プログラミング演習におけるエラー自動解説の有用性の評価

大和 祐介[1,a)]　石尾 隆[1,b)]　嶋利 一真[1,c)]　松本 健一[1,d)]

概要：プログラミングを学ぶ初学者の多くが直面する困難の 1 つが，コンパイル時や実行時にプログラム開発環境（演習環境）から提示されるエラーメッセージを理解し，それらエラーの原因を特定，修正することである．本研究では，初学者のエラー特定や修正作業の支援を目的として，コンパイラが標準で出力するエラーメッセージに加えて，エラーの原因となる情報を含む解説文を自動生成し提示する手法を提案する．提案手法では，英文として提示されるエラーメッセージと共に，初学者がコード記述を行う画面と同じ画面でその解説文を閲覧することができる．オンライン授業システム C2Room 上で提案手法を実装し，評価実験を行った．評価実験では，初学者により引き起こされるコンパイルエラーや実行時エラー，また，演習中に学生とティーチングアシスタント間でやり取りされた質問チャットなどを分析した．その結果，解説の提示でプログラミングに関する質問数の割合が減ることを確認した．

1. はじめに

近年，技術革新の急速な発展と IT 人材の需要の高まりにより，情報教育が盛んに行われている．それに伴い，小学校では 2020 年度，中学校では 2021 年度，高校では 2022 年度からプログラミング教育が必修化される等，初学者がプログラミングを学ぶ機会が増えている．

プログラミングの授業では，初学者は実際に手を動かしながらプログラムの作成，エラーの修正を繰り返し，課題を進める．プログラミング演習において学生が遭遇する問題の１つがコンパイルエラーや実行エラーである [3]．初学者にとってエラーメッセージの意味を正しく理解し，エラーを修正することは難しい [2], [13]．実際に，プログラマーはプログラミングの多くの時間をエラー原因の検索や修正に費やしている [8]．この問題に対し，教員が学生を個別に指導することは難しいため，学生が遭遇した問題を自己解決できるように支援する技術が研究されている [15], [18]．また初学者にとってエラーメッセージはわかりにくいことから，既存研究 [12] では，エラーメッセージに対応する Stack Overflow の質問を自動検索し，リンクを提示する手法が提案されている．しかし，Stack Overflow 上で行われている質問や回答は，学生が使っているプログラミング環境に関係がないため，教員が行えるような，プログラミング環境に対する操作の指示のような情報を含まない．

そこで本研究では，エラーメッセージの自動解説を作成した．そして実際の授業に導入し，有用性の評価を行う．後程説明するデータセットを分析し，以下の項目について分析を行う．

- RQ1. エラーメッセージ自動解説は学生から必要とされているか
- RQ2. エラーメッセージ自動解説は個人に対して効果があるか
- RQ3. エラーメッセージ自動解説はクラス全体に影響を与えるか

この分析により，エラーメッセージ自動解説の有用性を調査する．またプログラミング初学者が発生させるエラーの特徴についても明らかにすることを目指す．

2 章では本研究の技術的背景および関連研究について述べる．3 章では使用したデータセットと事前分析について説明する．4 章では本研究で導入するエラーメッセージ自動解説について説明する．5 章で各調査課題の動機，方法，結果について報告し，6 章で考察を行う．7 章で妥当性への脅威について述べる．最後に 8 章でまとめと今後の展望について述べる．

2. 背景

本研究では，対象言語として Python を採用した．理由としては Python の構文は比較的簡単であり，プログラマーはより少ないコード数で処理を記述することができる [9]．そのため，Python を，最初に学ぶべきプログラミング言

[1] 奈良先端科学技術大学院大学
Nara Institute of Science and Technology, Ikoma, Nara 630–0192, Japan
[a)] yamato.yusuke.yy7@is.naist.jp
[b)] ishio@is.naist.jp
[c)] k.shimari@is.naist.jp
[d)] matumoto@is.naist.jp

語であると主張する研究者も多い [14]．またデータ分析やグラフの作成に役立つライブラリが豊富であることから，近年注目されている言語である．今後学び始める人も多いと考えられるため Python を対象とした．

以下の節で Python のエラーメッセージについて，そして本研究に関連する既存研究について解説する．

2.1 エラーメッセージ

Python のエラーメッセージには大きく二つの種類があり，「構文エラー (SyntaxError)」と「例外」がある．構文エラーは，書いたコードが Python の構文として正しくない場合に発生するエラーである．コロンの付け忘れや，括弧の閉じ忘れなどがある場合に発生する．一方で例外は，構文は正しいがプログラム実行中に発生するエラーである．NameError や TypeError などが含まれる．エラーメッセージは通常英語で出力される．

