連相ともいわれるが・・・せめて報告だけでも・・・）、何が大事か考え理解できる
問題を発生前に察知し、検討する能力。
自覚的、自発的、自律的に行動できること。
人間力
行動力のある方、IT 業界で働く覚悟
社会人、企業人になるにあたっての心構えや要素 対人感受性やコミュニケーション能力
オーダーに沿ったシステムを構築(考えること)ができる人材。また、オーダーを正しく理解できる能力とそれを聞き出す能力に長けた人材。
発生した課題を解決するためにインターネットやコミュニティ、その他の情報ソースから的確に必要な情報を導き出すための知識・スキル
設計書を読み助力ありで正しくプログラミングできること。自分の考えをはっきり言えること。わからない事は分からないといえること。データベースの知識、SQL を使える。デザインセンスがある。
多少の幅広い知識は必要だか、その中で自分が得意とする事を1つ深く理解している事。複数人で開発する時にリーダー的存在になれる人。お客さん、また社内でも打合せ時には相手の言っている事を素早く正確に理解し、また自分の発言については分かりやすく伝える事。お客さん、また社内でも打合せ時には相手の言っている事を素早く正確に理解し、また自分の発言については分かりやすく伝える事。

当社の事業は保守・運用が中心となり、開発は保守開発となります。そのためプログラミングスキルは選考基準として求めてはおらず、ICT 技術に対する興味・関心を一番に求めております。
分からない時に間違ってもかまわないので、「まず自分で調べる」といったくせは重要と考えております。
データベースの概念を理解しており、自分で構築・活用がある程度できること。
技術レベルや知識よりも、物事を多角的に捉えることができ、柔軟な発想ができる人材
文系学生は Excel を使えないことも多いので、Excel スキルはあると良い。情報系の専門学校生は IT スキルがあっても他人に伝える力が不足していることがあるため、文章作成スキルがもう少しあると良いなと感じることがあります。
能動的に考え動ける人材
自分で学習する意欲、素直さ、アルゴリズム
新しい技術への適応力。クライアント会社の業務理解力・改善提案力
コミュニケーション能力。自己解決能力。一度決心したことをやり遂げることができる能力
知的好奇心が高く、自主的に新しい知識を取得していけるような人材。
・コミュニケーションスキル・自分でモチベーションを維持できる・集中力と散漫力のバランスが取れている
インフラ領域
コミュニケーション能力、向上心です。
読解力・探求心
失敗を乗り越えた経験や糧
コミュニケーション能力、報連相の基本
学校で学ぶ技術は相応でも良く、QCD 目標を達成するのに必要な社会人基礎力の下地がある方が望ましい
困ったときに周りに相談できる力。
コミュニケーション能力
AWS、GCP を始めとした Saas に関する知識
提案力、交渉力を身に着けるための土台となるコミュニケーション能力
一定のコミュニケーション能力があること、自己研鑽意欲があること
浅く広い、業務や業界のしくみに対する知識。
国家資格取得者、コンピュターの基礎知識
コミュニケーションスキル、自身を成長させるための意欲・モチベーション（働く事の意義の理解）
コミュニケーション力（相手の話を聞く、自身の考えを伝えられる）
知識については、入社時点では IPA の IT パスポートに合格できるレベルぐらいの人であれば十分です。社内外の人に対して思いやりがあるコミュニケーションが取れ、やる気があり、すぐに諦めず、落ち着いて行動できる、遅刻しない、挨拶ができる、会話や文章が論理的である、誤字脱字がない、忘れ物をしない…など知識以外のスキルの方がどちらかというと重要です。
・素直さ・業務に対する理解や意欲
自身で目標へ向かって努力できること
IT エンジニアとして、入社することをゴールではなく、スタートとして捉えて、学生時代以上に主体性を持って業務や自己研鑽に臨む意欲があること。
コミュニケーションスキル（自ら考え行動できる、周りと協調しながら行動できる　等）

2.2.4. 考察

今回のアンケートの結果をもとに、いくつかの結果から関係性を計ってみた。

IT 系専門学校向けの質問３と IT 系企業向けの質問４の関係性

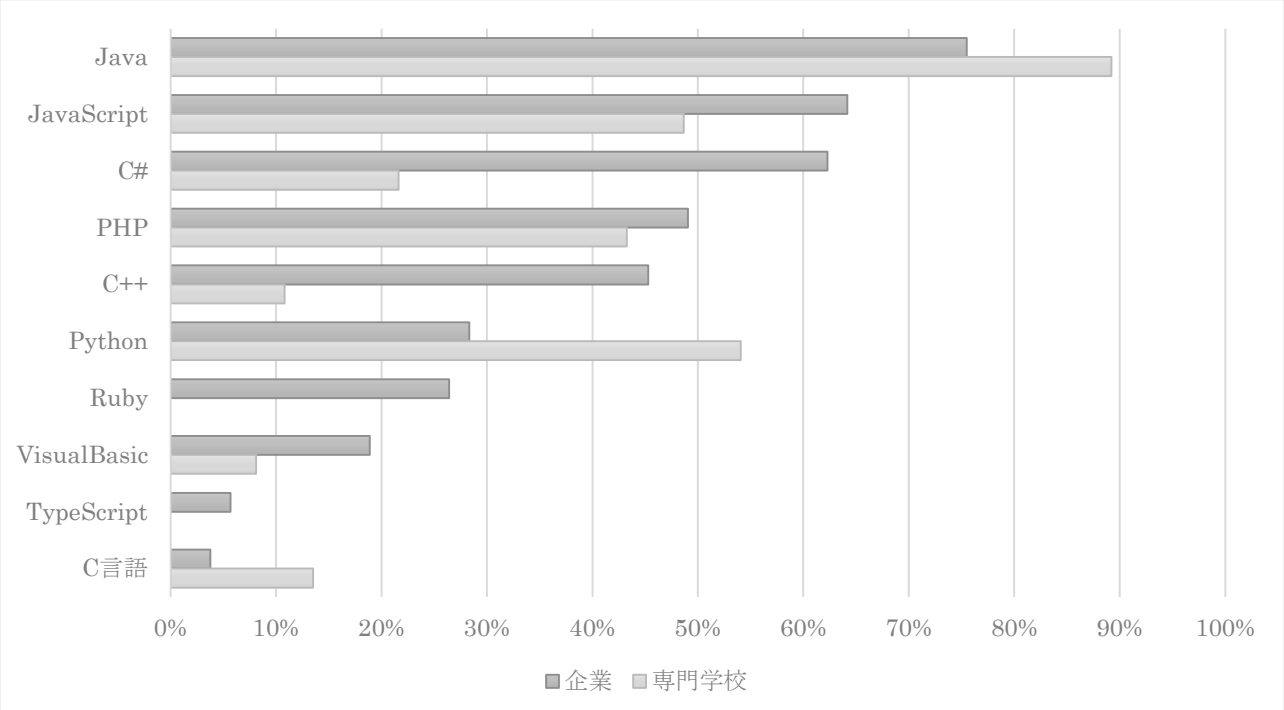
IT 系専門学校向けの質問 3 と IT 系企業向けの質問 4 では、学習・使用しているプログラミング言語を確認している。

これらの質問で比較できる言語の学習率・使用率をまとめてみたところ、「C#」「C++」「Python」「Ruby」に大きな回答差がみられた。

特に、「C#」「C++」「Ruby」には 25%以上の回答差がみられ、IT 系企業で使われている（求められている）言語を IT 系専門学校で教えられていないことが確認できる。

また、「Python」は近年人気の高まっている言語で学習している専門学校が多くある一方で、IT 企業ではそれほど使われていないことが確認できる。

	企業	専門学校	差
Java	75.5%	89.2%	-13.7%
JavaScript	64.2%	48.6%	15.5%
C#	62.3%	21.6%	**40.6%**
PHP	49.1%	43.2%	5.8%
C++	45.3%	10.8%	**34.5%**
Python	28.3%	54.1%	**-25.8%**
Ruby	26.4%	0.0%	**26.4%**
VisualBasic	18.9%	8.1%	10.8%
TypeScript	5.7%	0.0%	5.7%
C 言語	3.8%	13.5%	-9.7%



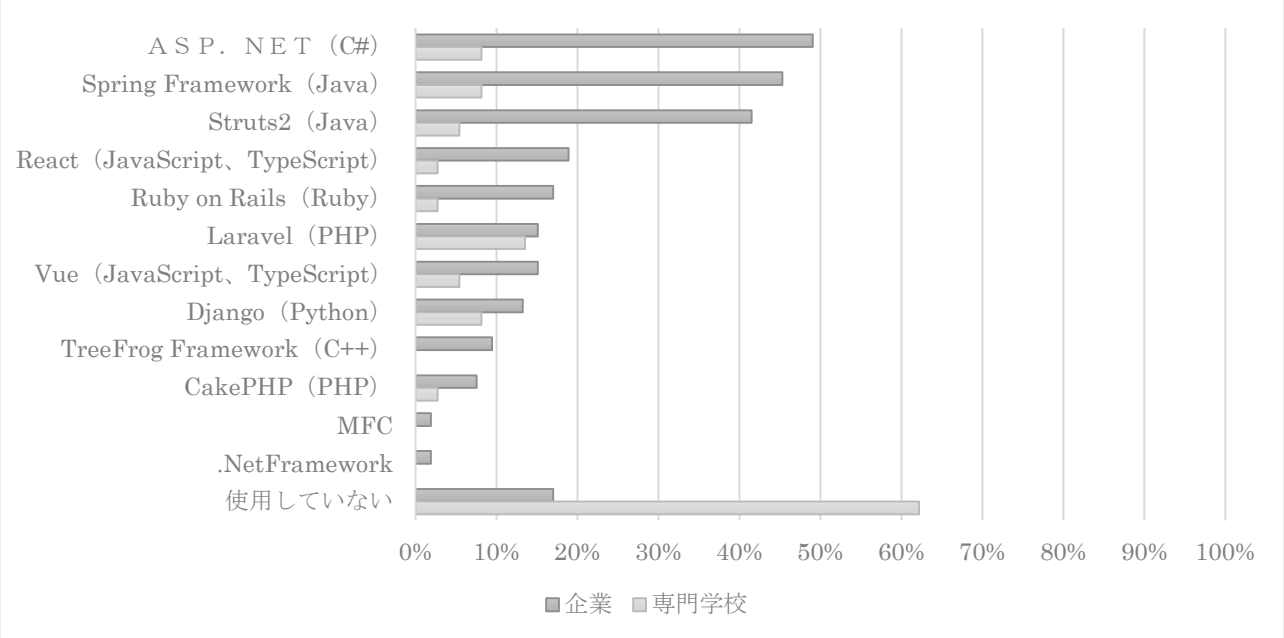
IT 系専門学校向けの質問４と IT 系企業向けの質問５の関係性

IT 系専門学校向けの質問 4 と IT 系企業向けの質問 5 では、学習・使用しているフレームワークを確認している。

これらの質問で比較できるフレームワークの学習率・使用率をまとめてみたところ、フレームワークを「使用していない」と回答した企業は 20%未満で、フレームワークの使用が開発の主流になっていることを示しているのに対して、フレームワークを「使用していない」と回答した専門学校は 60%以上もあり、IT 企業の開発の主流に合わせるのであれば、フレームワークを使用した学習を急ぎ取り入れる必要があることを示唆している。

また、フレームワークの使用は言語にも関係することや、IT 企業のフレームワークの使用状況を見ると、複数のフレームワークを使用し、複数の言語を学習することを考えなければならない状況にあると言える。

	企業	専門学校	差
ＡＳＰ．ＮＥＴ（C#）	49.1%	8.1%	**40.9%**
Spring Framework（Java）	45.3%	8.1%	**37.2%**
Struts2（Java）	41.5%	5.4%	**36.1%**
React（JavaScript、TypeScript）	18.9%	2.7%	16.2%
Ruby on Rails（Ruby）	17.0%	2.7%	14.3%
Laravel（PHP）	15.1%	13.5%	1.6%
Vue（JavaScript、TypeScript）	15.1%	5.4%	9.7%
Django（Python）	13.2%	8.1%	5.1%
TreeFrog Framework（C++）	9.4%	0.0%	9.4%
CakePHP（PHP）	7.5%	2.7%	4.8%
MFC	1.9%	0.0%	1.9%
.NetFramework	1.9%	0.0%	1.9%
使用していない	17.0%	62.2%	**-45.2%**

