広島工業大学紀要教育編
第23巻（2024）17-22
論　　文

じゃんけんゲームの段階的改良に基づく C言語プログラミング演習教材とその教育効果

松岡　雷士*・佐々木　克実**

（令和5年11月24日受付）

Teaching material for C language programming practice based on stepwise improvement of the Janken game and its educational effect

Leo MATSUOKA and Katsumi SASAKI

（Received November 24, 2023）

Abstract

C言語の基礎的な諸概念の使い方を学ぶために開発した演習教材、および、それを用いた授業実践について報告する。本教材は単純なじゃんけんゲームを題材とし、各回で学んだ内容を使って、ゲームのプログラムを段階的に改良していく構成となっている。その過程において、受講者は配列、関数、ポインタ、構造体などの概念とそれらの使い方を体験的に学習できる。受講者アンケートの結果に基づけば、各項目に対する理解度と教材の有効性について、全体として高い評価が得られている。特に、一般的に難解な「ポインタの基本的な使い方」について、半数以上の受講者に理解の実感を与えることに成功している。プログラミングに興味が持てた学生や得意科目と自覚する学生を増やすことにも成功しており、本教材を用いた授業の高い教育効果が示される結果を得ることができた。

Key Words: Computer programming education, Pointer, Pass by pointer, Effective exercise

1．はじめに

プログラミングは創造力と論理的思考を結びつける訓練として、あらゆる科目の基礎となるべき科目であり、また、直接的・間接的な幅広い応用範囲を持つ科目でもある。小学校でもプログラミングに関する授業が開始され、小学生がScratch（Scratch Foundation）などを通してプログラミングに触れる機会も着実に増えている。しかしながら、著者が見る限り、少なくとも地方私立大学においては初年次でプログラムをゼロから書ける大学生は皆無に等しく、特に変数のスコープやポインタなどの抽象的な概念に理解が及ぶ学生は非常に少ない。大学におけるプログラミング教育にはまだまだ工夫が必要であるように思える。

受講者の多くが授業を通してプログラミングを理解できない、もしくは、面白さが全く分からないというケースは残念ながら多くの大学で散見される。失敗の原因はいくつかあるが、まず多いのはプログラミング教育を「言語文法の教育」と同義にとらえてしまっているケースである。プログラミングを思い通りに組立てるスキルは文法の知識と同じものではない。例えば、授業で配列などの文法概念を教えた後、それを確かめるための小さなプログラムを学生に逐一作成させて確認を促す授業スタイルは多くの大学で実施されているようである。しかしながら、学生にとってはそのような小さなプログラムは抽象的で意味がわからず、また役に立つものではない。結果として小さなプログラムで文法概念を確かめさせても、学生にはプログラミン

* 広島工業大学工学部電気システム工学科
** 有限会社工房知の匠（広島工業大学非常勤講師）

グの利用価値が全く伝わらないのである。

著者は以前から、学生に言語文法の範囲にないプログラミング力の向上を実感させるため、ゲームの作成を授業に取り入れてきた[1]。これまでにも Fortran や Python を用いたプログラミングの授業において、最低限の文法知識で学生のプログラミング力を一旦完成させることを目標にする授業を構成してきた。このために、for や if 以外の文法は極力使わない段階で「確率を使ったゲーム」を作成する演習を行い、文法に過剰にとらわれない論理的思考力の育成を試みてきた。この試みは概ね好評であり、自由にゲームを改良する課題等においても、手の込んだコードを提出してくる学生を年々増やしていくことに成功していた。

しかしながら、C 言語などの言語を深く学んでいくためには単純な制御構造から一歩踏み込んだ文法の諸概念とその使い方の理解が必要であり、特にポインタや変数のスコープなどの概念を教科書の記述に合わせて明確に理解していくことが重要となる。このためには、ゲーム型教材にも一歩踏み込んだ文法を取り入れる必要が生じる。ゲーム型教材の利点は、抽象概念を現実のイメージに結び付けながら考えることが容易であることであるが、一歩進んだ文法概念を理解する段階においても、イメージと抽象的概念を乖離させない具体例としてゲームは有効であると考える。

