総和・最大値を計算するプログラム作成問題の足場かけとしてのParsonsパズル問題の作成と評価

西田 誠幸[1,a)]　植野 俊治[1,b)]

概要：総和や最大値を計算するプログラムに現れる変数を学ぶ機会を与えることを意図してパズル問題を作成し，情報系の大学生向けのプログラミング演習科目で出題した．演習中に収集した学習ログから，パズル問題とプログラム作成問題の解答に要した作業時間を算出し分析したところ，作成したパズル問題が他の難易度が高い問題に比べて作業時間が長くなるという証拠は得られないものの，中程度の作業時間がかかることがわかった．また，パズル問題出題後に総和，最大値を計算するプログラム作成問題 3 問のうちの 2 問について，解答に要した作業時間がパズル問題を出題しなかった過去の年度の同一問題の作業時間に比べ，短縮したことがわかった．このことは，事前に課題として与えたパズル問題が総和，最大値を計算する問題の解答に効果があったことを示唆している．

キーワード：プログラミング教育，繰り返し処理，総和・最大値計算問題，パズル問題．

1. はじめに

プログラミング初学者に出題するプログラム作成問題のうちでも難しいとされるものに，総和や最大値を計算するプログラムを作成する問題（以下，総和・最大値計算問題）がある．この問題は for 文や while 文の構文とともに繰り返し処理の概念を学んだ後に出題されるが，学生は新たに学ぶ構文や繰り返し処理とともに，`sum` や `max` などの名前でしばしば宣言される，総和，最大値を保持する変数の働きを理解した上でプログラムを作成する必要がある．

我々は，総和・最大値計算問題の前に出題する問題として Parsons パズル問題（以下，パズル問題）を作問し，情報系の大学生向けのプログラミング演習科目において出題した．このパズル問題は，総和，最大値を保持する変数の働きについて学ぶ機会を与えることを意図しており，繰り返し処理の授業回以前に出題した．本稿では，総和・最大値計算問題の足場かけとして出題したパズル問題について述べる．また，プログラミング演習科目で収集する学習ログから，パズル問題と総和・最大値計算問題の解答に要した作業時間を算出し，パズル問題の効果について論じる．

[1] 拓殖大学，〒 193-0985, 東京都八王子市館町 815-1
Takushoku University, 815-1, Tate-machi, Hachioji, Tokyo 193–0985, Japan
[a)] snishita@cs.takushoku-u.ac.jp
[b)] sueno@cs.takushoku-u.ac.jp

2. 関連研究

2.1 パズル問題

問題文とともにプログラムの断片の集合を与え，問題文に示された仕様を満たすように断片を選択し並べ替えてプログラムを完成させる問題に Parsons パズル [1]， 短冊形プログラミング問題 [2]，ジクソーコード [3] などがある．これらのパズル形式の問題では，プログラムを構成する可能性がある要素をその断片に固定し，各断片の使用回数を制限することにより，プログラム作成問題に比べて問題空間が小さいという特徴を持つ．Parsons パズルのレビュー論文 [4] によると，Parsons パズルはプログラミング演習の授業の課題として使用され，学習の足場かけとして利用される．また，レビュー対象の論文の 52.9% が高等教育機関の初学者向けプログラミング演習科目における Parsons パズルの使用に関する研究報告であり，21.8% が繰り返し処理を対象とするものであった．

これまでにパズル問題を授業に取り入れることによる学習効果が報告されている．Zhi らは，非情報系の大学生のプログラミング演習科目において，ブロックプログラミングの問題の一部を Parsons パズルに置き換えて出題したところ，学生がパズル問題を解くのに要した時間は，同等のプログラミング問題に比べて半分程度の時間に短縮され，その後の課題においては成績や作業時間の点で生徒の成績が悪くなったことを示唆する証拠は見つからなかったこと

を報告した [5]．また，Harms らは，プログラミング初学者向けの環境として，エラーを最小限に抑えながらユーザを一連のステップに誘導するチュートリアル形式で学習した学生よりも，パズル問題で学習した学生の方が，課題の転移で 26%良い成績を収め，演習の完了にかかる時間は 23%短縮されたを報告した．[6]．これに対して，Haynes と Ericson は，与えられた問題に対して学生が書く一般的なプログラムと異なるような解を持つ Parsons 問題は，プログラム作成問題と比べて解答に要する時間が有意に短くなることはないことを示した [7]．