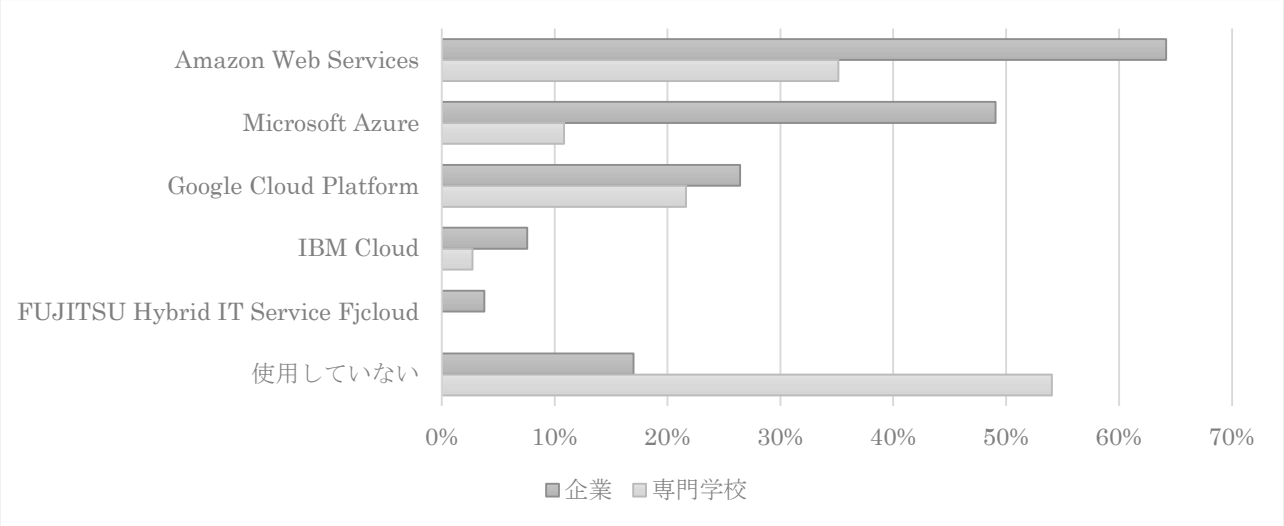


IT 系専門学校向けの質問５と IT 系企業向けの質問６の関係性

IT 系専門学校向けの質問 5 と IT 系企業向けの質問 6 では、学習・使用しているパブリッククラウドサービスを確認している。

これらの質問で比較できるパブリッククラウドサービスの学習率・使用率をまとめてみたところ、パブリッククラウドサービスを「使用していない」と回答した企業は 20%未満に対して、専門学校では 50%以上が「使用していない」と回答している。先のフレームワークの比較と同じく、IT 系企業ではパブリッククラウドサービスの使用が主流になっている中で、IT 系専門学校もパブリッククラウドサービスを使用した学習を急ぎ取り入れる必要があると言える。

	企業	専門学校	差
Amazon Web Services	64.2%	35.1%	**29.0%**
Microsoft Azure	49.1%	10.8%	**38.2%**
Google Cloud Platform	26.4%	21.6%	4.8%
IBM Cloud	7.5%	2.7%	4.8%
FUJITSU Hybrid IT Service Fjcloud	3.8%	0.0%	3.8%
使用していない	17.0%	54.1%	**-37.1%**



2.2.5. まとめ

IT 系専門学校向けアンケートから

IT 系専門学校向けアンケートから、約 70%の IT 系専門学校で遠隔授業によるプログラミング実習を実施したことが確認でき、それらの学校では、「学生のつまずいている個所が把握しにくい」「プログラミングにつまずいている学生の個別フォローを行いにくい」という状況が確認された。

また、「指導担当の教員による指導の個人差がある」との回答が 80%以上あり、「プログラム品質に関する指導は限定的になっている」ことや「評価の時間が不足していることから、十分な評価ができていない」ことが確認できた。

以上のことから、遠隔授業下において、学生が作成するプログラムを個別に確認することができ、適切なフォロー（特に個別のフォロー）ができるシステムが必要であり、特にプログラム品質に関する教材と、プログラム品質を考慮して作成したプログラム課題の成果物に対する評価が自動で行えるようなシステムが必要である。

IT 系企業向けアンケートから

IT 系企業向けアンケートからは、プログラム品質において「正当性が高いこと」「データ完全性が高いこと」「バグが無いこと」「ソースコードの可読性が高いなどの保守性に優れていること」「テスト仕様書の項目数が揃っていること」「プログラムの実行速度、アルゴリズムの精度」などに対する高い重要性が確認でき、専門学校のプログラミング教育においては、これらのプログラム品質を高める教育の必要性を確認できたことから、これらに準じた教材の開発が必要である。

2.3. システムと課題付きテキストの開発

2.3.1. システム開発の成果

【サインイン画面】

2022 年 2 月 14 日時点で開発が完了している。図 1 で画面のスクリーンショットを示す。本画面はユーザがシステムにサインインするための画面である。サインインとパスワードの再設定を実施できる。メールアドレスとパスワードを入力フォームに入力し、「サインイン」ボタンをクリックすると、システムに登録されているユーザのメールアドレスとパスワードと一致しているならば、システムにサインインして、ホーム画面に遷移する。「パスワードがわからない」リンクをクリックすると、パスワードの再設定を行える。図 1 のように、開発版は誰でも「アカウントを作成」ボタンからアカウントを作成できるが、最終版では本画面を削除して、管理者のみがアカウントを作成するフローとする。

図 1. サインイン画面

【ホーム画面】

2022 年 2 月 14 日時点で開発が完了している。図 2 および図 3 で画面のスクリーンショットを示す。ユーザがサインイン後に最初に閲覧するダッシュボードとしての役割を持つ。「科目」タブには受講中の科目の一覧が表示される。図 2 では、「科目 0」「科目 1」「Python プログラミング基礎」という名前の科目を表示している。また、教育者と管理者が閲覧した際は、科目ごとにペンのアイコンの「編集」ボタンも表示する。科目名をクリックすると、科目画面に遷移する。科目に紐づく「編集」ボタンをクリックすると、該当科目の科目編集画面に遷移する。

「問題」タブには問題に関連したダッシュボードとおすすめの問題の一覧が表示される。ダッシュボードには、自身が解いた問題数と問題を解いたことで獲得した点数、システムを利用する学習者全

体の中での点数の順位が表示される。「おすすめの問題」には、教育者が学習者画面で指定した問題の一覧が表示される。図 3 では、「問題 0」「問題 1」という名前の問題を表示している。「すべての問題」リンクをクリックすると、問題一覧画面に遷移する。

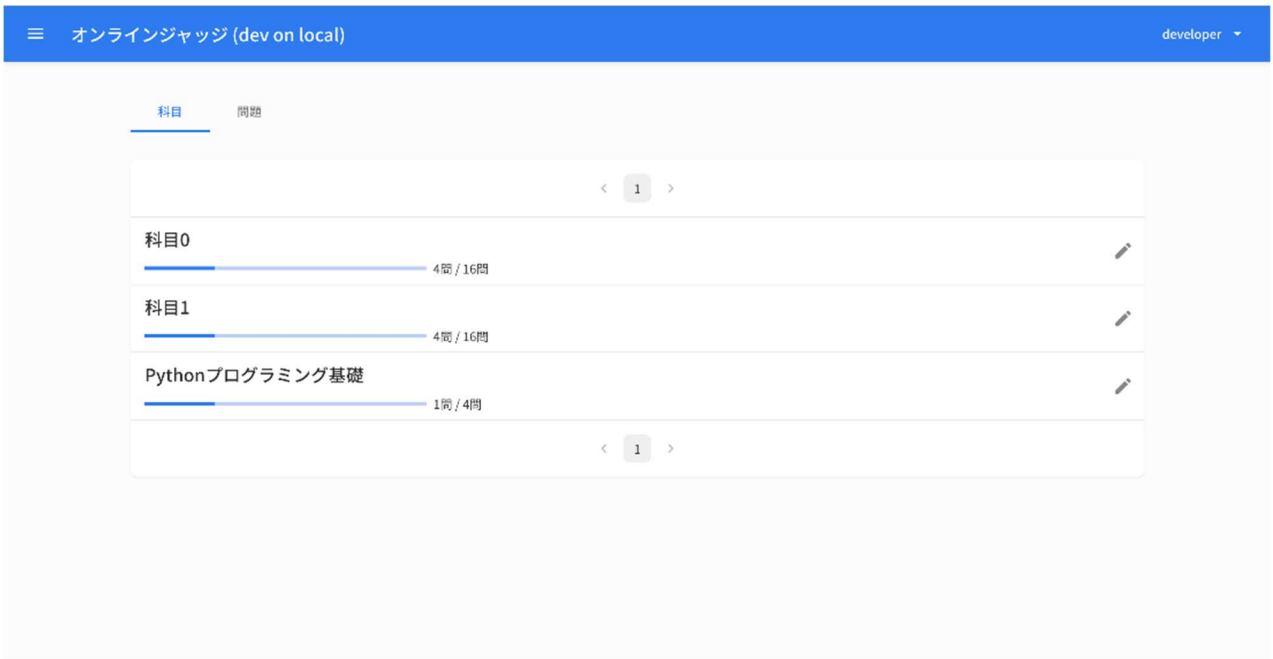


図 2. ホーム画面の科目タブ

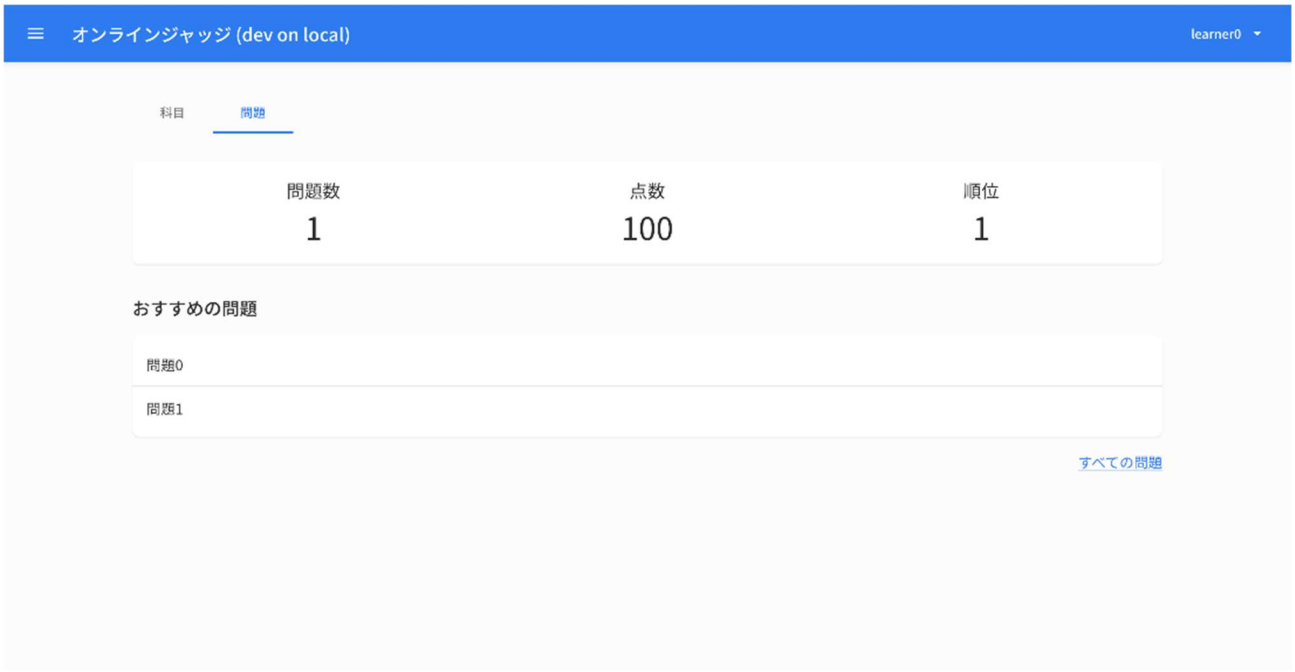


図 3. ホーム画面の問題タブ

【設定画面】

2022 年 2 月 14 日時点で開発が完了している。図 4 で画面のスクリーンショットを示す。本画面はユーザが設定を変更するための画面である。ユーザの表示面を変更できる。フォームに従って「表示