本論文では、著者らが C 言語の授業のために開発した、じゃんけんゲームを題材とした演習教材とその教育効果について紹介する。この教材は誰もがルールをよく知るじゃんけんを題材とし、一つのプログラムを授業内容に従って段階的に改良していく構成となっている。じゃんけんが題材であることにより、受講者は抽象的な概念を身近な具体例に当てはめて考えることが可能となる。また、受講者にとっては毎回の題材が同じであることにより、具体例のイメージがしやすい。教科書における概念の登場に合わせてプログラムにその概念を一つずつ導入していくため、対象となる概念の便利さを実感しやすい教材となっている。この論文では、まず演習教材の概要を示し、本学における授業実践とアンケート結果に基づいた教材に関する考察を示していく。

2．教材の概要

開発した演習教材の内容について述べる。本教材の出題スライド一式と解答の一部を著者の GitHub リポジトリに公開する[2]。本教材は著者らが独自に作成したものであるが、現在も授業課題として利用中であるため、リポジトリでの解答の公開は一部に留める。希望があれば、教育関係者にはメールで全解答を送付する。

例として第一回課題の出題スライドを図 1 に示す。本課

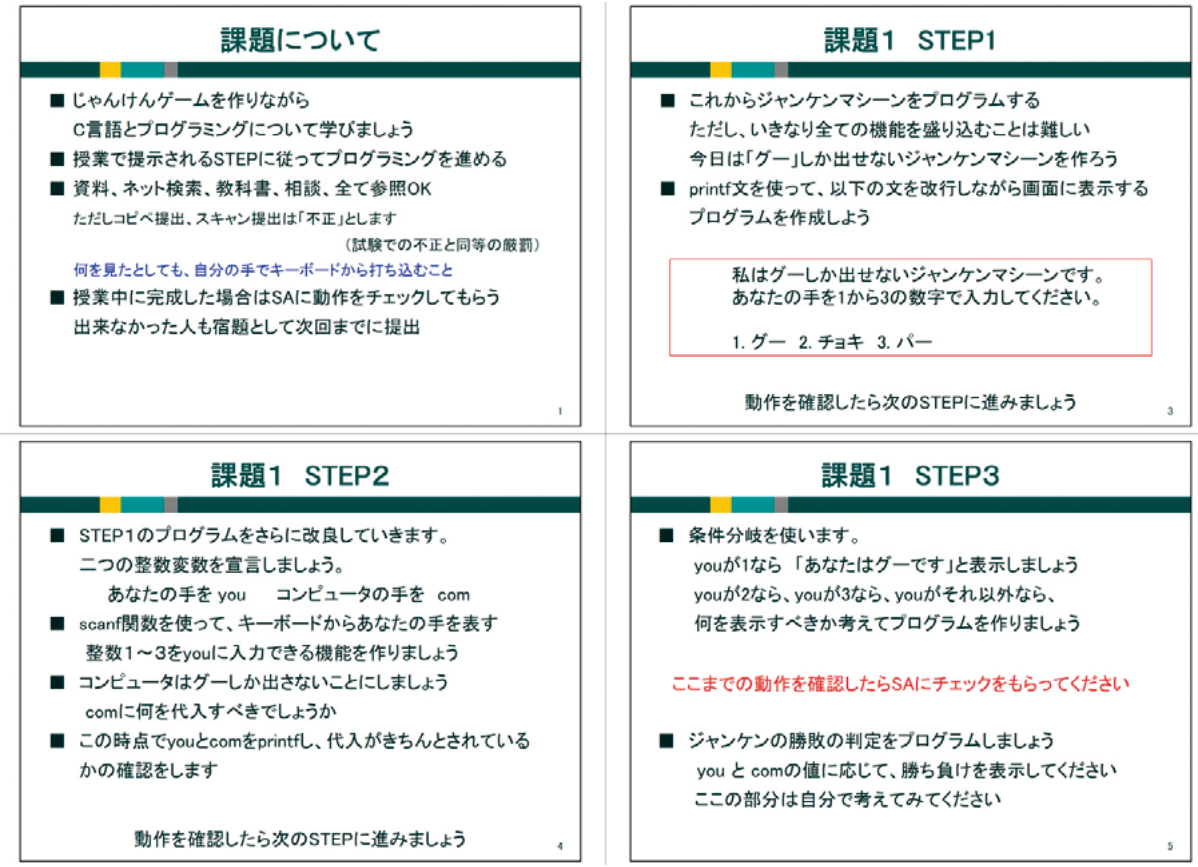

図 1． 第一回課題の出題スライド

表1． 各回の実施事項の概要

回	STEP	習得事項	ミッション
1	1	printf()	画面にじゃんけんの説明を表示
	2	変数とscanf()	キーボードから変数 youに整数1, 2, 3のいずれかを入力
	3	if文	変数 com を常に1（グー）とし、変数 youの入力結果と勝敗を表示
2	1	while文	変数 you に1, 2, 3以外が入力されたときに、入力をやり直させる機能
	2	for文	じゃんけんを五回勝負とし、今何回目かを表示
	3	配列	配列 com_array[5]を使ってcomのパターンを事前入力可能にする
3	1	関数（戻り値）	プレーヤーの入力部分を独立させ、関数 input_you() を作成する
	2	関数（引数）	勝敗判定の関数 void win_lose(int you, int com)を作成する
4	1	グローバル変数	一回の勝負を関数 void one_junken() として独立させる。com_array[5]はグローバル変数とする （常にcom_array[0]が使用されてしまう）