2.2 変数の役割

我々は総和・最大値計算問題の正解プログラムに含まれる変数の役割に注目してパズル問題を作問する．変数の役割 [8], [9], [10] はプログラムに現れる変数の定型的な使い方を説明する．10 種類の変数の役割が，初心者が書くプログラムに現れる変数の 99%をカバーすることが知られる．総和・最大値を保持する変数の役割は Gatherer と Most-wanted holder である．これら 2 つの役割の定義 [10] を次に示す．

Gatherer 変数は一連の値の 1 つずつを何らかの形で結合するために使用され，こうして蓄積された結果を変数は保持する．

Most-wanted holder 一連の値の中で現れる「最良の値」を保持する変数を表す．

3. パズル問題の作成方針

総和・最大値計算問題は，繰り返し処理を学んだ段階で出題される問題である．例として，1 から 3 までの総和を繰り返し処理で計算する Java プログラム ProgA を図 1 に示す．このプログラムを書くためには，事前に繰り返し処理の構文（この例の場合は for 文の構文）とともに，変数 sum の使い方を知る必要がある．一般に，総和・最大値を計算するプログラムでは，変数の役割 Gatherer と Most-wanted holder にあたる変数を繰り返しの過程で更新することによって目的の計算を実現するが，このような変数は繰り返しの概念を学んでまもない段階で学生に出題される．

次に，繰り返し処理を展開したプログラム ProgB を 図 2 に示す．このプログラムは繰り返しの構文を使用しておらず，学生は変数を学んだ段階で読むことができる．また，変数 sum は引き続き Gatherer の役割を持つため，このようなプログラムを読んだり書いたりすることで，総和・最大値を計算するプログラム作成の足場かけになる可能性がある．変数の単元では整数型の変数の内容に 1 を加えて再代入する文 x=x+1; を学ぶのが標準的である．したがって，ProgB のようなプログラムを読み書きするための基礎はあるが，単なるプログラム作成問題として出題したときに学

```
public class ProgA {
  public static void main(String[] args) {
    int sum = 0;
    for (int i=1; i<=3; i=i+1) {
      sum = sum + i;
    }
  }
  System.out.println(sum);
}
```

図 1 総和を計算するプログラム A

```
public class ProgB {
  public static void main(String[] args) {
    int sum = 0;
    sum = sum + 1;
    sum = sum + 2;
    sum = sum + 3;
    System.out.println(sum);
  }
}
```

図 2 総和を計算するプログラム B

```
public class ProgC {
  public static void main(String[] args) {
    int sum = 1+2+3;
    System.out.println(sum);
  }
}
```

図 3 1+2+3 を計算するプログラム C

生がこのようなプログラムを作成することは稀である．すなわち，「1,2,3 の合計を計算し，その計算結果を sum に代入するプログラム」を作成する問題を学生に出題すると，多くは図 3 のように合計の計算結果を直接変数に代入するプログラムを作成する．このプログラムにおける変数 sum の役割は Gatherer ではなく，一時的な計算結果を保持する Temporary であり，総和・最大計算問題のプログラム作成に対する効果は期待できない．

そこで，本研究ではパズル問題として，繰り返しを学ぶよりも前の段階で ProgB のようなプログラムを作成させる問題を出題する方法をとる．この問題は Gatherer や Most-wanted holder などの役割を持つ変数を使い慣れない学習段階の学生には難易度が引き上げる要因となりうるが，一方でパズル問題はあらかじめ与えられるカードを並べることでプログラムを作成する形式であるため，プログラム作成問題に比べて問題空間が小さく，問題の難易度を引き下げる効果も働く．そこで，Gatherer と Most-wanted holder の役割を持つ変数を使用するプログラムを作成させるパズル問題を，学生が変数を学んだ段階で出題し，その後繰り返しの学習回で出題する総和・最大値計算プログラムの作成の際にパズル問題による効果があるかどうかを調べる．

4. パズル問題の出題と評価

作問したパズル問題を情報系の大学 1 年生向けのプログラミング演習科目で出題した．本節ではこの演習科目と出題したパズル問題，パズル問題出題の評価方法，評価結果について述べる．