名」を入力し、「設定を変更」をクリックすることで、ユーザの表示名を変更できる。

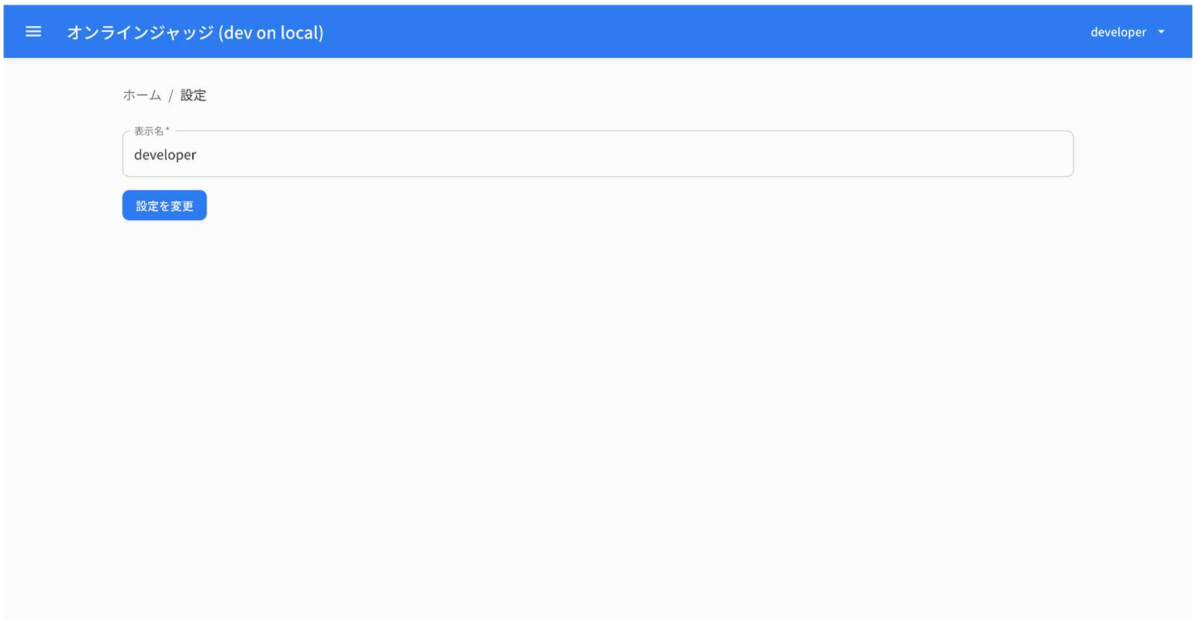


図 4. 設定画面

【科目一覧画面】

2022 年 2 月 14 日時点で開発が完了している。図 5 で画面のスクリーンショットに示す。本画面は科目一覧を表示する画面である。学習者が閲覧した際は、本人に紐づいている科目を表示する。教育者と管理者が閲覧した際は、システムに登録されている全ての科目を表示して、さらに、「作成」ボタン、科目ごとにペンのアイコンの「編集」ボタンも表示する。図 5 では、「科目 0」「科目 1」「科目 2」「科目 3」「Python プログラミング基礎」という名前の科目を表示している。科目名をクリックすると、該当科目の科目画面に遷移する。「作成」ボタンをクリックすると、科目作成画面に遷移する。科目に紐づく「編集」ボタンをクリックすると、該当科目の科目編集画面に遷移する。

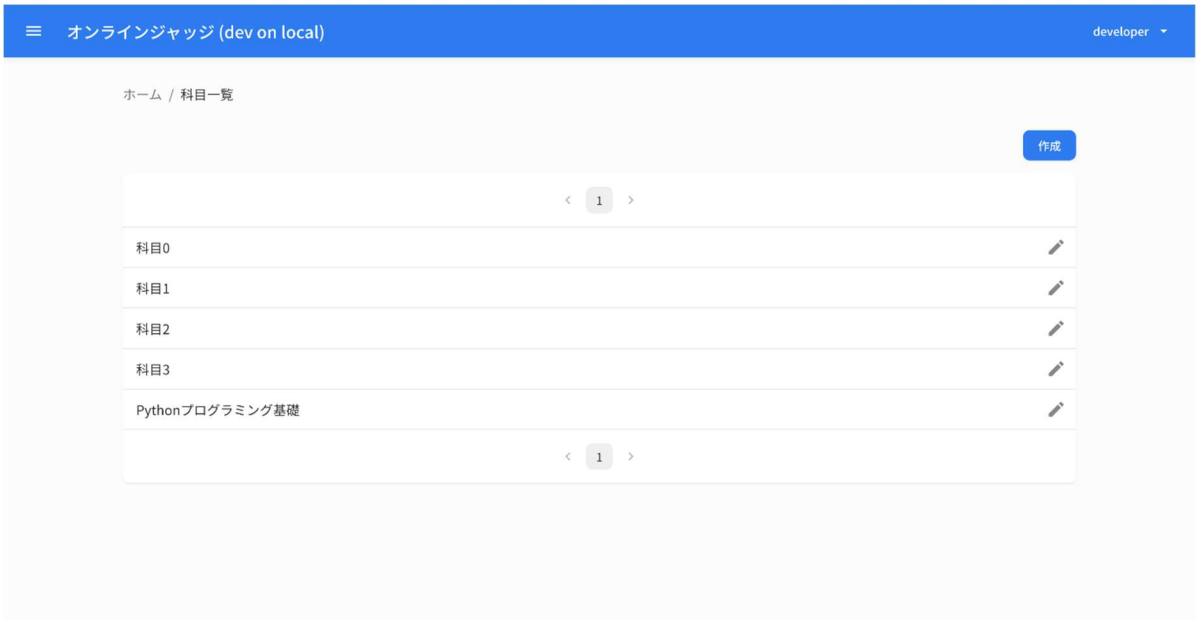


図 5. 科目一覧画面

【科目画面】

2022 年 2 月 14 日時点で開発が完了している。図 6 で画面のスクリーンショットを示す。本画面は科目を表示する画面である。本画面は科目の詳細および科目に紐づく講義の一覧を表示する。図 6 では、「講義 0」「講義 1」「講義 2」「講義 3」という名前の講義をと表示している。教育者と管理者が閲覧した際は、科目にペンのアイコンの「編集」ボタンと「受講学習者一覧」リンクも表示する。講義名をクリックすると、講義画面に遷移する。科目に紐づく「編集」ボタンをクリックすると、科目編集画面に遷移する。「受講学習者一覧」リンクをクリックすると、該当科目の進捗画面に遷移する。

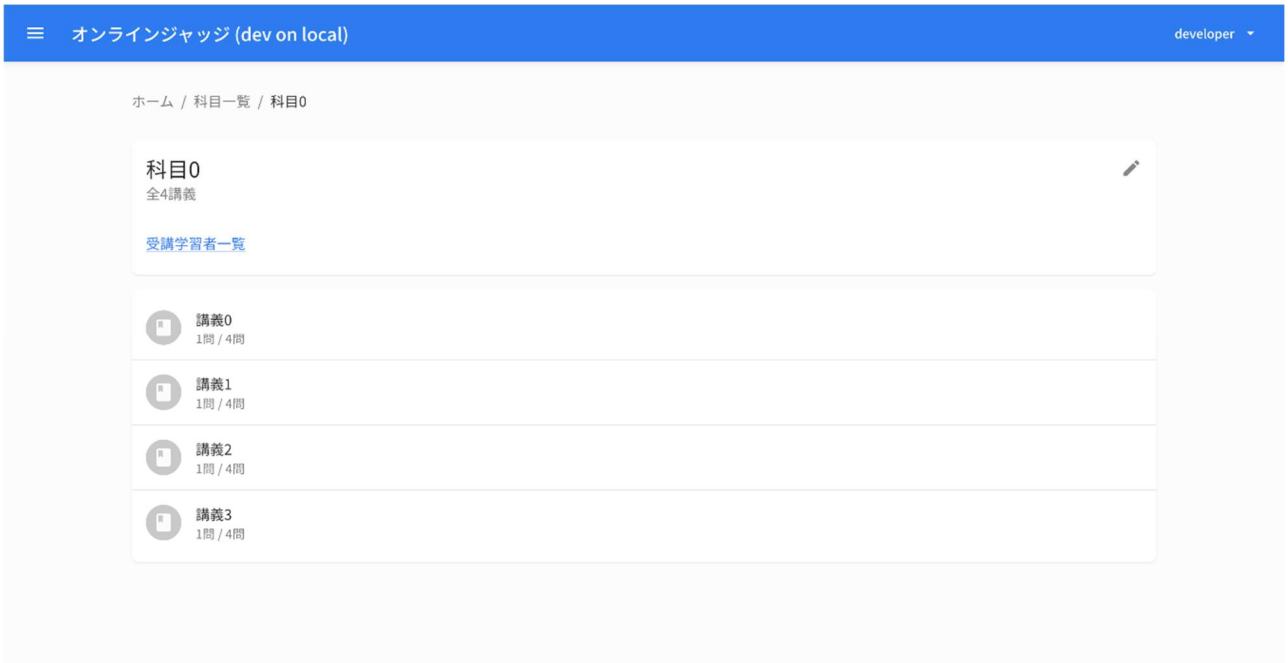


図 6. 科目画面

【講義画面】

2022 年 2 月 14 日時点で開発が完了している。図 7 で画面のスクリーンショットを示す。本画面は講義のテキスト教材および講義に紐づく問題の一覧を表示する画面である。図 7 では、「問題 0」「問題 1」「問題 2」「問題 3」という問題を表示している。教育者と管理者が閲覧した際は、問題ごとにペンのアイコンの「編集」ボタンを表示する。問題名をクリックすると、該当問題の問題画面に遷移する。問題に紐づく「編集」ボタンをクリックすると、該当問題の問題編集画面に遷移する。

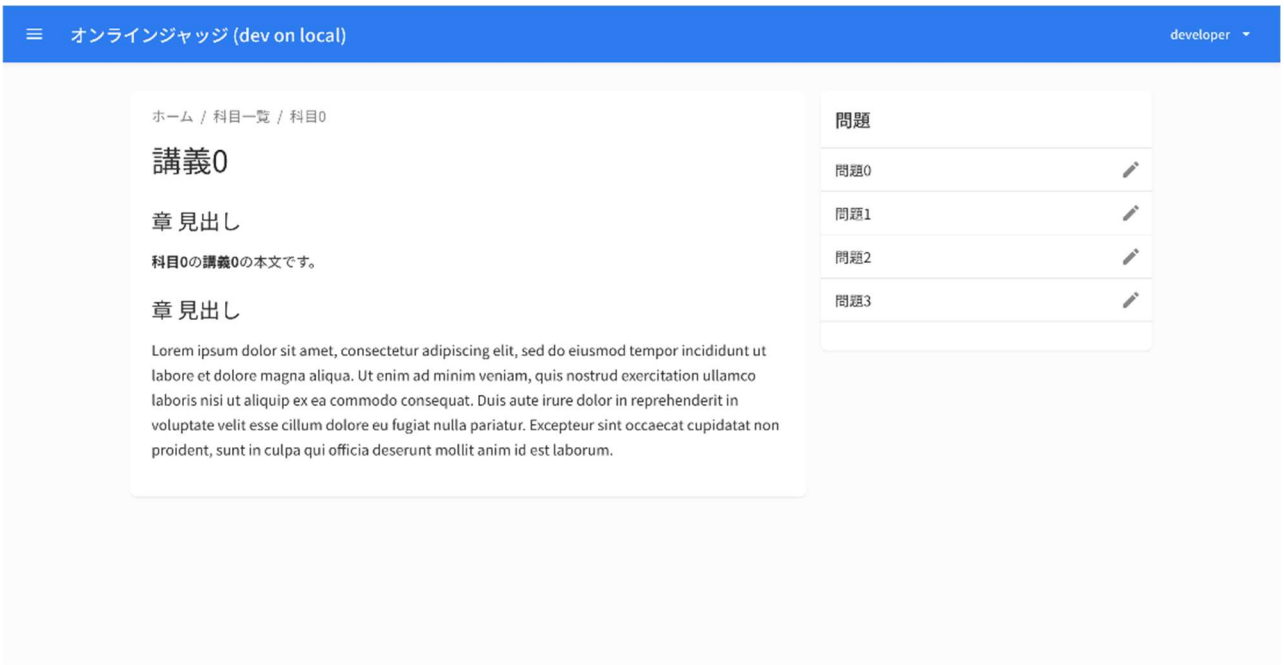


図 7. 講義画面

【問題一覧画面】

2022 年 2 月 14 日時点で開発が完了している。図 8 で画面のスクリーンショットを示す。本画面は問題の一覧を表示する画面である。図 8 では、「問題 0」「問題 1」「問題 2」「問題 3」という名前の問題を表示している。教育者と管理者が閲覧した際は、「作成」ボタンと問題ごとにペンのアイコンの「編集」ボタンも表示する。「作成」ボタンをクリックすると、問題作成画面に遷移する。問題をクリックすると、問題画面に遷移する。問題に紐づく「編集」ボタンをクリックすると、該当問題の問題編集画面に遷移する。