	2	静的変数	one_junken()内のインデックスを静的変数とし、com_arrayを正しく参照する
5	1	配列のポインタ渡し	com_arrayをmain関数のローカル変数とし、one_junkenに配列のアドレスを渡す
	2	文字列の使用とポインタ渡し	勝敗を文字で記録するchar result[6] を導入し、one_junken, win_loseにポインタ渡しする。勝負ごとにW, L, E を記録し、最後に出力する
6	1	変数の値渡し	勝敗の回数を記録する num_win, num_lose, num_evenをmain 関数で宣言し、one_junkenに値渡しして勝敗をカウントし、最後に出力する（正しい勝敗回数が得られない）
	2	変数のポインタ渡し	上記の変数をポインタで宣言し、one_junkenにポインタ渡しして最後に出力する
7	1	機能の共存	5-2と6-2の機能が共存したプログラムを作成する
	2	構造体メンバのポインタ渡し	com_array, num_win, num_lose, num_even, resultを一つの構造体にまとめ、one_junkenにはメンバ変数をポインタ渡しする（冗長）
8	1	構造体の値渡し	構造体を直接 one_junkenに値渡しできるようにする(カウンタが機能しない)
	2	構造体のポインタとアロー演算子	構造体を one_junkenにポインタ渡し出来るようにする
9	1	ファイル出力	result を外部ファイルに出力する（上書き・追記）
10	1	ファイル入力	com_array を外部ファイルから取り込む
11	1	乱数	comをランダムに決める関数 random_com()を実装する

題の解答例はリポジトリの Janken1_3.c である。第一回は「グーしか出せないジャンケンマシーン」と題したテキストベースのゲームプログラムを作成する。簡単なコードを作成する課題ではあるが、学生は例題やプロトタイプのないスクラッチからのコーディングを体験することになる。ここが本課題の一つ目のポイントである。例を写すのではなく、口語で書かれた出題文をプログラムに翻訳するという、プログラマーにとっては当たり前の作業でありながら、文法に傾倒した大学のプログラミング教育では省略されがちな手順をしっかりと踏ませる。学生は問題文をまずよく読み、授業で学習した概念と照らし合わせながら、プログラミングするべきことを自ら考えることになる。学生には問題文をよく読むこと、例のあるコードを書き写すことがプログラミングではないことを繰り返し呼びかける。