2.2 関連研究

学生にフィードバックを行う研究について紹介する．Ahmed らはプログラミングのスキル習得と実践を支援するために，コンパイルエラーを自動修正するツール TEGCER についての研究を行っている [16]．また，堤らは AR を用いたプログラミング教育支援システムの研究を行っている [19]．AR によってプログラムのエラーの原因となる部分を赤い文字で表示し，教授者にフィードバックを行うというものであるが，学習者への効果についての細かい分析は行われていない．Bob らは，ヒートマップを用いてプログラムのエラーの原因となっている可能性が高い箇所を可視化する研究を行った [4]．その結果，与えられた課題を達成するまでの時間が短縮されている．Fu らは，C 言語のプログラミング演習において，どの種類のコンパイルエラーが生じたかをまとめている [5]．

また，プログラミング授業での質問についての研究についてもいくつか研究が行われており，プログラミング経験が浅い学生は，質問自体を考えることができないのではないかという仮説のもとで，学生に対してプログラムの動作を質問することで理解を促す仕組みが考えられている [6], [7]．

3. データ収集

本章では，本研究で利用したデータセットについて説明する．本研究では，著者らの大学で実施した「プログラミング演習」という授業で得たデータを利用した．2021 年度および 2022 年度にプログラミング演習を受講した学生（2021 年度 78 名，2022 年度 94 名）とティーチングアシスタント（2021 年度 6 名，2022 年度 8 名）計 186 名の次のデータを収集した．

表 1: 授業回ごとの学習テーマ

授業回	テーマ
1	Python 言語の概要，変数とデータ型
2	コレクション
3	条件分岐
4	繰り返し
5	関数
6	オブジェクト
7	モジュール
8	Python を取り巻く世界

- チャットデータ
- 実行したコード
- 課題の一覧

また上記のデータには下記の情報を含む．

- 学生 ID
- 課題 ID
- プログラム実行時刻
- 入出力情報
- 発生したエラー
- プログラム提出時刻
- 質問内容

なお，学生からは授業開始前に，プログラムおよび質問内容の研究への利用の許諾について質問しており，明示的に許諾を得られた学生のデータのみを収集している．またコードを記述する形式の課題がない授業回については分析の対象外とした．

3.1 プログラミング演習

奈良先端科学技術大学院大学では，プログラミング初学者を対象に「プログラミング演習」という授業が実施されている．この科目は，情報科学分野以外の出身で，博士前期課程で新たに情報科学を専攻したい学生向けに，Python によるデータ分析のためのプログラムの書き方を教えるとともに，プログラミングの考え方を理解してもらうことを目的としている．表 1 のテーマに沿って授業は全 8 回で構成されており， 学生は各授業回で練習問題と課題を与えられる． 授業の前半では教員が Python についての解説を行い， 学生はそれを基に練習問題に取り組む．授業後半は演習の時間で，各自で課題に取り組み， わからないことがあればチャットでティーチングアシスタントに質問するという流れで行う．授業時間中に解ききれなかった課題については期限日までに終わらせる必要がある．演習には C2Room[*1] というオンライン授業システムを使用している．本研究では，4 章で紹介する提案手法をプログラミング演習に導入する．

[*1] https://c2room.jp/

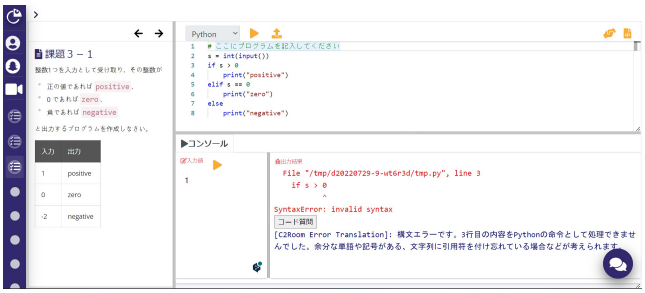

図 1: 授業で使用するオンライン授業システム C2Room

3.2 オンライン授業システム

本研究では，オンライン授業システムである C2Room を使用した．このシステムを使うことで，学生は場所を選ばずに個人の端末から授業に参加できる．学生には図 1 のような画面が表示される．左側に取り組んでいる問題，右上の枠にプログラム，右下の枠に出力結果が表示される．エラーが発生した場合は，右下に赤い文字でエラーメッセージが表示される．何かわからないことがあった場合は，右下の吹き出しマークを押すとチャット画面が開き，ティーチングアシスタントに質問することができる．学生が実行ボタン（プログラムが書かれた欄の上にあるオレンジの三角形のボタン）を押すたびに，課題 ID，学生 ID，実行したコード，入出力情報が保存される．また提出ボタン（実行ボタンの右にあるボタン）を押すことで，学生は完成させたコードをティーチングアシスタントに提出することができる．