4.1 プログラミング演習科目について

作問したパズル問題を拓殖大学工学部情報工学科 2022 年度のプログラミング演習科目「プログラミング I」で出題した．この科目は毎年 9 月から翌年 1 月にかけて 13 回で開講される科目で，その目標は与えられたプログラムの仕様を満たす基本的な Java プログラムを学生が 1 から作成できることである．この科目では，学生はエディタを使用した Java プログラム作成と，ターミナル（コマンドプロンプト）でのプログラムのコンパイル，実行により演習を進める．また，この科目用の Web アプリケーション，プログラミング演習支援システムにプログラムを提出しテストを実施することによりプログラムの正誤を判定し，一つの問題を完了する．この科目で教える Java の構文の範囲は標準出力への表示，変数，標準入力，条件分岐，繰り返し，配列などである．2022 年度の受講生は受講前の 4 月から 7 月にコンピュータリテラシをテーマとする別の演習科目を受講しており，エディタ上でのプログラム作成とターミナルでの Java プログラムのコンパイルと実行を体験的に学習済みである．なお，高校時代にプログラミングを経験済みの学生向けに事前にプログラミングの集中講義を開講しており，この集中講義を合格した学生は「プログラミング I」を受講しなかった．換言すると「プログラミング I」を受講した学生は，その前の期間に Java プログラムのコンパイルと実行を体験的には学習しているものの，プログラミングの経験が少ない学生であった．

「プログラミング I」で出題する問題は，授業前，中，後にそれぞれ出題する予習問題，演習問題と追加問題，復習問題からなる．予習問題はテキストで示される例題プログラムを写経する問題やそのパラメータを変更し実行結果を確認する問題である．また，演習問題と復習問題はプログラムを 1 から作成する問題で，後者は宿題であるため難易度がやや低く設定している．追加問題は授業中の演習問題が全て終わった学生向けのもので，難易度を高く設定している．

「プログラミング I」で出題した問題のうち追加問題を除くものは合計で 106 問で，そのうちプログラム作成問題 91 問に対し，パズル問題は 15 問と比較的少数だった．また，総和・最大値計算問題のプログラム作成の足場かけとして出題したパズル問題は 15 問中 4 問だった．これらのパズル問題を出題したのは，第 3，4，7，10 回の講義だった．これら 4 つの授業回に出題した問題のうち総和・最大の計算に関連する問題を表 1 に示す．表の問題名のうち「Puzzle」で始まる問題はパズル問題，それ以外の問題はプログラム作成問題である．また，表中の「入力」は標準入力を使ってユーザが入力した整数を受け取ることを意味する．問題 Goukei では 整数 n をまず入力し，次に n 回整数を入力するプログラムを作成する問題である．問題 Max2 では最大値が負の数になる場合も想定することを求めた．なお，4 つのパズル問題と Saidai は for 文を使用しないプログラムの作成を想定している．

表 1 総和・最大値計算プログラム作成問題とパズル問題

授業回	問題名	内容
3 変数	Puzzle33	3 つの整数の合計
4 標準入力	Puzzle3b	入力した 3 つの整数の合計
7 if 文	Puzzle6	入力した 3 つの整数の最大
	Saidai	入力した 4 つの整数の最大
10 for 文	Puzzle9	入力した 3 つの整数の合計 ただし for 文を使用しない
	Goukei2	n 個の入力した整数の合計
	Max2	入力した 5 つの整数の最大 ただし for 文を使用すること

本研究では，パズル問題の効果を図るためのベースラインとして，2020 年度「プログラミング I」[*1]の授業結果を用いる．ここで 2020 年度の講義実施状況を説明する．

新型コロナウィルス感染症拡大防止のため，2020 年度「プログラミング I」はリアルタイムオンライン授業の形態で実施された．2020 年度に出題された問題の形式はプログラム作成問題のみであった．2022 年度の第 7 回と第 10 回で出題された問題と，2020 年度の同授業回で出題された問題を確認したところ，それぞれ 7 問，5 問が同一問題であった．これらの問題には，表 1 に示した Saidai, Max2 が含まれる．また，問題 Goukei2 はほぼ同一問題が出題されているが，2020 年度は入力する整数の個数は n ではなく 5 個に固定しており，2022 年度に比べてやや易しい問題であった．

プログラミング I では Web ベースのプログラミング演習支援システムを使用しており，学生の学習状況をシステムが収集する学習ログから把握できるが，対面形式だった 2022 年度と比較して，2020 年度の各問題のプログラム提出率が低かったことがわかっている．このことから 2020 年度は講師や TA の指導が行き届かなかったことが推測される．したがって，2020 年度と 2022 年度の学習ログを用いてパズル問題の効果を分析するには両年度の授業形態の違いを考慮する必要がある．

[*1] 2021 年度の同科目では，総和・最大値計算に関連するパズル問題を部分的に出題した．本研究ではパズル問題を出題しなかった 2020 年度授業結果をベースラインとする．