第一回は if 文までを範囲とするが、この時点でじゃんけんの手を変数にうまく格納できているかどうかは学生によって出来が異なる。じゃんけんの手を変数に格納し、さらに勝敗判定を部分として独立に実装できていれば、プログラミングの基本的なセンスが身につき始めていると考える。変数を使用することが出来ていれば、コードを少し書き換えるだけでチョキやパーしか出せないじゃんけんマシーンもすぐに作成できる。しかしながら、単に if 文でプレーヤーの手に対する勝敗を分岐するだけの、取り回しの効かないコードを提出してくる学生も多い。（すなわち、こちらがグーなら引き分け、チョキなら負け、パーなら勝ちと出力するだけのプログラムのこと。）このような学生には、この時点で一度解答を配布し、方向性の間違いに気づかせることが重要となる。

表1に各回の課題の実施概要を示す。第二回以降、繰り返し、配列、関数（戻り値と引数）の利用に進んでいく。学生にとって最も踏ん張りどころとなるのは、ポインタ（および、ポインタ渡し）の概念を学ぶパートである。ここでは一旦グローバル変数を使って関数間でのデータの受け渡しを行うステップを経由し、その上でポインタを使ったコード、構造体を使ったコード、構造体のポインタを使ったコードへとプログラムを改良していく。C言語のポインタの概念は決して簡単なものではないが[3]、ここでは「変数のポインタ渡し」の使い方を理解することのみに重

点を置いている。

教材には途中でわざと「結果が間違いとなる課題」を出している箇所もある。そのような箇所は表1において赤字で示されている。例えば、独立した関数の中で繰り返し呼ばれる変数を静的変数にし忘れるケース、もしくは、ポインタをつかわずに変数を値渡ししてしまい、関数の中で想定通りに変数の値が書き換わらないケースなどが含まれている。受講者はこのような間違いの例なども作成しながら、新しい概念の重要性や使い方を体験的に理解する。

この教材の二つ目のポイントは、最初から自分で書き始めたコードを何度も実行・デバッグしながら改良して使用していくことにある。従来のように例文を写す形式の演習では、プログラミングは、正しく写せるかどうかの課題になりがちである。この教材では小さくとも自分で最初から作成したコードを用いるため、受講者は一貫してコードの実行イメージがつかみやすい状態にある。課題を段階的な改良型にすることにより、受講者は常に動く状態のコードをキープし、動作チェックを行いながら改良を行うという作業スタイルをとることになる。実はこの小さなトライ＆エラーの繰り返しのスタイルこそがプログラミングにおいて最も理想的であり、本教材を使用すればそのスタイルを体験的に身に着けることが可能となっている。

この教材の三つ目のポイントは、この課題を通して同様のコードが見やすく保守しやすいコードに変わっていく過程が体験できる点にある。プログラミングは原理的にはforとifさえ使えれば、どのようなコードでも書くことが出来る。これはこれで大切なことであり、受講者には早めに理解してもらいたい事項でもある。一方で、一歩踏み込んだ文法概念は他人が見て理解しやすいコードを作成するために導入されている場合が多い。本教材では、構造体のポインタを用いて変数の受け渡しをシンプルにしていく過程を通じ、保守しやすいコード記述の重要性を体験させることが可能となっている。