4. 提案手法

2.2 節で紹介したように，プログラミング学習者が発生させたエラーを自動的に修正するツールはいくつか存在する [1], [10]．しかしながら，自動的なエラーの修正はプログラミング初学者の，エラーを自力で修正する能力を成長させない可能性がある．そこで本手法では，学生が発生させたエラーの解説文を提示する．解説文には，学生が発生させたエラーの原因と，修正のためのヒントが含まれている．本章では，エラーメッセージ自動解説の作成方法と仕組みについて説明する．

4.1 エラーメッセージ自動解説の作成手順

まず，エラーメッセージ自動解説を作成するにあたり，学生が実際にどのようなエラーに遭遇しているのかを調べる必要があると考えた．そこで 2021 年度のプログラミング演習で，学生 78 人から取得したデータを分析した．全 8 回の授業で学生が発生させたエラーの数を集計し，次にどのようなコードの誤りがそれぞれのエラーの原因になるのかを調べた．実際に学生が書いたコードを目視で分析し，頻度の高い誤りのパターンを調査した．学生の誤りのパターンを基に，エラーの原因とエラー解消のヒントを含む解説文を 74 個作成した．

```
1 k = 10
2 k.append(5)
```

Listing 1: コード例

```
1 Traceback (most recent call last):
2   File "c:\users\yamat\errtest.py",line 2, in <module>
3     k.append(5)
4 AttributeError: 'int' object has no attribute 'append'
```

Listing 2: エラーメッセージ例

```
1 Traceback (most recent call last):
2   File [実行ファイル名], line [行数], in <module>
3     [エラーの原因となっているプログラムの一部]
4 [エラー名]
```

Listing 3: エラーメッセージの形式

4.2 エラーメッセージ自動解説の仕組みと使用方法

エラーメッセージが自動で出力される仕組みと，使用方法について説明する．

4.2.1 仕組み

エラーメッセージを学生に提示する仕組みについて説明する．あらかじめ通常の英語のエラーメッセージとそれに対応する解説文がセットになっているリストを作成した．表 3 に出力された英語のエラーメッセージと対応する解説文の例を示す．本手法では表 3 で示したような解説文を提示するために正規表現を使用した．エラーが発生した際に，英語のエラーメッセージと同じ文字列をリストから検索し，対応する解説文を表示する．

具体的に説明すると，通常はコード 1 のようなプログラムを実行した場合，エラーメッセージ例 2 のような形式でエラーが発生する．この例では，変数 k は append() という属性を持ち合わせていないため，エラーが発生している．このように，エラーが出力される形式は決まっている．構造はエラーメッセージ形式 3 のようになっている．

[ ] 内の部分は，実行プログラムによって変化する部分である．その中でも，行数，エラーの原因となっているプログラムの一部，エラー名は解説文の出力に必要であるため，必要な情報が取得できるような正規表現のルールを作成した．エラー名は，リストから対応する解説文を取得する際に使用する．最後にエラーメッセージと作成した正規表現の例を表 2 に示す．

4.2.2 使用方法

ユーザーの使用方法について説明する．ユーザーがエラーを発生させると，図 1 の赤字部分のように英語のメッ

表 2: 発生エラーと正規表現の例

エラー	正規表現の例
IndexError	^IndexError: ((list)|(tuple)) index out of range
NameError	^NameError: name ‘(?P<NAME>[^’]+)’ is not defined

セージが表示される．その英語のメッセージの下に解説文表示ボタンが配置されている．ユーザーはエラーの修正に行き詰まった場合，ボタンを押すことでエラー解説文を見ることができる．解説文は図 1 のように青字で表示される．

4.2.3 表示方法

前述したように，エラーメッセージ自動解説では画面内のボタンを押すことで解説文をエラーメッセージの下に青字で表示する．本手法を導入する上で解説文をどのように学生に提示するかを考えた．