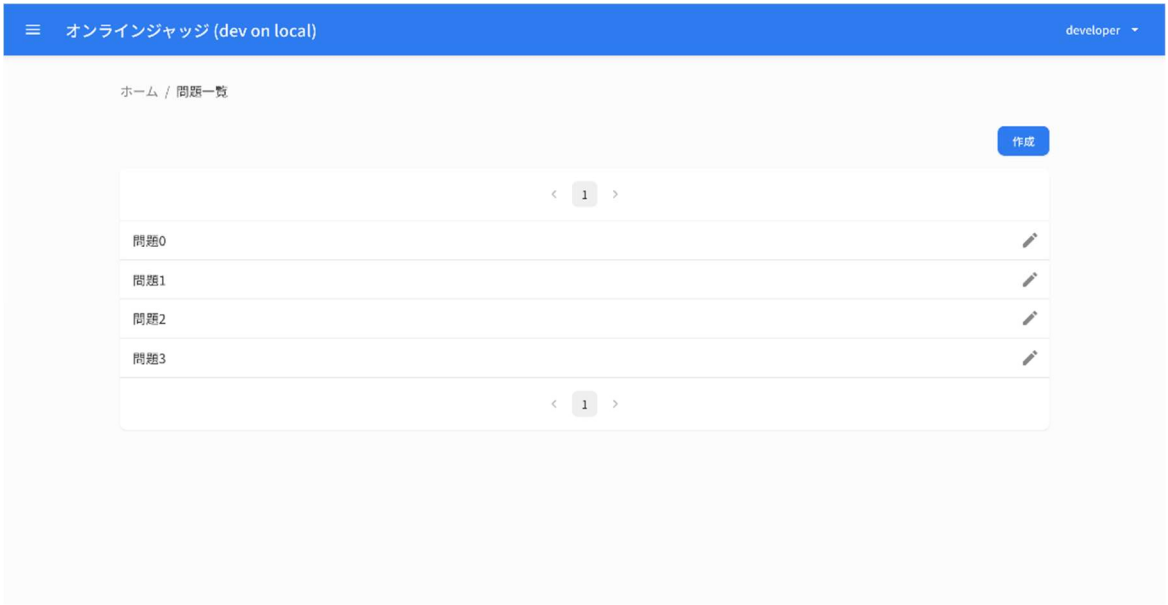


図 8. 問題一覧画面

【問題画面】

2022 年 2 月 14 日時点で開発が完了している。図 9、図 10 および図 11 で、問題画面のスクリーンショットを示す。本画面は問題を表示する画面である。問題の詳細と答案送信フォームを表示する。答案送信フォームは、コードエディタの機能を有する。図 9、図 10 および図 11 のように、画面左側に問題を表示している。画面右上にコードエディタを表示している。画面右下にコードエディタに記載されているソースコードのプログラムを実行する機能（テスト実行機能）と、コードエディタに記載されているソースコードのプログラムを提出する機能（提出機能）がある。テスト実行機能と提出機能はタブで切り替えることができる。

テスト実行機能では、「実行」ボタンをクリックすると、コードエディタに記載されているソースコードのプログラムを実行できる。この際に、「入力」のフォームに記載されている文字列は、標準入力を介してソースコードのプログラムに渡される。実行結果である標準出力と標準エラー出力は、「出力」、「エラー」の読み取り専用のフォームに表示される。図 10 で、プログラム実行後の「変数タブ」の表示を示す。「変数」タブでは、プログラム実行後に実行環境に存在した変数とその値の表が表示される。

提出機能では、「提出」ボタンをクリックすると、コードエディタに書かれたソースコードのプログラムが実行されて、実行結果を「判定」に表示する。また、テストケースごとの実行時の実行時間、メモリ使用量をそれぞれ「実行時間」、「メモリ使用量」に表示する。判定結果には、「RE」、「TLE」、「MLE」、「WA」、「AC」がある。プログラムが異常終了した場合は「RE」を表示する。プログラムが正常終了しても、実行時間が問題の「実行時間制限」を超過した場合は「TLE」を表示する。プログラムが問題で指定した実行時間内に正常終了しても、メモリ使用量が問題の「メモリ制限」を超過した場合は「MLE」を表示する。プログラムが問題で指定したメモリ使用量以内かつ実行時間内に正常終了しても、テストケースと異なる標準出力が出力された場合は「WA」を表示する。プログラムが問題で指定したメモリ使用量以内かつ実行時間内に正常終了して、テストケースと同じ標準出力が出力された場合は「AC」を表示する。図 11 で、全てのテストケースにおいて「AC」の判定が得られた場

合を示す。提出したソースコードは判定結果とともにデータベースに保存されて、答案一覧画面で、ソースコードと判定結果を閲覧できるようになる。

図 9. テスト実行後の問題画面

図 10. テスト実行後の問題画面の変数タブ

図 11. 提出後の問題画面

【答案一覧画面】

2022 年 2 月 14 日時点で開発が完了している。図 12 で画面のスクリーンショットを示す。本画面は答案の表を表示する画面である。学習者が閲覧した際は、自身が提出した答案を表示する。教育者と管理者が閲覧した際は、すべての答案を表示する。答案ごとの提出「日時」、提出先の「問題」、提出した「ユーザ」、実行したプログラミング言語を示す「実行環境」、採点結果の「点数」、採点結果の「判定」が表示される。「問題」列のリンクをクリックすると問題画面に遷移する。「詳細」列のリンクをクリックすると答案画面に遷移する。

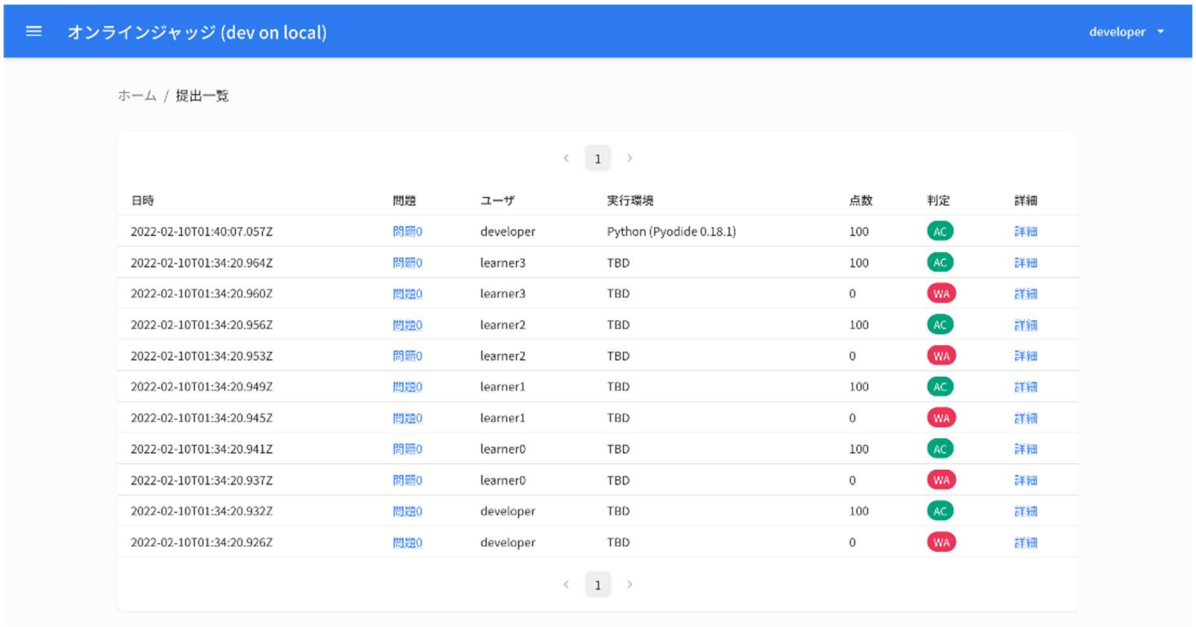


図 12. 答案一覧画面

【答案画面】

2022 年 2 月 14 日時点で開発が完了している。画面のスクリーンショットを図 13 に示す。答案の詳細を表示する画面である。図中の「日時」は答案の提出日時、「問題」は提出先の問題名、「ユーザ」は提出したユーザの表示名、「実行環境」は実行環境を示す文言、「点数」は採点結果の点数、「判定」は採点結果の判定結果、「実行時間」は実行に要した時間、「メモリ使用量」はメモリの使用量、「ソースコード」は学習者が投稿したソースコードである。「コメント」では、答案に対して投稿されたコメント、コメントしたユーザ名、コメントした日時が表示される。開いている答案を提出した学習者と教育者、管理者がコメントを投稿できる。フォームに従って「行番号」「本文」を入力して、「コメント」をクリックすると、コメントを投稿できる。「行番号」でソースコードの行を指定すると、コメントでソースコードの当該行を引用できる。