4.2 出題したパズル問題

パズル問題は「プログラミング I」で使用しているプログラミング演習支援システムの Web ページ上に js-parsons ライブラリ [11] を用いて実装した．Web ページは，問題文，選択肢，解答欄，実行結果で構成される．パズル問題のカードは選択肢と解答欄に配置される．学生には，問題文の仕様通りに動作するように選択肢から適切なカードを選んで解答欄に配置してプログラムを完成させることが求められる．学生がプログラムを提出すると，提出プログラムに対してシステムは実行結果ベースのフィードバックを学生に提供する．

パズル問題の 1 枚のカードには，プログラムの 1 文，または複数の連続する文が印字される．また，選択肢のカードの中にはディストラクタ（正解で使用しないカード）が含まれることがある．さらに，問題によってはいくつかカードが解答欄に配置された状態で出題される．なお，もともと解答欄に配置されるカードはディストラクタではなく，学生はこれらのカードの前後，間に選択肢のカードを配置することによって解答が完了するように設定されている．

総和最大値計算問題の足場かけとして出題したパズル問題 4 つとその模範解答を付録 A.1 に示す．4 つの図は各問題のカードの初期状態を示している．4 つ全ての問題は初期状態でほとんどのカードが配置済みで，学生は追加のカードを選択肢から選んで解答欄に配置済みのカードの間に挿入することによってプログラムを完成させる．

問題 Puzzle33，Puzzle3b, Puzzle9 は総和計算問題を対象として出題した．これらの問題の変数 `s`, `x`, `sum` の役割は Gatherer である．なお，Puzzle33 については，`s=s+678;` と `s+t` の表示文のカードを組み合わせた意図しない別解があることが出題後に判明した．一方，Puzzle6 は最大値計算問題を対象として出題した．このプログラムの変数 `max` の役割は Most-wanted holder である．

4.3 パズル問題出題の評価方法

2020 年度と 2022 年度の「プログラミング I」で収集した学習ログから各年度，各学生，各問題の作業時間を算出し，これを利用して次の 2 つの方法ではパズル問題を評価する．

(1) 2022 年度のパズル問題のうち，表 1 で示した 4 つの問題と，それ以外の問題の作業時間の分布の比較
(2) 2020 年度と 2022 年度に出題したプログラム作成問題のうち，表 1 で示したプログラム作成問題 3 問の作業時間の Z 値の分布の比較

このうち，(1) では総和・最大値計算問題のプログラム作成の足場かけとして出題した 4 題のパズル問題が各授業回に出題する他のパズル問題の中で，解答に時間がかかる難しい問題ではなかったかどうかを明らかにしたい．調査対象とするパズル問題を第 1 回から第 10 回までに出題した 9 問とし，表 1 のパズル問題 4 つとそのほかの 5 つに分けて，それぞれの問題について各学生が問題文を確認し，カードを配置し，提出するまでの作業時間の分布を比較する．作業時間の算出にはプログラミング演習支援システムで収集した学習ログのうち，問題ページの表示，カード移動，保存，実行の時刻を使用する．学生 s が問題 p を解くのに要した作業時間は，学生 s の問題 p に関する学習ログをイベント発生時刻を昇順に並べて一連のイベント発生時刻の差の合計と定義する．ただし，時刻差 5 分と 1 時間を閾値として計算方法を調整する．すなわち時刻差が 1 時間を超える場合はその間のログが正しく取得できなかったと判断して作業時間に算入しない．また，5 分以上 1 時間以下の場合は，学生の休憩等を考慮して 5 分間だけを作業時間に算入する．さらに，学生 s が問題 p のプログラムを完成できなかったときには，演習に取り組まない場合や，逆に比較的長い時間取り組んで諦める場合が想定され，前述の計算方法で作業時間を求めるのは妥当とはいえない．このため，学生 s が問題 p のプログラムを完成できなかったときの作業時間は，問題 p のプログラムを完成した別の学生の作業時間の最大値とする．この方法により算出した 2 種類のパズル問題の作業時間の平均値の違いを，ウェルチの t 検定（有意水準 5%, 対応なし，両側検定）を用いて示す．