1: エラーメッセージと解説文のリストを学生に配布する
2: 別のページに遷移させて解説文を表示する

上記のような案を他に考えたが，1 の案では，学生が該当するエラー解説を見つけることに時間がかかる問題がある．また学生によって解説文をカスタマイズすることができない．2 の案では 2 つの画面を同時に見る必要があるため，学生が煩わしく感じてしまうと考えた．そこで，学生がより容易に解説文を閲覧できるようにするため，エラーメッセージの真下に表示するようにした．また，エラーが発生したら解説文を表示する形式の場合，解説文を必要としない人の画面にも表示されてしまう．そのため学生が必要なタイミングで解説文を見ることができるように前述した表示方法を採用した．

5. 評価実験

エラーメッセージ自動解説の有効性の評価項目として，以下の RQ を設定した．

- RQ1. エラーメッセージ自動解説は学生から必要とされているか
- RQ2. エラーメッセージ自動解説は個人に対して効果があるか
- RQ3. エラーメッセージ自動解説はクラス全体に影響を与えるか

5.1 RQ1. エラーメッセージ自動解説は学生から必要とされているか

本節では，エラーメッセージ自動解説が学生から必要とされているかを， RQ1.1 どのくらいエラーメッセージ自動解説は使われたか，RQ1.2 学生の経験度によって解説が求められるエラーに違いはあるか，の 2 つの観点から調査する．

表 3: エラーメッセージ自動解説の具体例

発生エラー	表示される解説文
EOFError	「入力」欄のデータが不足しています。n 行目の input() で入力データを 1 行読み込もうとしましたが、「入力」欄に指定されたデータの終わり（End-Of-File 略して EOF）に到達してしまったので、エラーとして報告されています。「入力」欄にデータを追加してください。なお、input() は 1 回で 1 行ずつ読まれていきますので、複数回 input() を実行する場合は、その回数と同じ行数の入力が必要です。
TabError	Python のプログラムではインデント（字下げ）によって構造を表現しますが、スペースキーで入力される 1 文字ずつの空白と Tab キーで行われる字下げは、1 つのプログラムではどちらか一方に統一する必要があります。現在は 2 つの方法が混ざった状態になっていますので、どちらか一方に統一してください。
未対応の場合	解説機能が対応していないエラーメッセージでした。TA に質問してください。

5.1.1 RQ1.1. どのくらいエラーメッセージ自動解説は使われたか

RQ1.1 では，プログラミング演習を受講した学生がエラーメッセージ自動解説をどの程度使用したかを調査した．また，経験度が低い学生の方がエラーメッセージ自動解説をより多く使用しているのではないかと考えた．そこで学生の経験度によって使用回数に違いはあるのかについても調べた．

実験方法

2022 年度プログラミング演習を受講した学生の総人数とアクティブユーザー数を，経験度別に集計する．アクティブユーザー数とは，課題に取り組むにあたり，1 回以上エラーメッセージ自動解説を使用した学生の人数である．経験度は，初回の授業で行ったアンケート結果に基づいて振り分けた．アンケートでは，ティーチングアシスタントのサポートがどの程度必要かという問いに対し，

A：いらないと思う
B：少し必要と思う
C：大いに必要と思う

の中から学生に選択するように促した．経験度の高い学生ほど，ティーチングアシスタントのサポートが必要ないと考え，

A：経験度が高い
B：経験度が中
C：経験度が低い

として分類を行った．

表 4: 経験度別の人数とアクティブユーザー数

	総人数	ユーザー数
A	24	13
B	41	15
C	24	15
合計	89	43

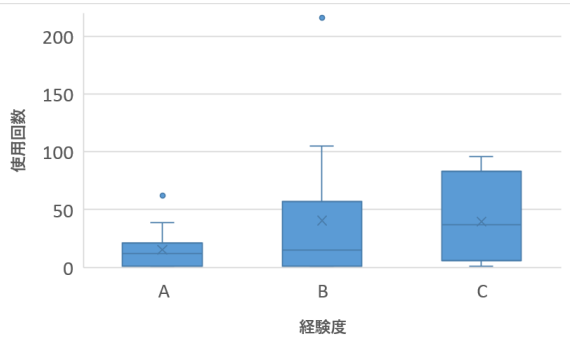

図 2: 経験度別使用回数の分布

実験結果

分析結果を表 4，図 2 に示す．エラーメッセージ自動解説は授業全体を通して合計 1399 回使用された．表 4 では，2022 年度プログラミング演習を受講した学生の総人数とアクティブユーザー数を，経験度別に示した．その結果，89 人の学生のうち，43 人がエラーメッセージ自動解説を使用していることがわかった．また，経験度別の使用回数は図 2 のような分布になった．経験度によって使用回数に有意な差があるかどうかを検証するため以下の帰無仮説に基づき，一元配置分散分析を行った．