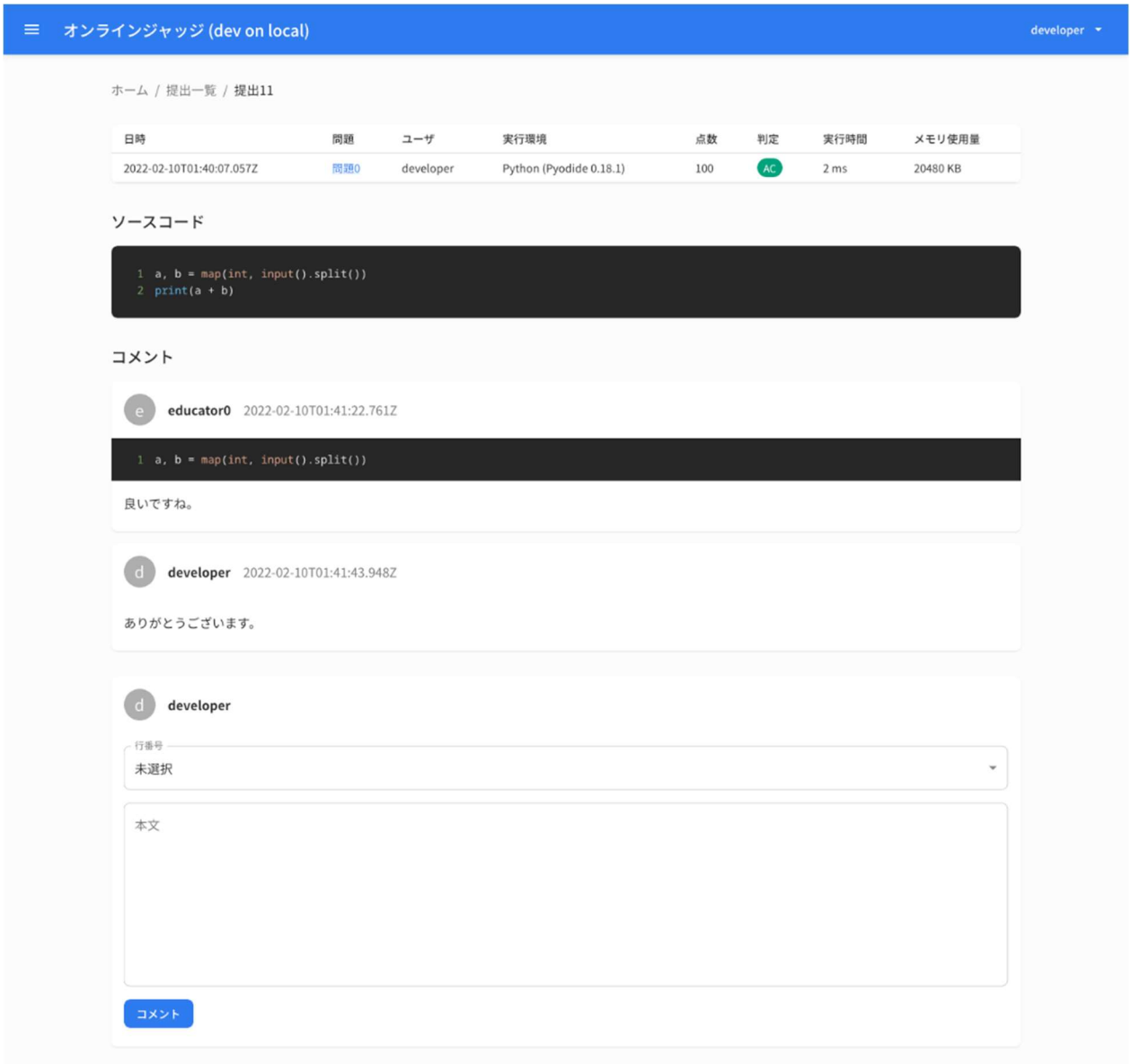


図 13. 答案画面

【科目作成画面】

2022 年 2 月 14 日時点で開発が完了している。図 14 で画面のスクリーンショットを示す。本画面は科目を新しく作成する画面である。科目および科目に紐づく講義（テキスト教材を含む）を作成できる。また、システムに登録されている問題を講義と紐づけることができる。フォームに従って「科目名」、「講義名」、「本文」を入力して、システムに登録されている「問題」の ID を入力することで、新しく科目を作成できる。「講義」、「本文」、は、ファイルをインポートして入力できる。「講義」の「本文」は Markdown 形式で入力する必要がある。

「講義を追加」をクリックすると、「講義」の入力フォームを追加でき、追加で講義を入力できる。「問題を追加」をクリックすると、「問題」の入力フォームを追加でき、追加で問題の ID を入力できる。

「プレビュー」ボタンをクリックすると、モーダルウィンドウで「講義」の「本文」を講義画面上での描画結果を確認できる。

「科目を作成」ボタンをクリックすると、入力した「科目名」、「講義名」、講義の「本文」、講義に紐づく「問題」の ID がデータベースに登録されて、科目一覧画面に表示されるようになる。

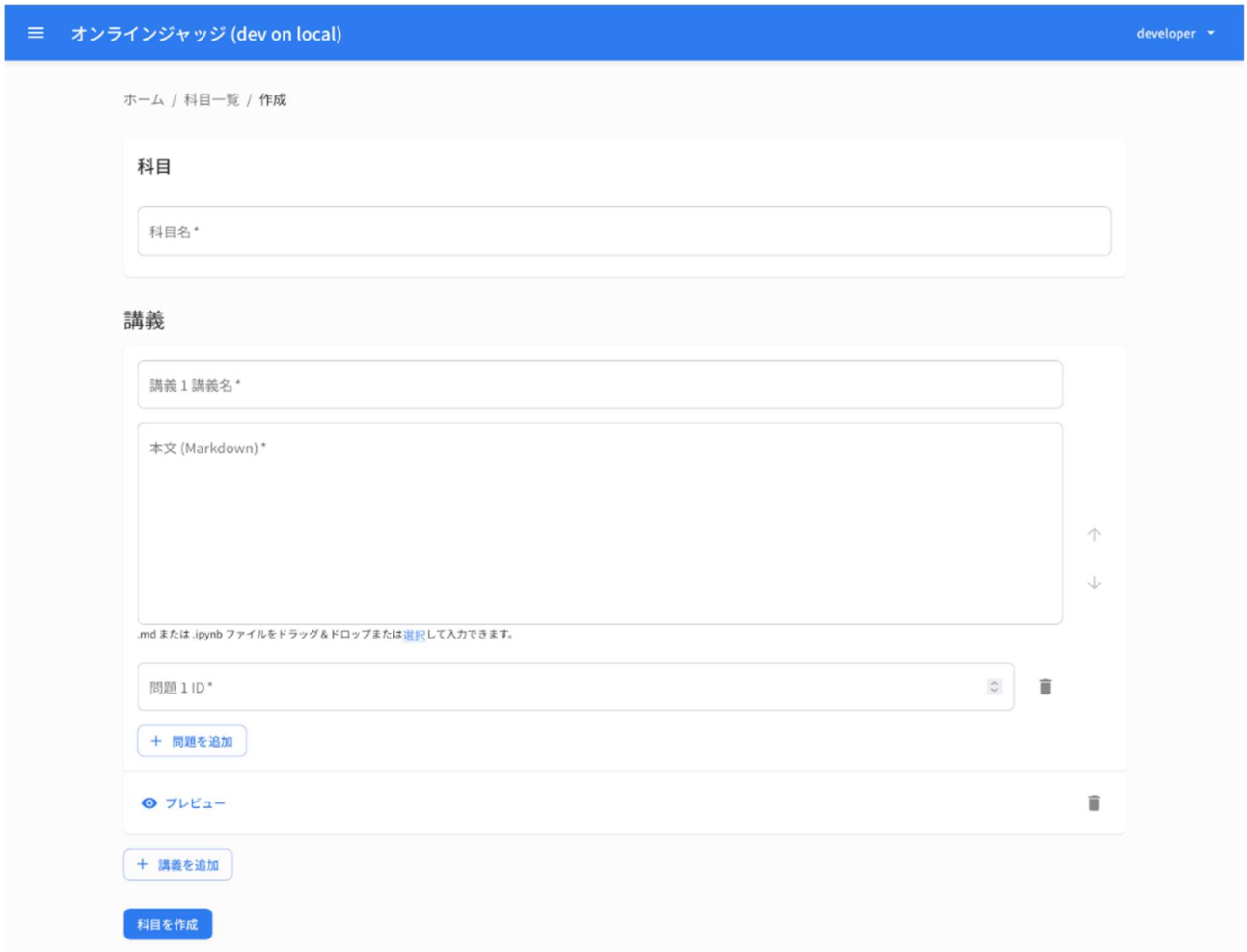


図 14. 科目作成画面

【科目編集画面】

2022 年 2 月 14 日時点で開発が完了している。図 15 で画面のスクリーンショットを示す。本画面は既にシステムに登録されている科目を編集する画面である。入力フォームは科目作成画面と同じだが、編集対象の科目が既に入力されている。「科目を保存」ボタンをクリックすると、変更後の科目がデータベースに保存され、科目一覧画面などに表示されるようになる。「科目を削除」ボタンをクリックするとデータベースから科目が削除され、科目一覧画面などでも表示されなくなる。

図 15. 科目編集画面

【問題作成画面】

2022 年 2 月 14 日時点で開発が完了している。図 16 で画面のスクリーンショットを示す。本画面は問題を新しく作成する画面である。フォームに従って「問題」、「テストケース」、「模範解答」を入力することで問題を作成できる。「問題」の「本文」、「テストケース」の「標準入力」と「標準出力」、「模範解答」の「ソースコード」は、ファイルをインポートして入力できる。また、「問題」の「本文」は Markdown 形式で入力する必要がある。

「テストケースを追加」をクリックすると、「テストケース」の入力フォームを追加でき、追加でテ

ストケースを入力できる。「模範解答を追加」をクリックすると、「模範解答」の入力フォームを追加でき、複数の模範解答のソースコードを入力できる。

「問題を作成」ボタンをクリックすると、入力した「問題」、「テストケース」、「模範解答」がデータベースに登録されて、問題一覧画面などに表示されるようになる。

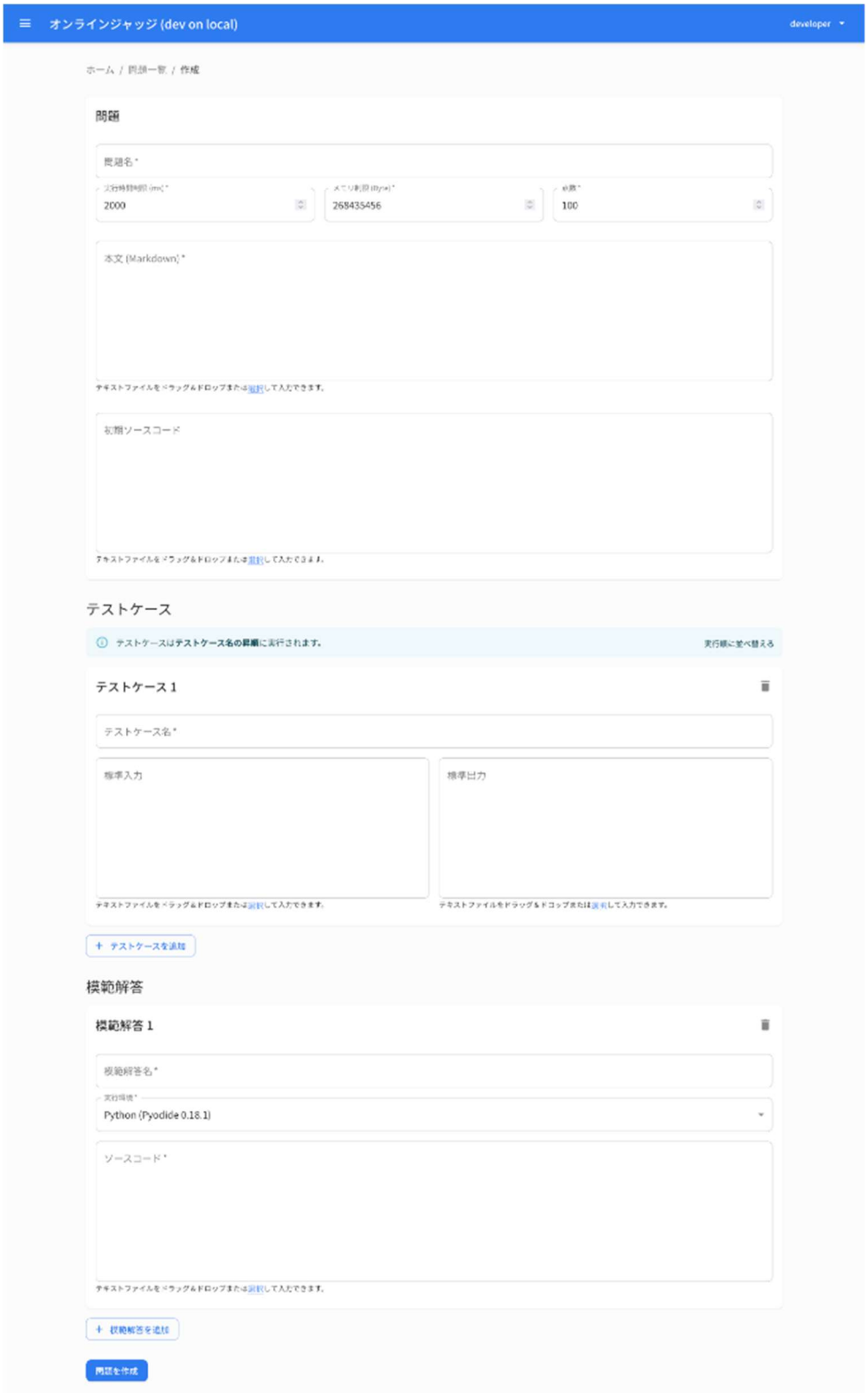


図 16. 問題作成画面

【問題編集画面】

2022 年 2 月 14 日時点で開発が完了している。図 17 で画面のスクリーンショットを示す。本画面は既存の問題を編集する画面である。入力フォームは問題作成画面と同じだが、編集対象の問題が既に入力されている。「問題を更新」ボタンをクリックすると、変更後の「問題」、「テストケース」、「模範解答」がデータベースに保存され、問題一覧画面などに表示されるようになる。「問題を削除」ボタンをクリックするとデータベースから問題が削除され、問題一覧画面などでも表示されなくなる。