また，(2) では，パズル問題を出題しなかった 2020 年度と，パズル問題を出題した 2022 年度に出題した総和と最大値計算プログラム作成問題 3 つの作業時間を比較することによって，パズル問題の学習効果を明らかにしたい．作業時間の算出には，学習ログのうち，問題ページの表示，プログラムファイルの保存，コンパイル，実行，テストを実行した時刻を用いる．作業時間の算出方法は (1) と同様であるが，パズル問題に比べログ記録の頻度が低いことから閾値としては 5 分ではなく，10 分を使用する．すなわち連続するログの時刻差が 10 分以上 1 時間以内の時に，10 分だけを作業時間に算入する．ただし，4.1 節で述べたように，両年度の授業実施形態が異なることから作業時間の分布が大きく異なることが予想される．そこで，第 7 回と第 10 回に出題された問題の中で，注目する 3 問の問題がどの程度解答に時間がかかる難しい問題であったかを測る指標として Z 値を用いる．Z 値（Z スコア）は，個々の作業時間が授業回の作業時間の平均からどの程度離れているかを表す指標である．各授業回の全学生の作業時間の Z 値の平均は 0，標準偏差は 1 となる．第 7 回と第 10 回に両年度で共通して出題された問題に Goukei2 を加えた 13 問を対象として，作業時間の授業回ごとの要約統計量（平均値と標準偏差）を算出しておいて，各作業時間の Z 値をもとめ，その分布の平均値の差を検定する．

表 2 総和・最大値計算問題の足場かけとそれ以外のパズル問題の解答作業時間の要約統計量

パズル問題の種類	N	作業時間 (秒) 中央値	平均値	標準偏差	p 値
総和・最大	532	450.5	697.5	752.70	p<0.01
その他	665	430.0	928.6	1473.64	

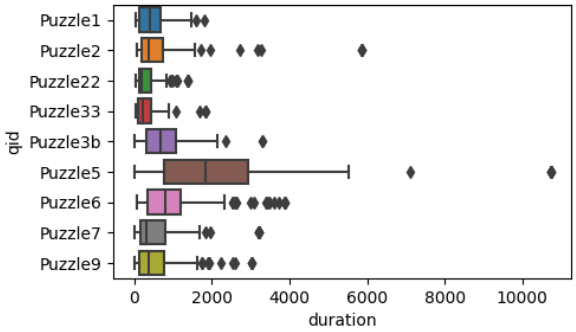

図 4 プログラム作成問題の作業時間の比較

4.4 作業時間の分析

4.5 パズル問題の作業時間の分析

2022 年度パズル問題のうち，総和・最大値計算問題の足場かけとして出題した Puzzle33, Puzzle3b, Puzzle6, Puzzle9 の 4 問とそれ以外の 5 問に分けて，各学生のそれぞれの問題の解答作業時間の要約統計量を表 2 に示す．調査対象の学生は，9 つのパズル問題のいずれかで学習ログを 1 つ以上残している学生 133 名である．表中の N は学生数にそれぞれのグループの問題数 4, 5 の積である．それぞれのグループの平均値は 11 分，15 分で，中央値よりも大きいことから，右に歪んだ分布で，解答に長い時間を要した学生がいることが読み取れる．また，総和・最大値計算問題の足場かけとして出題したパズル問題の方が作業時間の平均値は小さいことがわかった．検定の結果，2 つのグループのパズル問題の作業時間の平均には有意差があることが明らかになった．この結果は，総和・最大値計算問題の足場かけとして出題したパズル問題は他のパズル問題に比べて，解答に時間がかかったとは言えないことを示唆している．

次に各問題ごとの作業時間の分布を図 4 に示す．この図からは，Puzzle5 の作業時間が他の問題の作業時間に比べて大きいことが読み取れる．Puzzle5 は if 文の範囲の問題であり，他のパズル問題とは違って，最初に解答欄に提示されたプログラムから if 文の構造を大きく変える必要がある．したがって，Puzzle5 は他と比べて難易度が高い問題と考えられる．

また，Puzzle5 に続いて作業時間が大きい範囲に分布しているのは Puzzle6, Puzzle3b である．これらのパズルの解答に要した作業時間の平均はそれぞれ約 12 分，16 分であった．これらのことから，総和・最大計算問題の足場かけとして出題した問題は，他のパズル問題のうちで難易度が高い問題ほど難しいとは言えないものの，パズル問題の中では比較的難しい問題になりうることがわかった．

表 3 各授業回のプログラミング作業時間の要約統計量

授業回	年度	N	作業時間 (秒) 中央値	平均	標準偏差
7 if 文	2020	784	1131.5	1729.8	1910.66
	2022	917	901.0	1443.8	1657.98
10 for 文	2020	672	2028.0	3221.2	3378.57
	2022	786	1595.5	2428.8	2408.84