H_{01}: 経験度によってエラーメッセージ自動解説の使用回数に有意な差はない．

その結果，$p = 0.2187$ $(p > 0.05)$ となった．したがって，経験度によってエラーメッセージ自動解説の使用回数に有意な差はないことが確認できた．

5.1.2 RQ1.2 経験度によって解説が求められるエラーに違いはあるか

RQ1.2 では，経験度によって躓くエラーに違いがあるのではないかと考えた．そこで経験度によって解説を求めたエラーについて調査する．

実験方法

2022 年度の，プログラムを実行した際に発生したエラーに着目し，経験度ごとに分類を行った．またエラーの種類にも着目し，どの種類のエラーでの解説文使用数が多いのかについて調べる．

実験結果

エラーメッセージ自動解説別各エラーでの解説文使用回数を表 5 に示す．A，B，C 全ての経験度において，TypeError についての使用が最も多いことがわかった．また，

表 5: 経験度別各エラーでの解説文使用回数

	A(13 人)	B(15 人)	C(15 人)
TypeError	76	176	160
SyntaxError	28	77	124
NameError	7	92	71
EOFError	22	76	51
AttributeError	17	50	51
TabError	20	23	33
IndentationError	3	25	42
IndexError	5	41	15
ValueError	11	21	13
ZeroDivisionError	4	9	13
UnboundLocalError	3	3	8
合計	196	593	581

SyntaxError は，経験度が A，B の学生に比べ，C の学生の使用が非常に多いことが確認できた．SyntaxError は Python の構文として誤りがある場合に発生するエラーであるため，経験度の低い学生ほど自力での解決が難しく，解説機能を利用したのではないかと考えられる．

以上の結果から，エラーメッセージ自動解説は学習者の経験度によらず一定の需要があることが確認された．またエラーの種類によっては，経験度によるエラーメッセージ自動解説の使用回数に違いが見られた．

5.2 RQ2. エラーメッセージ自動解説は個人に対して効果があるか

本節では，エラーメッセージ自動解説の学生個人に対する影響について調査を行う．エラーメッセージ自動解説を使用することで学生は発生したエラーの原因が分かり，直後の実行でプログラムの実行を成功させる可能性が高い．そこで，学生がエラーメッセージ自動解説を使用した直後の実行で，有意にエラーが解消できているかによって，学生個人に対する影響を調べる．調査方法は Ahmed らの研究と同じ基準を採用した．図 3 に示すように，エラーメッセージ自動解説使用直後にプログラムを実行した際に，エラーが出力されていない，つまり実行に成功していればsuccess とした [17]．そのため，エラーメッセージ自動解説使用直後に別のエラーが発生した場合でも，実行が成功していないため，failure としている．

実験方法

調査には，2022 年度実施のプログラミング演習で収集した，実行結果データを使用した．フィッシャーの正確確率検定により有意性を評価するため，表 6 のようなクロス集計表を作成した．表の項目について説明する．縦軸の“使用直後”はエラーが発生し，次の実行を行うまでにエラーメッセージ自動解説を使用している場合である．“使用時以外”は，エラーが発生した後，エラーメッセージ自

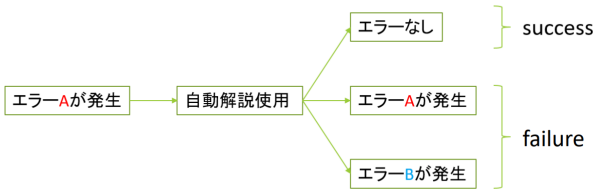

図 3: 実行データの分類方法

表 6: 解説文表示直後の実行結果

	success	failure
使用直後	446	911
使用時以外	5,824	10,712
合計	6,270	11,623

動解説を使用せずに次の実行を行った場合である．横軸の“success”はエラーが発生した次の実行が成功している場合である．“failure”はエラーが発生した次の実行でもエラーが解消されていない場合である．

実験結果

エラーメッセージ自動解説を使用した場合はエラーを有意に解消できているかを検証するため，下記の帰無仮説に基づき，フィッシャーの正確確率検定（片側検定）を行った．

H_{02}: エラーメッセージ自動解説使用の有無は直後の実行結果に影響しない．

その結果，$p = 0.9626$ $(p > 0.05)$ となり，有意な差は見られなかった．したがって，エラーメッセージ自動解説の使用は直後の実行結果に影響しないことが確認できた．