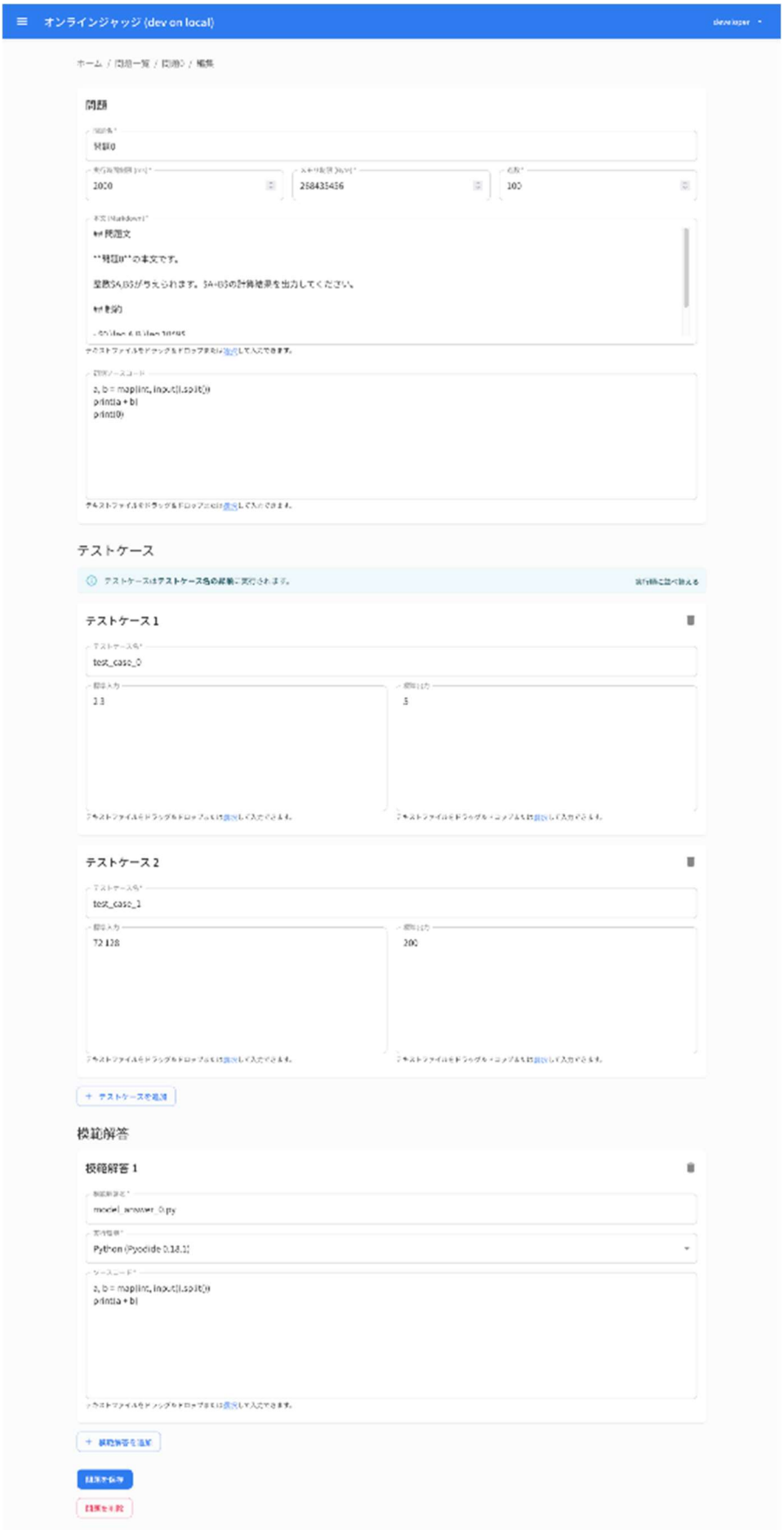

オンラインジャッジ (dev on local)
developer
問題
問題0
2000
258435456
100
問題0の本文です。
整数SA,SBが与えられます。SA+SBの計算結果を出力してください。
a, b = map(int, input().split())
print(a + b)
print(0)
テストケース
テストケースはテストケース名の昇順に実行されます。
テストケース 1
test_case_0
2 3
5
テストケース 2
test_case_1
72 128
200
模範解答
模範解答 1
model_answer_0.py
Python (Pyodide 0.18.1)
a, b = map(int, input().split())
print(a + b)

図 17. 問題編集画面

【学習者一覧画面】

2022 年 2 月 14 日時点で開発が完了している。図 18 で画面のスクリーンショットを示す。本画面はシステムに登録されている学習者の一覧を表示する画面である。各学習者の情報として、「メールアドレス」、「表示名」、「氏名」を表示する。

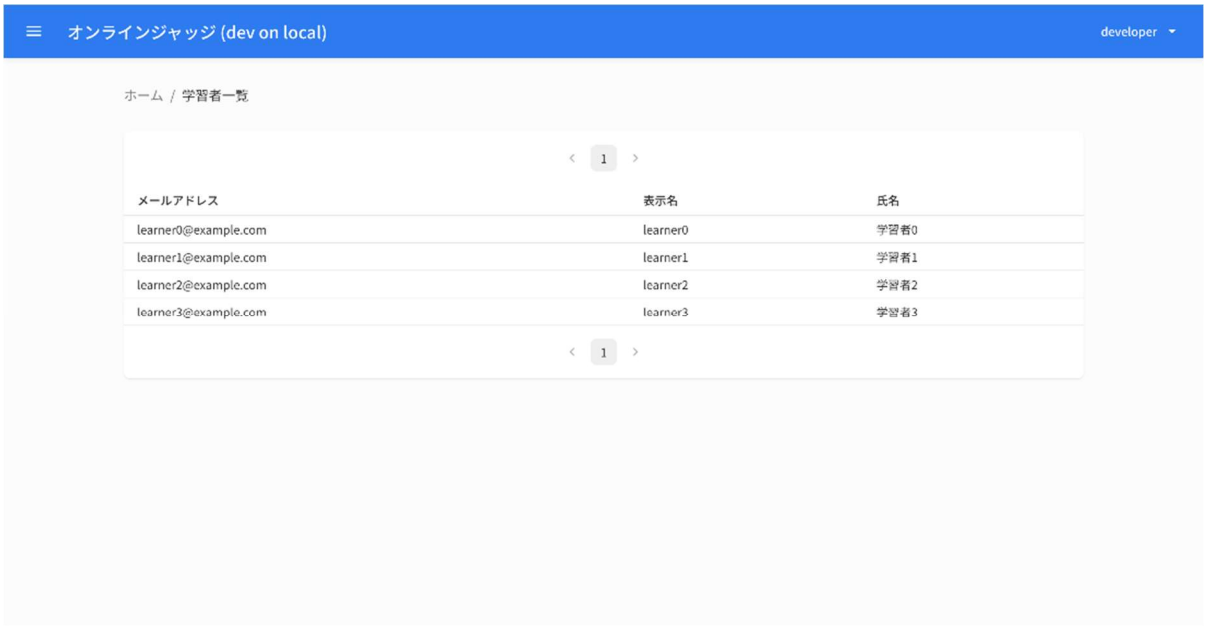


図 18. 学習者一覧画面

【学習者画面】

2022 年 2 月 14 日時点で開発が完了している。図 19、図 20 および図 21 で画面のスクリーンショットを示す。本画面は学習者の学習の進捗を表示したり、学習者の設定を変更したりする画面である。「進捗」タブでは当該学習者が正解した問題数が表示される。「推奨問題」タブでは当該学習者のホーム画面で「おすすめの問題」として表示する問題を編集できる。「設定」タブでは当該学習者に適用する動機づけ AI の設定を編集できる。