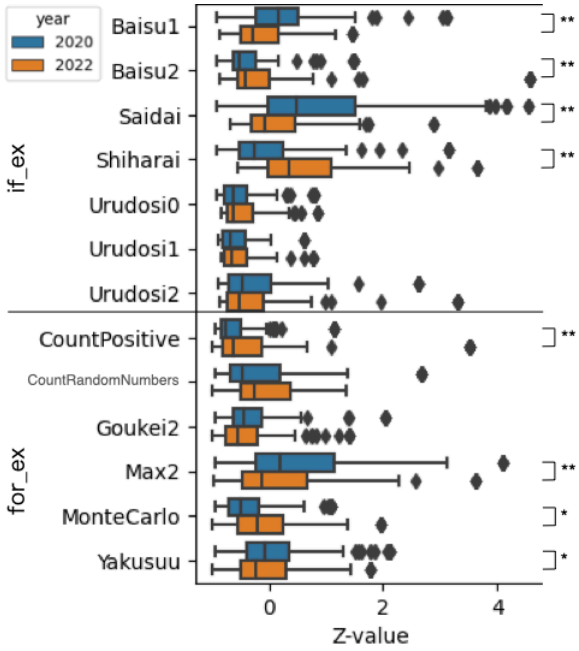

図 5 プログラム作成問題の作業時間の比較

4.6 プログラム作成問題の作業時間の分析

2020 年度と 2022 年度の第 7 回と第 10 回のプログラム作成問題の解答に要した作業時間の要約統計量を表 3 に示す．なお，調査対象の問題の解答の際に収集した学習ログが 1 つ以上記録されている学生を調査対象とした．各年度の調査対象学生の人数はそれぞれ 112, 131 であった．これらと各授業回の問題数（それぞれ 7，6） とのかけ算の結果が表中の N である．

各年度各授業回の作業時間の平均値は中央値に比べて大きく，作業時間の分布はすべて右に歪んでいる．また，2020 年度に比べて 2022 年度の作業時間の中央値と平均値はいずれも小さかった．2020 年度の各授業回 1 問あたりのプログラミング作業時間の平均はそれぞれ約 29 分，54 分で，for 文の作業時間は特に大きい．プログラミングに苦労した学生にとっては演習が大きな負担であったことが推測される．両年度の作業時間の違いの原因は，オンライン授業と対面授業の授業形態の違いにより，2020 年度は学生の状況を講師が把握しづらかったこと，講師の指導が行き届きにくかったことにあったと考えられる．一方，各授業回の中央値はそれぞれ約 19 分，34 分で，平均値より

も 10 から 20 分短い．プログラムが完成しなかった学生の作業時間を完成した学生の作業時間に置き換えたことにより，平均値を引き上げられた可能性がある．

2020 年度と 2022 年度のプログラム作成問題ごとの作業時間の Z 値の分布を図 5 に示す．グラフの横軸は Z 値，縦軸は出題した問題であり上段は if 文演習回，下段は for 文演習回に出題した問題である．Z 値 0 はそれぞれの年度の各授業回で出題した問題のうちプログラム完成に平均的な作業時間を要したことを意味する．また，Z 値が小さいほどその授業回で他の問題に比べて作業時間が短かったこと，逆に Z 値が大きいほど作業時間が長かったことを意味する．

プログラム作成問題のうち，総和，最大を計算する問題は，if 文演習回の Saidai と for 文演習回の Goukei2, Max2 の 3 つであるが，このうち Saidai と Max2 の作業時間の Z 値の平均は 2020 年度よりも 2022 年度の方が小さく 2 つの間に有意差があった．このことは事前に取り組んだパズル問題の効果によるものである可能性がある．一方，Goukei2 の作業時間の Z 値については有意差があるとはいえなかった．Goukei2 は 4.1 節で述べたように 2022 年度の方が 2020 年度に比べ難しい問題であったために作業時間が伸びて，パズル問題の効果が隠れてしまった可能性がある．

5. 本研究の限界

本研究では 2020 年度と 2022 年度の授業環境に大きな違いがあったことから，プログラミングにかかった作業時間の Z 値を比較する方法をとったが，この方法の妥当性は明らかではない．特に，Z 値は各授業回ごとの作業時間の全体に対して 1 つの問題の作業時間の相対的な関係を示す値であるため，隠れた要因によりある授業回で出題された問題の学生全体の作業時間が著しく大きくなると，その授業回の別の Z 値に影響を及ぼす．また，本研究では作業時間の算出対象を両年度で共通して出題した問題に限定したが，それ以外の問題による影響は考慮しなかった．さらに各回の授業冒頭での講師の説明内容の記録が残っていないため，説明内容の違いによる影響を考慮していない．