5.3 RQ3. エラーメッセージ自動解説はクラス全体に影響を与えるか

本節では，エラーメッセージ自動解説のクラス全体に対する影響を調査する．具体的には，エラーメッセージ自動解説を使った人と使ってない人で，学生がティーチングアシスタントへ授業中に行った質問の数に有意な差はあるか，すなわちエラー解説の導入により，全質問に占めるエラーに関する質問の割合が有意に減少したかを調べる．

実験方法

まず学生からの質問を図 4 のように，課題（Python）に関するものとその他の質問に授業の進行に関するものに目視で分類し，課題（Python）に関する質問のみを取り出した．その他の質問には，授業の進行に関するもの，システムの利用法に関するものが含まれる．取り出した質問を，さらにエラーに関する質問とそれ以外の質問に分類し，エラーに関する質問はどの程度減少しているか分析を行った．分類については，まずチャット内に「エラー」という

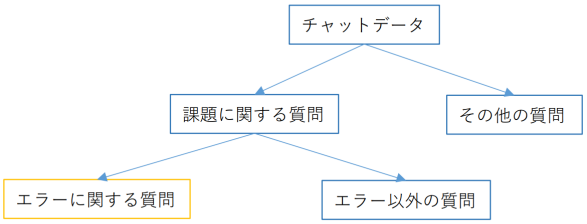

図 4: チャットデータの分類方法

表 7: Python に関する質問の分類結果

	2021		2022	
エラーに関する質問	20	(16%)	16	(9%)
エラー以外の質問	105	(84%)	162	(91%)
合計	125	(100%)	178	(100%)

文字列を含む質問や，エラーメッセージをそのまま貼り付けている質問を抽出した．その後，文脈から判断して，エラーの修正に躓いているような質問は，エラーに関する質問として分類した．以下に質問の具体例を示す．

- エラーに関する質問の例：
  「elif x=0 のところでエラーになるのですが，Python の文法として正しくないのでしょうか？」
  「自分が提出した問 4-3 ですが TabError: inconsistent use of tabs and spaces in indentation というものが発生してしまいます. なぜか分かりますでしょうか」
- エラー以外の質問の例：
  「ディクショナリの中にさらにディクショナリがある時の、キーを取得するにはどうすれば良いでしょうか。」

上記の例のように，2021 年度と 2022 年度のチャットデータを分類した．

実験結果

表 7 に，エラーに関する質問とエラー以外の質問の個数を整理したものを示す．エラーに関する質問は，2021 年度は 78 人中 13 人から 1〜3 回，合計で 20 回であった．2022 年度は 94 名のうち 8 名から 1 回ずつ、1 名から 8 回の質問があり，合計 16 回であった．合計に占めるエラーに関する質問の割合は減少している．また, エラー解説機能は合計 1,563 回利用された．表 7 の結果に対し，以下の帰無仮説を設定した．

H_{03}: エラーメッセージ自動解説 2021 年度と 2022 年度でエラーに関する質問の割合は変化していない．

フィッシャーの正確確率検定（片側検定）を行ったところ，$p = 0.04769$ $(p < 0.05)$ となった．したがって，エラーメッセージ自動解説の導入により，学生からのエラーに関する問い合わせが有意に減少したことがわかった．

6. 考察

本章では，本研究の結果から得られた知見について考察する．

エラーメッセージの解説は，経験度に関わらず一定の需要がある

RQ1 の結果から，エラーメッセージ自動解説は学生から合計 1,399 回使用されており，一定の需要があることが確認できた．また，図 2 から，経験度に関係なくエラーメッセージ自動解説を使用していることがわかった．さらに，表 5 から，エラーの種類によって解説文の使用回数が大きく異なることが読み取れる．これらの結果から，エラーメッセージの自動解説は経験度に関わらず学生から必要とされていることがわかった．したがって，今後ユーザーの使用目的等を細かく分析し，ユーザーの使用目的に合うように解説文を洗練することで，プログラミング学習の助けになることが期待される．

エラーメッセージ自動解説は使用直後の実行に影響しないが，エラーに関する質問数の割合には影響を与える

RQ2 の結果からエラーメッセージ自動解説使用直後の実行では，使用していない場合と比較して実行結果に統計的に有意な差がないことがわかった．また RQ3 の結果からエラーメッセージ自動解説の導入により，エラーに関する質問数の割合が減少していることが統計的に確認できた．これらの結果からエラーメッセージ自動解説は学習者個人対して，本研究で調査した項目では統計的に有意な影響を与えないことがわかった．一方クラス全体に対しては有意に影響を与えることが確認できた．したがって，エラーの修正時間や書かれたコードの内容等のさらなる分析によりエラーメッセージ自動解説の効果の解明が期待される．