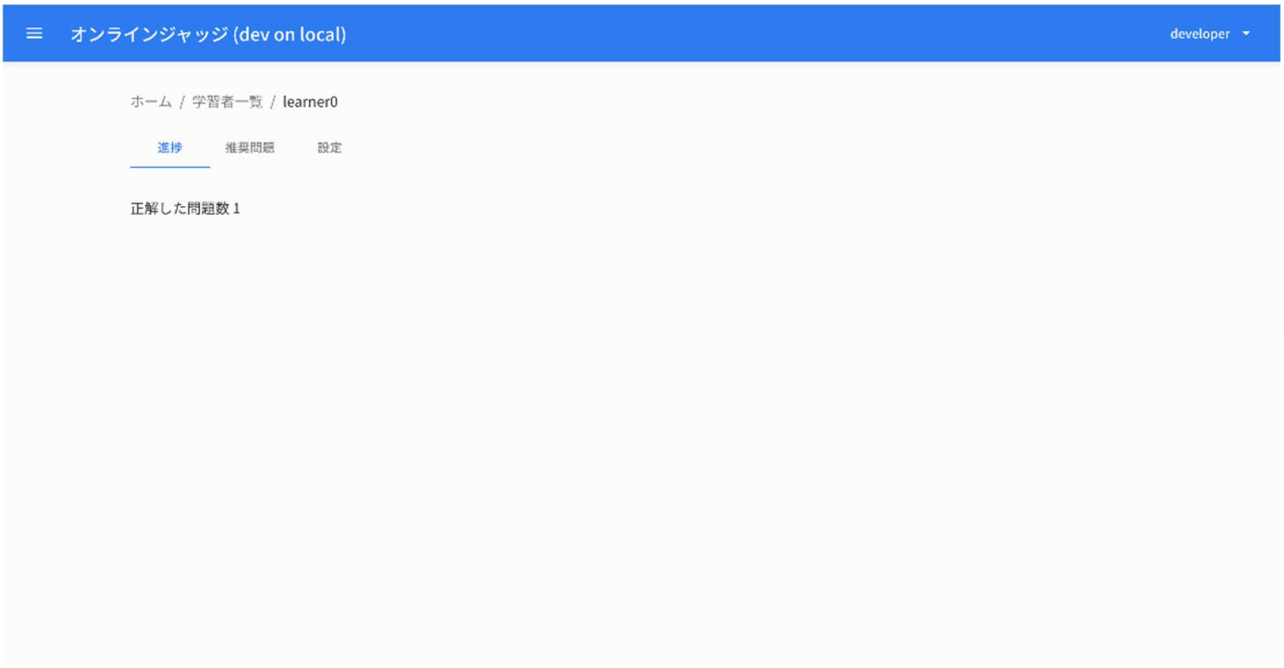


図 19. 学習者画面の進捗タブ

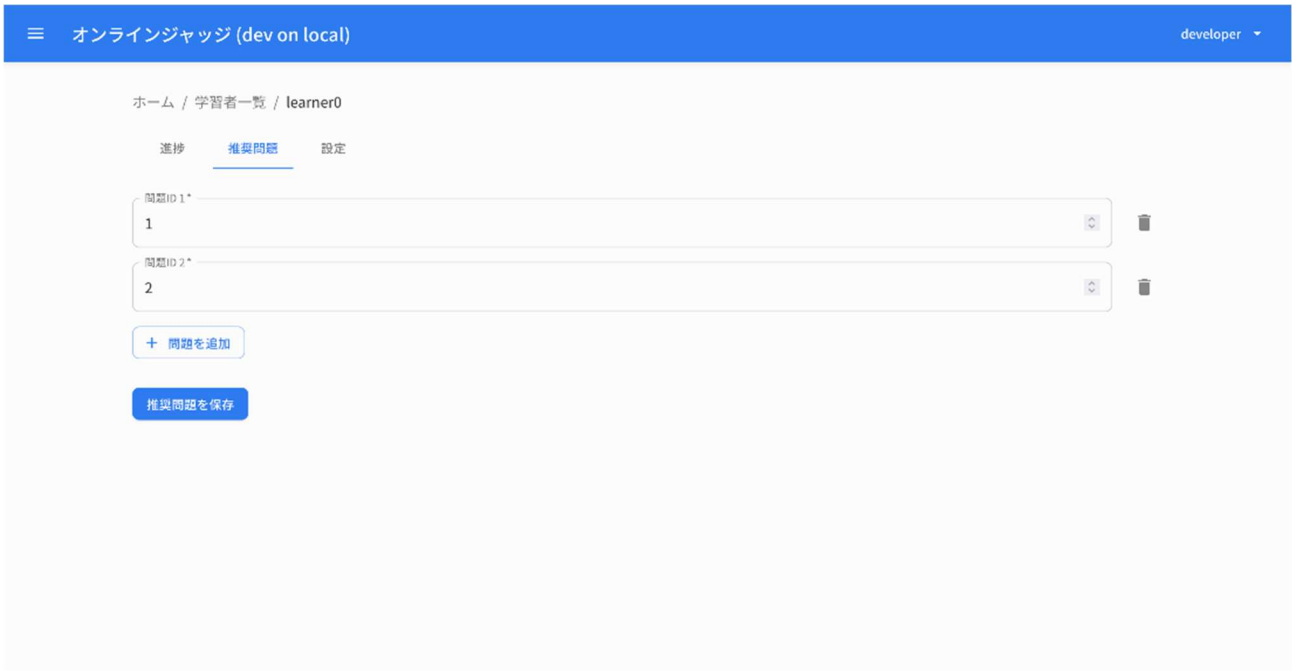


図 20. 学習者画面の推奨問題タブ

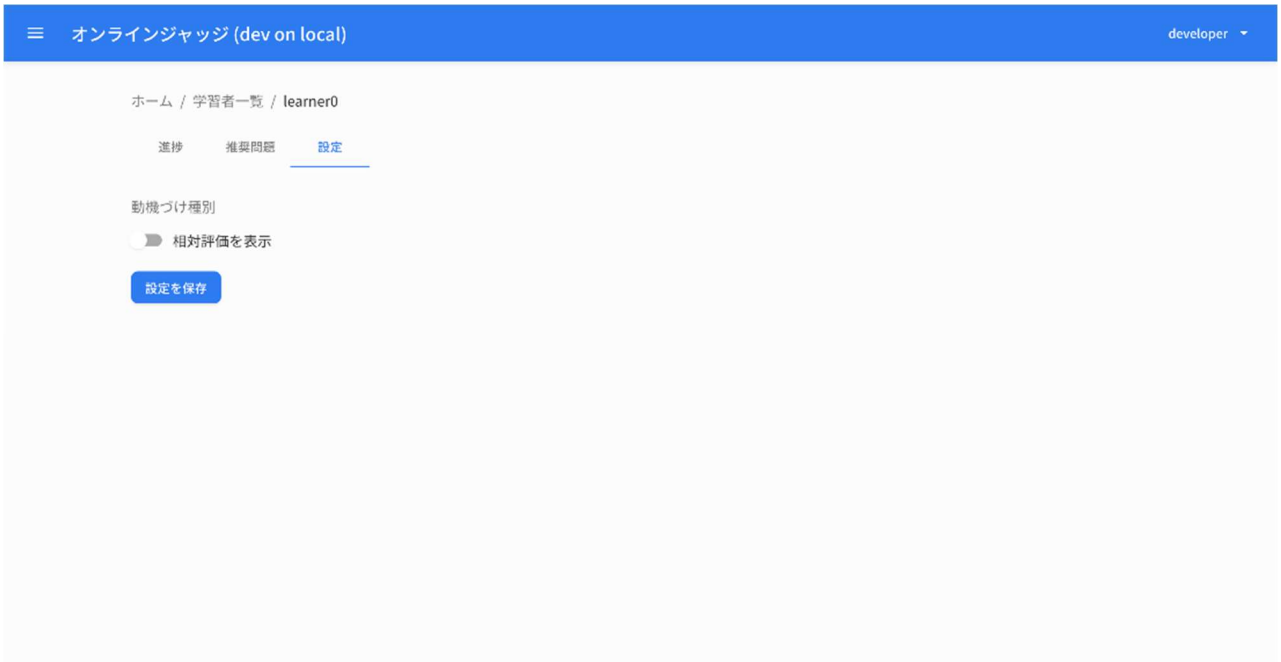


図 21. 学習者画面の設定タブ

【科目の進捗画面】

2022 年 2 月 14 日時点で開発が完了している。図 22 で画面のスクリーンショットを示す。本画面はある科目を受講する学習者が解いた問題を一覧する画面である。画面中の表の行が 1 学習者を、「X-Y」（X、Y は正整数）と名付けられた列が 1 つの問題を表す。「X-Y」は X 番目の講義の Y 番目の問題を表す。ある学習者がある問題に正解済みならば該当するセルにチェックアイコンが表示される。

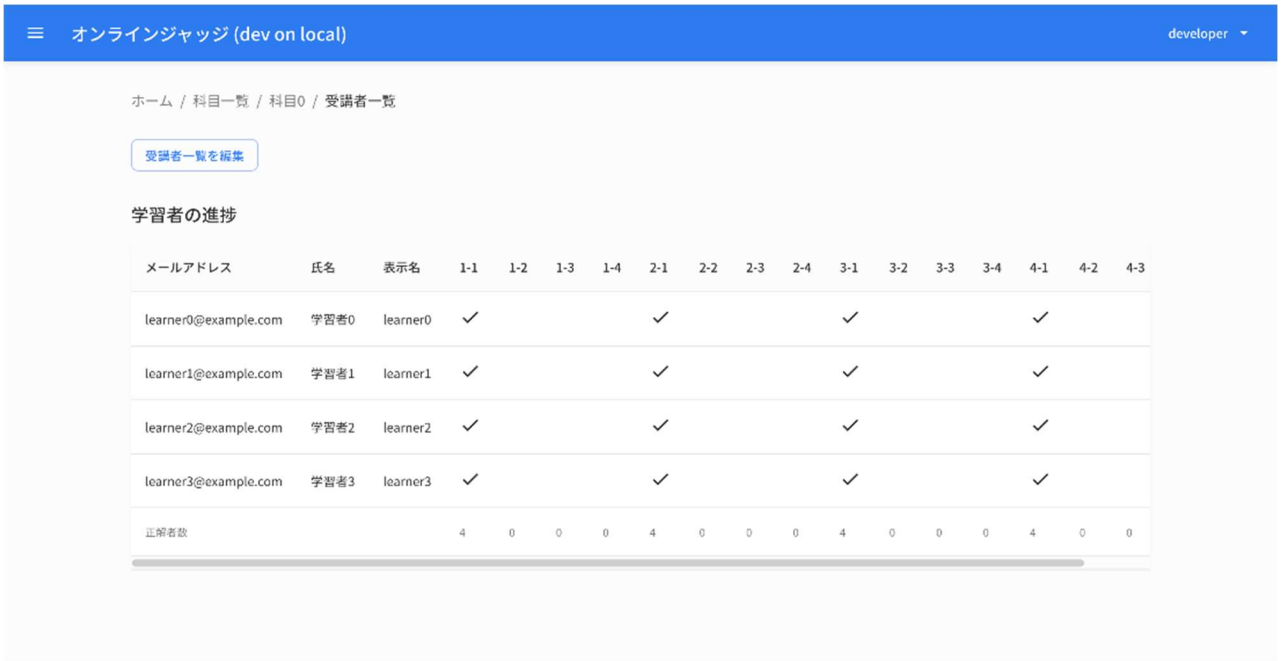


図 22. 科目の進捗画面

【ユーザ一覧画面】

2022 年 2 月 14 日時点で開発が完了している。図 23 で画面のスクリーンショットを示す。本画面はユーザ一覧を表示する画面である。ユーザ一覧を CSV でエクスポートできる。各ユーザの情報として、「メールアドレス」、「表示名」、「氏名」、「役割」を表示する。「CSV ファイルをエクスポート」ボタンをクリックすると、ユーザ一覧を CSV でエクスポートできる。「CSV ファイルをインポート」ボタンをクリックするとユーザ CSV インポート画面に遷移する。

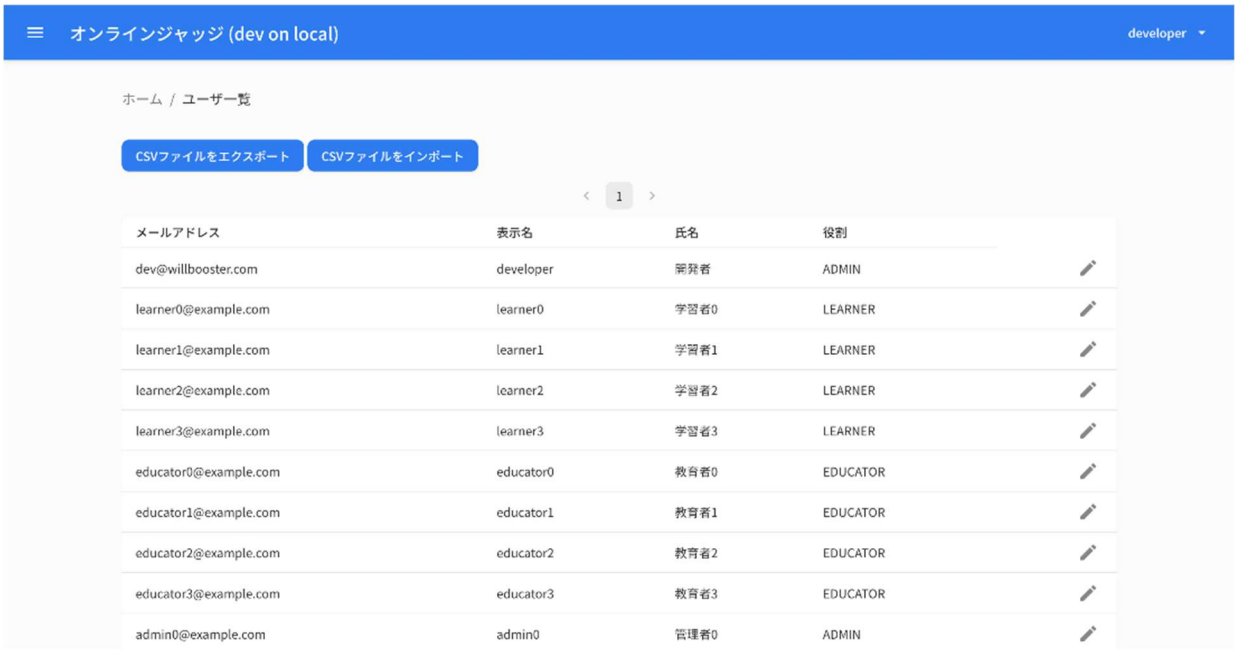


図 23. ユーザ一覧画面

【ユーザ編集画面】

2022 年 2 月 14 日時点で開発が完了している。図 24 で画面のスクリーンショットを示す。本画面はユーザのメールアドレスを変更する画面である。メールアドレスをテキストフィールドに入力できる。「メールアドレスを変更」ボタンをクリックすると、入力した値がデータベースに保存される。