6. おわりに

本稿では繰り返し処理の授業回で出題される総和や最大値を計算するプログラム作成問題の足場かけとして出題したパズル問題について論じた．このパズル問題は，総和や最大値などを保持する変数の役割に注目して作成したもので，繰り返しの授業回以前に出題した．プログラミング演習で収集した学習ログから，パズル問題とプログラム作成問題の解答にかかった作業時間を算出したところ，足場かけとして出題したパズル問題の作業時間はもっとも難易度が高かった別のパズル問題に比べると短かったもののその他のパズル問題よりも難しい可能性があることがわかった．パズル問題を出題しなかった 2020 年度と出題した 2022 年度の総和，最大値を計算するプログラム作成問題の作業時間の Z 値を比較したところ，3 問中の 2 問で 2022 年度の方が作業時間が短縮されたことがわかった．これは，事前に出題したパズル問題の効果を示唆するものである．

本研究では，問題の解答に要した作業時間を基準に問題の難易度を測ったが，今後の課題として，学生が作成したソースコードを調査することによって，パズル問題の効果の有無を明らかにしたい．例えば，最大値の計算問題で躓きやすいポイントの一つとして，最大値を保持する変数の初期化がある．0 で初期化してしまうと，入力整数が全て負の数であるときに最大値が正しく計算されないが，このような間違いをソースコードから見つけて修正されるまでの時間を見るという方法が考えられる．

また，本研究で示したパズル問題がプログラム作成問題の誘導になっていた可能性があるので，この点も今後明らかにしたい．

参考文献

[1] Parsons, D. and Haden, P.: Parson's Programming Puzzles: A Fun and Effective Learning Tool for First Programming Courses, *Proceedings of the 8th Australasian Conference on Computing Education - Volume 52*, pp. 157–163 (2006).

[2] 上野　真，照井佑季，今村瑠一郎，久野　靖，江木啓訓：短冊型プログラミング問題における解答プロセスに基づく指導支援，マルチメディア，分散協調とモバイルシンポジウム 2021 論文集，Vol. 2021, No. 1, pp. 1451–1458 (2021).

[3] 山口　琢，中村陽太，大場みち子：プログラム・コードの並べ替えパズルにおける正解との距離の変化，情報教育シンポジウム論文集，Vol. 2020, pp. 47–53 (2020).

[4] Ericson, B. J., Denny, P., Prather, J., Duran, R., Hellas, A., Leinonen, J., Miller, C. S., Morrison, B. B., Pearce, J. L. and Rodger, S. H.: Parsons Problems and Beyond: Systematic Literature Review and Empirical Study Designs, *Proceedings of the 2022 Working Group Reports on Innovation and Technology in Computer Science Education*, pp. 191–234 (online), DOI: 10.1145/3571785.3574127 (2022).

[5] Zhi, R., Chi, M., Barnes, T. and Price, T. W.: Evaluating the Effectiveness of Parsons Problems for Block-Based Programming, *Proceedings of the 2019 ACM Conference on International Computing Education Research*, pp. 51–59 (online), DOI: 10.1145/3291279.3339419 (2019).

[6] Harms, K. J., Rowlett, N. and Kelleher, C.: Enabling independent learning of programming concepts through programming completion puzzles, *2015 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, pp. 271–279 (online), DOI: 10.1109/VLHCC.2015.7357226 (2015).

[7] Haynes, C. C. and Ericson, B. J.: Problem-Solving Efficiency and Cognitive Load for Adaptive Parsons Problems vs. Writing the Equivalent Code, *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, New York, NY, USA,

Association for Computing Machinery, (online), DOI: 10.1145/3411764.3445292 (2021).
[8] Sajaniemi, J.: An empirical analysis of roles of variables in novice-level procedural programs, *Proceedings IEEE 2002 Symposia on Human Centric Computing Languages and Environments*, pp. 37–39 (online), DOI: 10.1109/HCC.2002.1046340 (2002).
[9] Kuittinen, M. and Sajaniemi, J.: Teaching Roles of Variables in Elementary Programming Courses, *SIGCSE Bull.*, Vol. 36, No. 3, pp. 57–61 (online), DOI: 10.1145/1026487.1008014 (2004).
[10] Sorva, J., Karavirta, V. and Korhonen, A.: Roles of variables in teaching, *Journal of Information Technology Education: Research*, Vol. 6, No. 1, pp. 407–423 (2007).
[11] Ihantola, P. and Karavirta, V.: Two-dimensional parson's puzzles: The concept, tools, and first observations, *Journal of Information Technology Education. Innovations in Practice*, Vol. 10, p. 119 (2011).