7. 妥当性への脅威

本節では，本研究に含まれる妥当性への脅威について述べる．本研究には構成概念妥当性，内的妥当性，外的妥当性への脅威がある．

構成概念妥当性

本研究では，エラーメッセージ自動解説の導入により，エラーに躓いた学生のエラー修正の支援を行った．既存研究では，エラーに躓いた学生は外部リソースの参照やプログラムの状態についての検索等を行うことが示されている [11]．本研究で対象としたプログラミング演習では外部リソースの利用を制限していないため，エラーを修正する際にエラーメッセージ自動解説だけでなく，外部のサイトや参考書を参照した可能性がある．今後は，そういったエラーメッセージ自動解説以外のリソースの影響も考慮しながら分析を行う必要がある．

内的妥当性

内的妥当性の脅威としては 2 つある，一つ目は RQ2 において，エラーメッセージ自動解説を使用した次の実行でエラーが出力されているか否かで提案手法の効果を分析したことが挙げられる．エラーの修正を試みる際に，一度エラーの原因となっている箇所を省いた状態で実行した可能性があることが妥当性の脅威となる．ただし，既存研究 [17] でも提案手法の有用性を測るために同様の方法で分析を行っているため，前述した評価方法は現段階においては最適であると考える．

二つ目は RQ3 において，学生からの質問回数の偏りである．2022 年度のエラーに関する質問についての分析を行った際に，一人で多くの質問をしている学生が存在した．そのような極端な値の取り扱い方により，異なる分析結果が出る可能性があることが二つ目の内的妥当性の脅威である．

外的妥当性

外的妥当性の脅威は 2 つある．一つ目は，本研究では本学の学生を対象とした授業にエラーメッセージ自動解説を導入し分析を行ったため，分析結果は本学の来年度以降の授業には一般化できると考える．しかしながら本学以外の授業に対しては，学生の特徴や学生に課される問題の内容等が異なるため，本研究の分析結果を一般化することはできない．本研究の分析結果を一般化するためには，学生の経験度について，同様のアンケート調査を行う，授業テーマを統一する等の工夫が必要であると考えられる．

二つ目は，授業における学習環境である．本研究で対象としたプログラミング演習では，オンライン授業システムを使用しており，学生はティーチングアシスタントに質問する場合，システム内のチャット機能を使用する．RQ3 ではチャットの履歴データを分析しているが，対面形式の授業では分析結果を一般化できない可能性がある．対面形式の授業では，同じ授業を受講する他の学生，先生とコミュニケーションがとりやすいため，質問の回数や頻度は大きく異なる可能性が高い．RQ3 の分析結果を一般化するためには，授業の形式をそろえる必要がある．

8. おわりに

本研究では，プログラミング初学者のエラーを正しく修正する力を向上させることを目的として，エラーメッセージ自動解説の作成・導入を行い，その必要性と個人・クラスに与える影響について調査した．その結果，プログラミング演習において学生が発生させやすいエラーの種類や，英語のエラーメッセージの解説文を表示することは，学生の経験度に関わらず一定の需要があることが確認できた．

また，エラーメッセージの自動解説により，エラーに関する質問数の割合には有意な減少が見られた一方で，使用直後の実行結果に対して有意な影響を与えないことが確認できた．この結果は，エラーメッセージの解説の必要性やクラスに与える影響が確認できた一方で，本研究で調査した項目では，学生個人に対して有意な影響はないことを示している．

今後の展望として，学生がエラーの修正にかけた時間がどのように変化したかを経験度別，エラーの種類別に調査する等，学習者への効果に関する更なる分析を行うことで，エラーメッセージ自動解説の効果が顕在化される可能性がある．またエラーが出力されてしまうコードと，それに対して表示された解説文を人手をかけて見比べることで，解説文を改善することが挙げられる．以上のような研究によって，プログラミング教育の場において有用な新しい知見を得られる可能性があると考える．そして本研究での手法が，プログラミング学習の助けになることが期待される．

謝辞

本研究は，JSPS 科研費 JP18H04094，JP20H05706，JP22K21279 の助成を受けたものです．

参考文献

[1] Ahmed, T., Ledesma, N. R. and Devanbu, P.: SynShine: Improved Fixing of Syntax Errors, *IEEE Transactions on Software Engineering*, pp. 1–13 (2022).