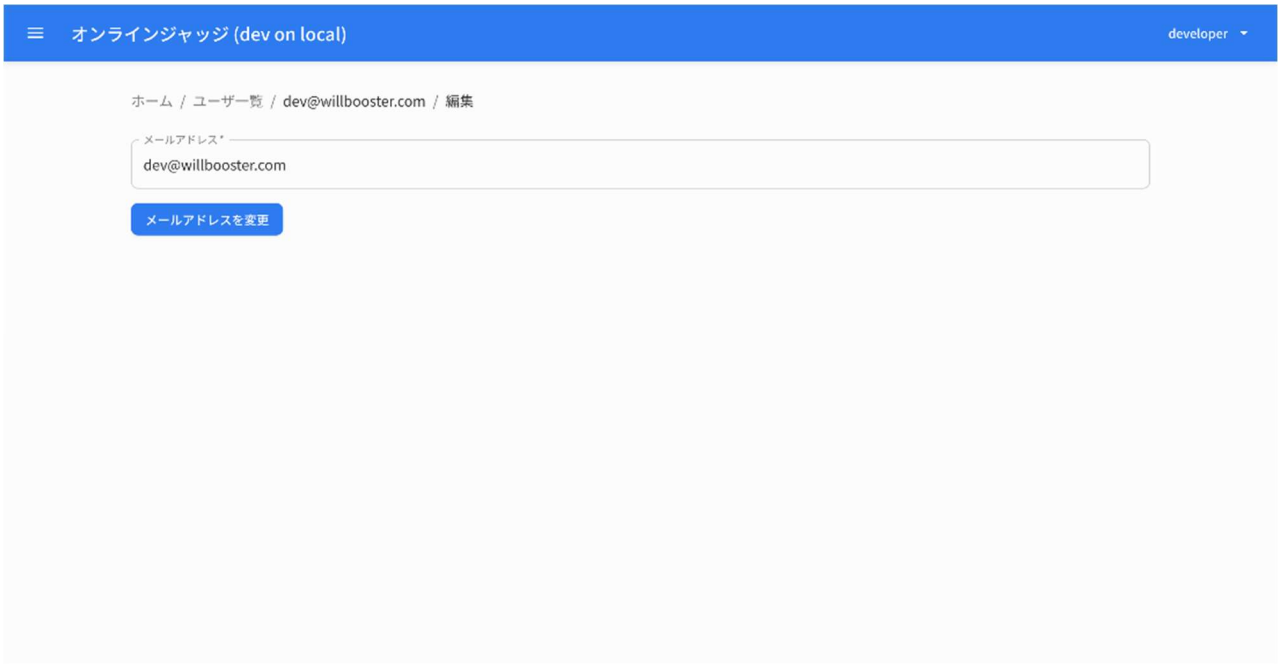


図 24. ユーザ編集画面

【ユーザ CSV インポート画面】

2022 年 2 月 14 日時点で開発が完了している。図 25 で画面のスクリーンショットを示す。本画面はシステムに登録されているユーザを CSV ファイルによって一括して変更する画面である。上部のドラッグ・アンド・ドロップ領域に CSV ファイルをドラッグ・アンド・ドロップするか、「ファイルを選ぶ」ボタンをクリックすることで CSV ファイルを入力できる。CSV ファイルが入力されると、現在のユーザ一覧と入力された CSV ファイルに含まれるユーザ一覧の差分が表示される。「インポートを実行」ボタンをクリックすると、変更がデータベースに保存される。

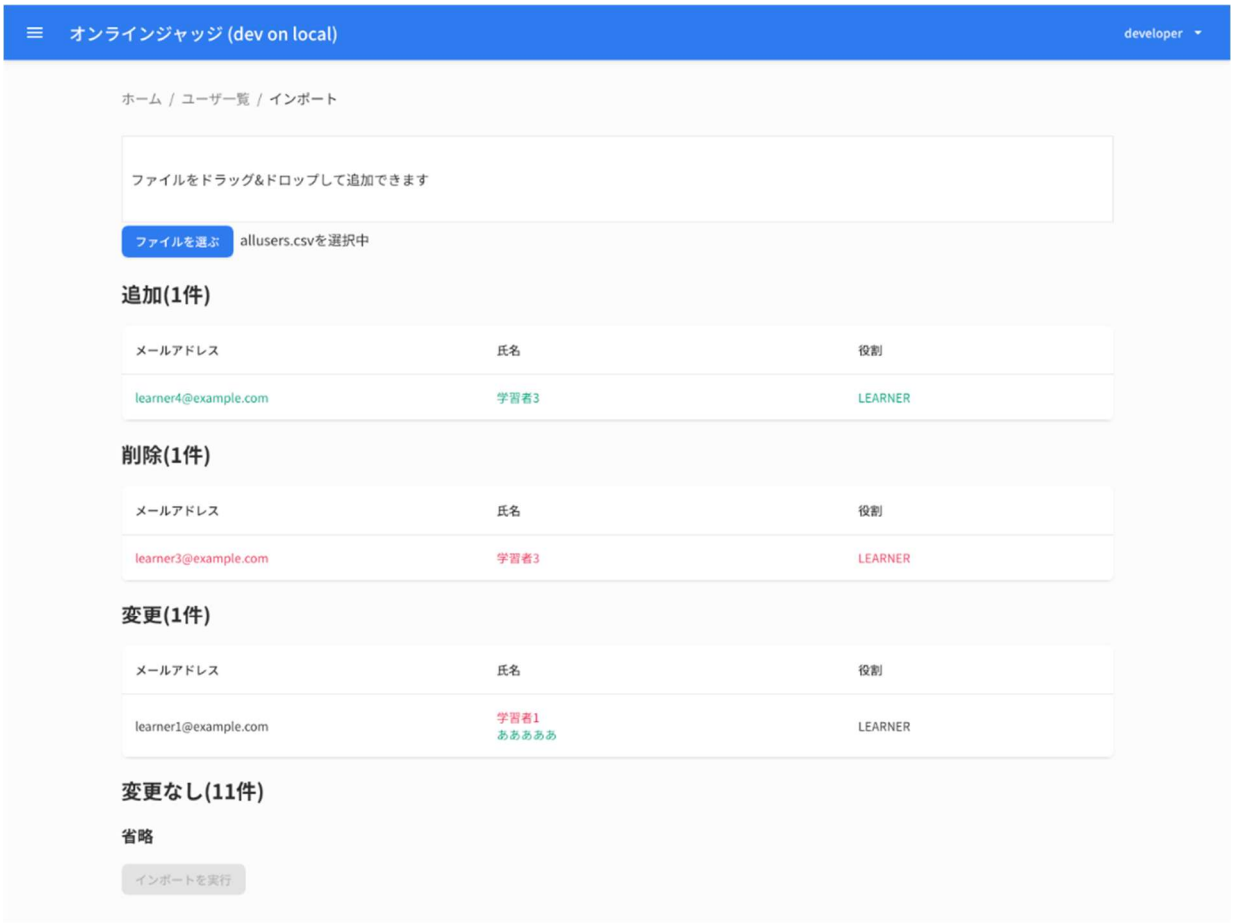


図 25. ユーザ CSV インポート画面

2.3.2. 課題付きテキスト開発の成果

課題付きテキストのテキスト部分は、1 章（90 分授業の 1 コマ分）を 1 つの Markdown ファイルで作成しており、閲覧に適したレイアウトになるように、開発中のシステムで Markdown フィルをレンダリングする。現在、レンダリング機能は開発中であるため、本資料では既存のコードエディタである Visual Studio Code でレンダリングした結果で代用する。

各章は複数の節から構成されており、各節は以下で述べる文章構成（以降、説明ユニットと呼ぶ）の繰り返しから構成されている。説明ユニットは、「導入」「サンプルコード」「解説」の 3 要素から構成されており、導入はこれから解説しようとする概念を取り上げるための文章、サンプルコードは取り上げた概念の解説に適したソースコードの例および実行結果、解説はソースコードの意味の説明を通して、取り上げた概念を解説するための文章である。

図 26 は第 2 章から抜粋した説明ユニットで、四則演算について解説している。該当説明ユニットでは、Python において数学で扱われている演算子の多くを利用できる旨を紹介した上で、四則演算のサンプルコードを示し、その後、Python プログラムと数学との表記の差異について解説している。

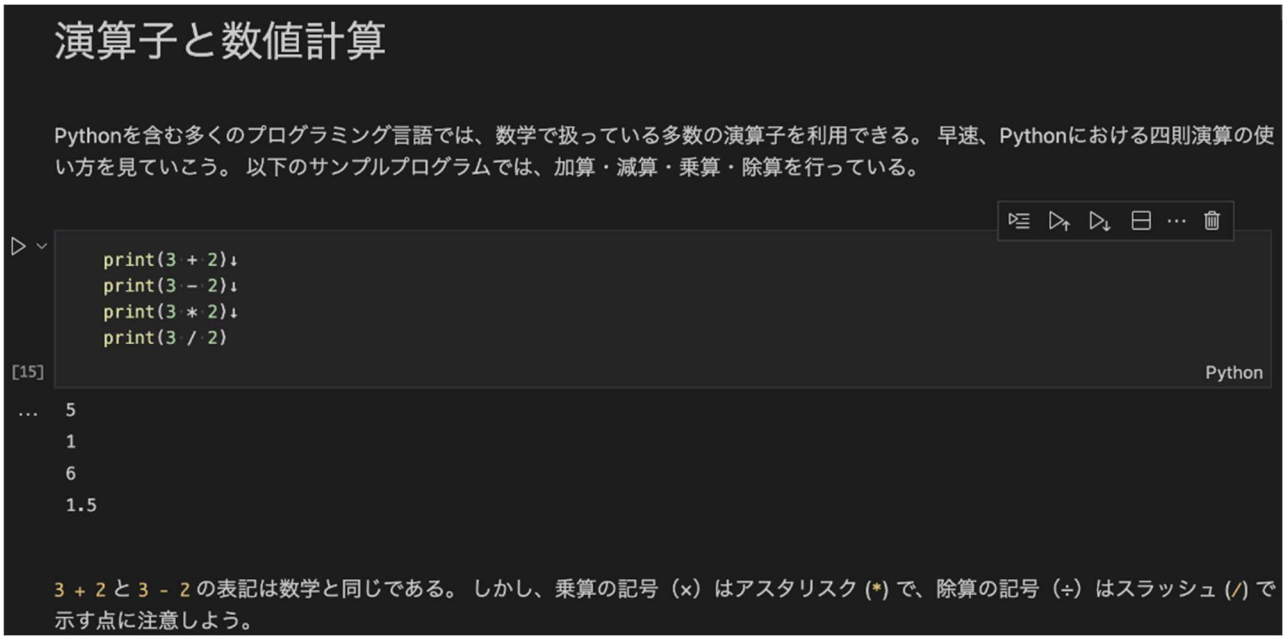


図 26. 第 2 章から抜粋した説明ユニット

各章はテキストに加えて、取り上げた内容に沿ったいくつかの演習課題を有している。各演習課題は問題文を格納した 1 つの Markdown ファイルと、学習者が投稿したプログラムが仕様を満たしているかどうかを判定するためのテストスイートのファイル群、模範解答の Python ファイルから構成されている。図 27 で演習課題の問題文の例、図 28 で図 27 の模範解答の Python ファイルを示す。

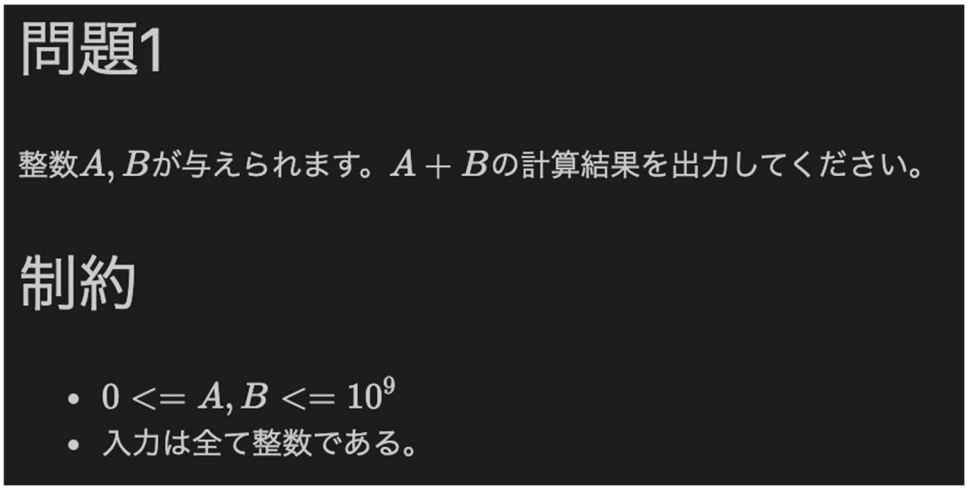


図 27. 演習課題の問題文の例

```
#!/usr/bin/env python3
↓
a, b = map(int, input().split())
print(a + b)
```

図 28. 図 27 の問題に対する模範解答の Python ファイル

また、上記演習課題に対するテストスイートを構成するテストケースの一例として、入力は「2 3」で、出力は「5」である。

2.3.3. プログラム評価基準案

河原学園の担当者様と協議の上、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、コーディング規約とソースコードメトリクス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。

本評価基準案は減点制を採用している。その理由として、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も理由に挙げられる。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。

以上を踏まえて、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、減点するような評価基準案を考案した。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した。

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。

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

なお、係数等は実際の運用を通して調整を要するものであるため、システム上で数式を調整するための仕組みを導入する予定である。

3. 課題付きテキスト

本年度作成した課題付きテキストを次ページ以降に掲載する。