3．授業実践

本教材は広島工業大学工学部電気システム工学科2年次1Q科目の「プログラミング演習」において利用した。「プログラミング演習」はC言語の基礎的内容の授業であり、クォーター制で週2回実施される。ポインタ・構造体を一通り網羅するレベルの書籍を教科書として指定している[4]。関連科目としては、事前科目としてC言語プログラミングの初歩的内容を学ぶ「プログラミング」が1年次の1Qに開講されている。並行して1、2年次の4Qにプログラム実践基礎、プログラム実践応用を開講しており、こちらではArduinoを搭載したロボット（Zumo）の制御を通して実践的なプログラミングを扱っている。「プログラミング演習」は2年前期に位置付けられていることもあり、受講者が自身のプログラミングへの向き不向きを、研究や就職に向けて自覚していくための重要なポイントになっていると考える。

「プログラミング演習」の各回は説明と演習で構成されており、前半はC言語の各概念について教科書とスライドを用いて説明する授業、後半はノートPCを用いた演習を行っている。コーディング環境としてはMicrosoft Visual Studioを用いる。演習時間においてはTA・SAと教員が教室を巡回しつつ、受講者同士の話し合いも推奨した状態で実施している。講義を進めつつ時間を小分けにして演習を実施する形式も考えられるが、受講者のレベルによってペースが異なることもあり、評判はあまり良くない。演習時間をある程度まとめて確保する方が良いことを示唆する意見はアンケート結果にも見られた。

演習中に特に重要なことは、受講者のプログラミングに対する意識について何度も繰り返し呼びかけることであった。多くの受講者はプログラミングとは手本となるコードを写すものだと考えている。このため、コードを写し間違えて実行が出来ないなどの事態になると、すぐに挙手して「動きません」と質問をする。実際に対応してみると、エラーの原因はエラーメッセージに全て書かれている。英語の専門用語が読めないという理由はあるだろうが、このような意識では、何度質問に対応していても上達にはつながらない。写し間違いは誰にでも起こることなので、むしろ、エラーが出た際にエラーメッセージを読みながら自分のコードと向き合うことが大切であることを何度も伝える。例えるなら、エラーメッセージときちんと会話することを何度も何度も呼びかけた。翻訳サイト等を利用して英語のエラーメッセージを理解することは当然必要となる。

printf文を使った制御位置の把握や変数の中身を把握するデバッグについては、本来であれば授業に取り込んでしっかりと教え込みたい内容である。しかしながら、デバッグを伴うコーディングの経験が浅いうちからそれについて教えても実感がわかないと現状では判断しており、printf文を使ったデバッグなどは個別対応の中で教えていった。

4．アンケート結果と考察

令和3年度・令和4年度の「プログラミング演習」終了時に実施した匿名の受講者アンケートの結果を用いて、本教材の有効性を示す。図2はC言語に関して学習した諸概念の使い方を理解できたかについてのアンケート結果である。ここでは各項目について「大いに当てはまる」「そこそこ当てはまる」と回答した受講者を「理解を実感した受講者」として定義することとする。まず序盤のifやfor