付　　録

A.1 総和・最大値計算プログラムの足場として作問したパズル問題

選択肢

```
int s = 123;
int t = 45;
int u = 678;
System.out.println("123+45+678=" + (s+t));
System.out.println("123+45+678=" + s);
s = s + 45;
s = s + 678;
```

解答欄

```
public class Puzzle33 {
  public static void main(String[] args) {
  }
}
```

図 A·1 第 3 回出題 Puzzle33: 3 つの整数 123, 45, 678 の合計を計算して結果を表示するプログラムを完成させてください.

```
public class Puzzle33 {
  public static void main(String[] args) {
    int s = 123;
    s = s + 45;
    s = s + 678;
    System.out.println("123+45+678="+s);
  }
}
```

図 A·2 Puzzle33 の模範解答: 別解に注意

選択肢

```
x = n + n + n;
x = x + n;
x = x + n;
x = x + n;
```

解答欄

```
public class Puzzle3b {
  public static void main(String[] args) {
    String s;
    int n;
    int x = 0;
    s = System.console().readLine("1つ目の整数: ");
    n = Integer.parseInt(s);
    s = System.console().readLine("2つ目の整数: ");
    n = Integer.parseInt(s);
    s = System.console().readLine("3つ目の整数: ");
    n = Integer.parseInt(s);
    System.out.println("合計: " + x);
  }
}
```

図 A·3 第 4 回出題 Puzzle3b: 3 つの整数を入力して合計を計算して表示するプログラム Puzzle3b を作成せよ.

```
public class Puzzle3b {
  public static void main(String[] args) {
    String s;
    int n, x=0;
    s = system.console().readLine();
    n = Integer.parseInt(s);
    x = x + n;
    s = system.console().readLine();
    n = Integer.parseInt(s);
    x = x + n;
    System.out.println("合計: "+x);
  }
}
```

図 A·4 Puzzle3b の模範解答

選択肢

```
max = n;
max = n;
if (max < n) {
if (max < n) {
}
}
```

解答欄

```
public class Puzzle6 {
    public static void main(String[] args) {
        String s;
        int n, max;
        s = System.console().readLine("整数を入力: ");
        max = Integer.parseInt(s);
        s = System.console().readLine("整数を入力: ");
        n = Integer.parseInt(s);
        s = System.console().readLine("整数を入力: ");
        n = Integer.parseInt(s);
        System.out.println("最大値: " + max);
    }
}
```

図 A·5 第 7 回出題 Puzzle6: 3 つの整数を入力し，その最大値 max を表示するプログラム Puzzle6.java を完成させよ．

```
public class Puzzle6 {
  public static void main(String[] args) {
    String s;
    int n, max;
    s = system.console().readLine();
    n = Integer.parseInt(s);
    max = n;
    s = system.console().readLine();
    n = Integer.parseInt(s);
    if (max < n) {
      max = n;
    }
    s = system.console().readLine();
    n = Integer.parseInt(s);
    if (max < n) {
      max = n;
    }
    System.out.println("最大値: "+max);
  }
}
```

図 A·6 Puzzle6 の模範解答

選択肢

```
        s = System.console().readLine("整数を入力: ");
        int m = Integer.parseInt(s);
        n = Integer.parseInt(s);
        sum = m;
        sum = sum + n;
```

解答欄

```
public class Puzzle9 {
  public static void main(String[] args) {
      int sum = 0;
      String s = System.console().readLine("整数を入力: ");
      int n = Integer.parseInt(s);
      sum = sum + n;
      s = System.console().readLine("整数を入力: ");
      n = Integer.parseInt(s);
      sum = sum + n;
      System.out.println("合計: " + sum);
  }
}
```

図 A·7 第 10 回出題 Puzzle9: 次のプログラムは，2 つの整数を入力し，その合計を計算して表示するプログラム Puzzle9.java である．このプログラムを，3 つの整数の合計を計算するように修正せよ．

```
public class Puzzle9 {
  public static void main(String[] args) {
    int sum = 0;
    String s = System.console().readLine();
    n = Integer.parseInt(s);
    sum = sum + n;
    s = System.console().readLine();
    n = Integer.parseInt(s);
    sum = sum + n;
    s = System.console().readLine();
    n = Integer.parseInt(s);
    sum = sum + n;
    System.out.println("合計: "+max);
  }
}
```

図 A·8 Puzzle9 の模範解答