[2] Becker, B. A., Denny, P., Pettit, R., Bouchard, D., Bouvier, D. J., Harrington, B., Kamil, A., Karkare, A., McDonald, C., Osera, P.-M., Pearce, J. L. and Prather, J.: Compiler Error Messages Considered Unhelpful: The Landscape of Text-Based Programming Error Message Research, *Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education*, pp. 177–210 (2019).

[3] Denny, P., Luxton-Reilly, A. and Tempero, E.: All syntax errors are not equal, *Proceedings of the 17th ACM annual conference on Innovation and technology in computer science education*, pp. 75–80 (2012).

[4] Edmison, B. and Edwards, S. H.: Turn up the Heat!: Using Heat Maps to Visualize Suspicious Code to Help Students Successfully Complete Programming Problems Faster, *Proceedings of the 2020 IEEE/ACM 42nd International Conference on Software Engineering: Software Engineering Education and Training*, pp. 34–44 (2020).

[5] Fu, X., Yin, C., Shimada, A. and Ogata, H.: Error Log Analysis in C Programming Language Courses, *Proceedings of the 23rd International Conference on Computers in Education. China: Asia-Pacific Society for Computers in Education*, pp. 641–650 (2015).

[6] Henley, A., Ball, J., Klein, B., Rutter, A. and Lee, D.: An Inquisitive Code Editor for Addressing Novice Programmers' Misconceptions of Program Behavior, *Proceedings of the 2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering Education and Training*, pp. 165–170 (2021).

[7] Lehtinen, T., Santos, A. L. and Sorva, J.: Let's Ask Students About Their Programs, Automatically, *Proceedings of the 2021 IEEE/ACM 29th International Conference on Program Comprehension*, pp. 467–475 (2021).

[8] Li, A., Endres, M. and Weimer, W.: Debugging with Stack Overflow: Web Search Behavior in Novice and Expert Programmers, *Proceedings of the 44nd International Conference on Software Engineering: Software Engineering Education and Training*, pp. 69–81 (2022).

[9] Lo, C.-A., Lin, Y.-T. and Wu, C.-C.: Which programming language should students learn first? A comparison of Java and Python, *Proceedings of the 2015 International Conference on Learning and Teaching in Computing and Engineering*, pp. 225–226 (2015).

[10] Mesbah, A., Rice, A., Johnston, E., Glorioso, N. and Aftandilian, E.: DeepDelta: learning to repair compilation errors, *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pp. 925–936 (2019).

[11] Robinson, D., Ernst, N. A., Vargas, E. L. and Storey, M.-A. D.: Error Identification Strategies for Python Jupyter Notebooks, *Proceedings of the 30th IEEE/ACM International Conference on Program Comprehension*, p. 253–263 (2022).

[12] Thiselton, E. and Treude, C.: Enhancing Python Compiler Error Messages via Stack, *Proceedings of the 2019 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, pp. 1–12 (2019).

[13] Traver, V. J.: On Compiler Error Messages: What They Say and What They Mean, *Proceedings of the Advances in Human-Computer Interaction 2010*, pp. 1–26 (2010).

[14] Vujoševic-Janicic, M. and Tošić, D.: The role of programming paradigms in the first programming courses, *The Teaching of Mathematics, vol.XI, no.2*, pp. 63–83 (2008).

[15] Wu, E. H.-K., Lin, C.-H., Ou, Y.-Y., Liu, C.-Z., Wang, W.-K. and Chao, C.-Y.: Advantages and Constraints of a Hybrid Model K-12 E-Learning Assistant Chatbot, *IEEE Access*, Vol. 8, pp. 77788–77801 (2020).

[16] Z.Ahmed, U., Srivastava, N., Sindhgatta, R. and Karkare, A.: Targeted Example Generation for Compilation Errors, *Proceedings of the 34th IEEE/ACM International Conference on Automated Software Engineering*, pp. 327–338 (2019).

[17] Z.Ahmed, U., Srivastava, N., Sindhgatta, R. and Karkare, A.: Characterizing the Pedagogical Benefits of Adaptive Feedback for Compilation Errors by Novice Programmers, *Proceedings of the 42nd International Conference on Software Engineering: Software Engineering Education and Training*, pp. 139–150 (2020).

[18] 鈴木舜也，吉野　孝：質疑応答のお知らせ機能をもつ授業支援チャットボットの提案，2021 年度情報処理学会関西支部 支部大会 講演論文集，pp. 1–4 (2021).

[19] 堤　和三，石居　歩，高野辰之，宮川　治：AR を用いたプログラミング教育支援システム，情報処理学会第 75 回全国大会講演論文集，pp. 671–672 (2013).