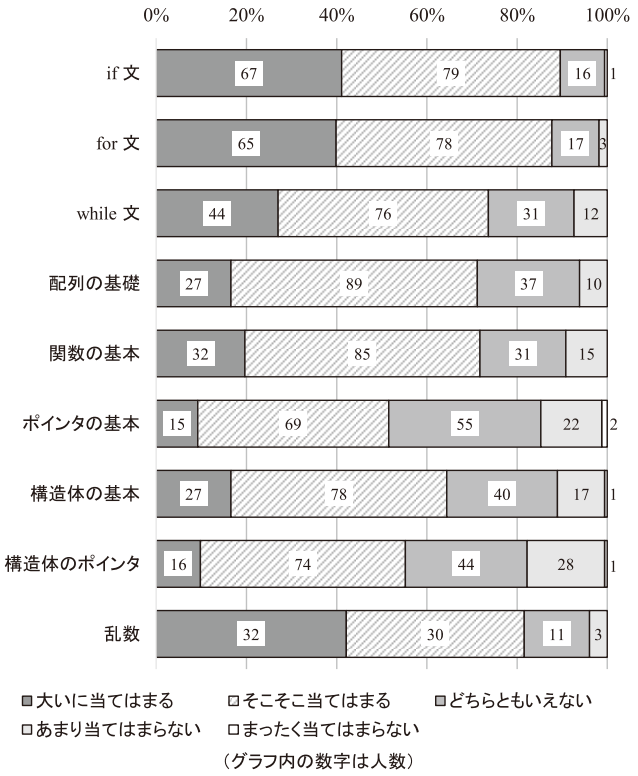

図２.「使い方を理解できたか」（令和３，４年度の受講者への終了時アンケートの結果）

については９割近い学生が理解を実感している。回を進めるにつれて理解度は下がり、ポインタのパートで理解を実感した受講者の割合は50％と底を打つ。ここで「50％しか理解出来ていない」という結果は、通常の授業であれば教育効果としては疑問の余地があるかも知れない。しかしながら、Ｃ言語のポインタの概念に関して言えば、地方私立大学における50％という数値は全くもって低くはない。ある国立大学において１学期のＣ言語の授業が終了した状態から授業を引き継いだ際、「Ｃ言語の授業でポインタを理解できたか」という問いかけについて挙手でアンケートをとったところ、約80人中２人程度しか挙手をしなかった経験なども著者にはある。それだけポインタに関して理解の実感を与えられる授業は難しいと考える。構造体や乱数に関する単元はポインタよりは理解度が高くなっており、これは現場の感覚とも一致している。

図３はじゃんけんゲームの段階的な改良が各事項の理解に役に立ったかについてのアンケート結果である。こちらも図２とほぼ同じ傾向を示したが、全体的に肯定的な回答をした受講者の割合が増えている。じゃんけんの作成をベースにした教材は少なくとも本学科の受講者にとってはかなりの高評価を得たことがわかる。

本アンケートにおいては自由記述欄にも肯定的な意見が多かったが、その中でも単なる感想を超えていたものとしては以下のものがあった。

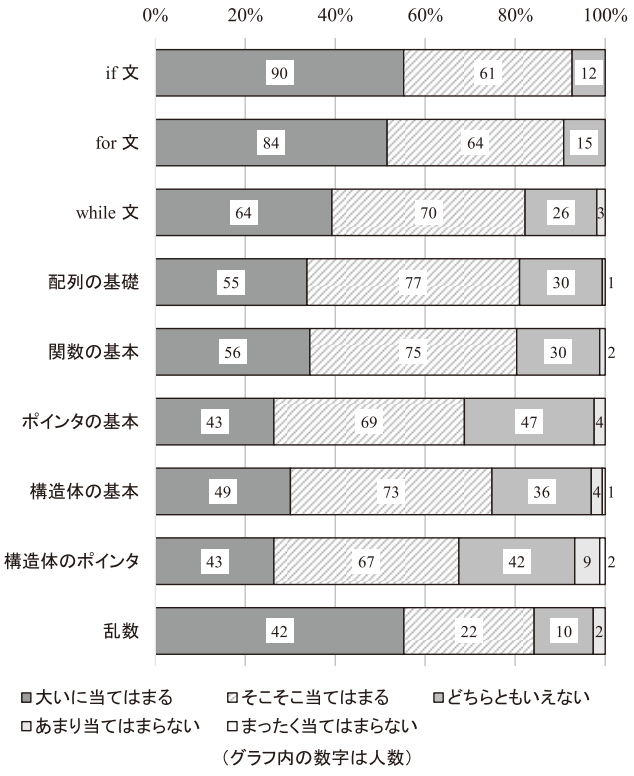

図３.「使い方を理解するためにジャンケンゲームの作成が役に立ったか」（令和３，４年度の受講者への終了時アンケートの結果）

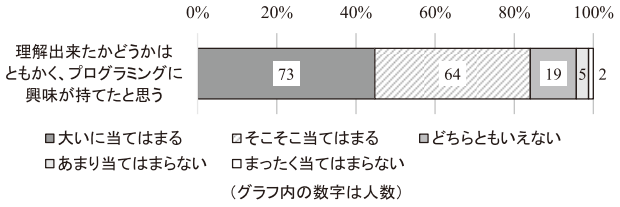

図４.「興味が持てたか」（令和３，４年度の受講者への終了時アンケートの結果）

・授業形式でじゃんけんゲームを毎授業で学ぶ知識を使って作成していくので、プログラムの知識をこつこつと学べていくことができ、良かったです。今後も何かのゲームを段階を踏んで作成する授業形式がいいと思いました。
・友達と相談して行うことができるため、質問をする前に考えることが出来る点がとても良かったと感じました。
・１年のプログラミングのときは、作業が多かったが、プログラミング演習では自分でゲームを作っていくので、やってみたいことがあれば調べて完成させていくのが楽しかったです。

図４は授業を通してプログラミングに興味が持てたかについてのアンケート結果である。「理解できたかどうかはともかく、プログラミングに興味が持てたと思う」という質問について８割を超える学生から肯定的な意見を得られた。並行して4Q に実施しているロボットプログラミング

との相乗効果でもあると考えられるが、これは近年の電気システム工学科においてはなかった傾向である。さらに、令和３年度にこの授業を履修した学生のうち、３年次の履歴書作成演習において「プログラミングが得意である・好きである」と記述した学生は71人中10人にも及んだ。比較対象を定量化することは出来ないものの、プログラミングに対してこれまでとは明らかに異なった意識を学生に与えることに成功した。

５．まとめ

本学科で使用したオリジナルのじゃんけんゲーム教材について紹介し、実践とアンケート結果に基づいて確認した教育効果について述べた。地方私立大学で実施されるプログラミング授業としては比較的高い理解度を得ることに成功し、プログラミングを得意と自覚する学生を一定数増やすことが出来たことも確認した。

プログラミングは研究においても企業においても基礎的なスキルとなるため、プログラミング力の向上は理系能力全体のベースアップにつながることが期待できる。一方で数学や物理とは異なり、現在のところ教育的な配慮や工夫がされにくい授業科目でもある。このような教育研究を通した情報共有は小さなことではあるが、今後も行っていくべきであると考える。教員のプログラミング能力を測る指標というものが存在しておらず、さらには、プログラミングが得意な教員が良いプログラミング教育を実施する能力が高いとも言い切れない。それぞれの大学のレベル差もあるとは思うが、学生目線のプログラミング授業は今後も積極的に開拓していくべきである。

ところで、本論文の草稿の作成を終えた直後の時期を起点とし、ChatGPT（OpenAI）に関する議論が急激な高まりを見せ、大学におけるプログラミング教育においても無視することの出来ない要素となってしまった。ChatGPTの利用を肯定的に捉えた上で本教材が果たすべき役割を論ずるには、まだ時間・データ・試行が不足している。しかしながら、時代に合わせた教育手法の急速なアップデートが必要とされる段階に入ったことについては疑いの余地はないと考えている。

謝辞

広島工業大学において関連科目の担当として本授業のサポートをしてくださいました電気システム工学科の先生方、TA を務めてくださいました学生の皆さま、ならびに、アンケートに協力してくださいました受講生の皆様に心より御礼申し上げます。

参考文献

[1]松岡雷士「プログラミング教育における確率ゲームの活用とその教育効果」応用物理教育，44（２），pp. 65-70（2020）.

[2]著者の GitHub リポジトリ：
https://github.com/leo-matsuoka/article_source_codes/tree/master/2022_C_Edu_Janken

[3]前橋 和弥「C 言語 ポインタ完全制覇」技術評論社（2017）.

[4]高橋 麻奈「やさしい C 第５版」SB クリエイティブ（2017）.