〔論　文〕

プログラム学習を支援する「プログラム駆け込み寺」の取り組み

石橋　俊二[*1]・工藤　達郎[*2]・千田　陽介[*2]

Last Resort for Computer Programming: Student Program Learning Support Activity Report

Shunji ISHIBASHI[*1], Taturo KUDO[*2] and Yosuke SENTA[*2]

Abstract

Our department provides 120 hours of programming classes in the initial two years of the program. However, despite these efforts, students' programming skills have not improved. To alleviate this situation, the authors organized the last resort for computer programming (LRCP) course, which was held once a week at the Learning Commons on campus, and it was attended by approximately 80 people. LRCP has immensely helped several students in improving their programming skills. This paper describes the curriculum of the current programming classes in our department, the level of programming expertise of the students, and the activities of LRCP. Finally, it considers the key points that are essential for sustainable LRCP activities.

Key Words：Computer programming, Learning, Extracurricular activities

1　はじめに

筆者らが所属する久留米工業大学情報ネットワーク工学科は，コンピュータのハード・ソフト・コンテンツに関する学問を学ぶ所である．これらの学問のベースとなるのがコンピュータプログラムである．コンピュータプログラムを学ぶことで，コンピュータを思い通り動かすだけでなくコンピュータが出来ること／簡単には出来そうにないことを区別するセンスを磨くことができる．そのため当学科では，１年と２年の前後期に渡り，プログラミングⅠ～Ⅳという授業を行っている．その総授業時間は 120 時間である．

しかしながら，学生のプログラミングスキルは期待している程伸びていない．この問題は当学だけでない．他大学でも問題となっており幾つかの興味深い報告がなされている．堀越[(1)]は筑波学院大学の学生に対し，プログラミングの経験，実力および意識に関する調査を行い，興味関心は高いものの「難しい」と感じて踏み込めてないことを明らかにした．また小松[(2)]は国内外のプログラミング教授法をサーベイし，それを元に独自の教授法を提案し山形大学や文京学院大学で実践している．この方法は，１.題材は学生が興味を持つようゲームとする，２.難易度は 90 分以内に完成できる程度とする，３.プログラミングは難しくないことを示すため，教員が一回リアルタイムに課題を解いてみせる，といった内容である．小松の提案する教授法は導入教育としては優れているかもしれない．しかし当学科はプログラミングを教養でなく，専門知識として修めることを目指しており，ゲームばかりを作るわけにはいかない．さらに近年のゲームアプリは非常に高性能で演習で作るゲームとのギャップが大きい．そのため逆に興味が薄れる危険もある．

プログラミングは目の前のコンピュータでコードを走らせることで，動くか動かないかがはっきり分かる．そのため授業は演習形態が多い．演習の場合特に，内容に一度ついていけなくなると以後の授業が分からなくなり，興味をなくし，ますますついていけなくなるループに陥る．そのような負のループを防ぐため，授業中 TA が見回りながら分からない所を教えて回っているが限界がある．さらに懇切丁寧なサポートは「分からないままでもなんとか単位が取れる」という受動的な姿勢を生み出す危険性を秘めており，その加減が難しい．分からなければ聞くという積極的姿勢を学生に求めている．

分からないことを聞くためには，それを気軽に聞ける窓口があることが望ましい．そこで工藤は「プログラミング駆け込み寺」を創設し，考えに賛同した残る筆者らが運営に協力した．対象者は情報ネットワーク工学科全学年である．

[*1] 電子情報システム工学専攻，[*2] 情報ネットワーク工学科
令和元年10月31日受理

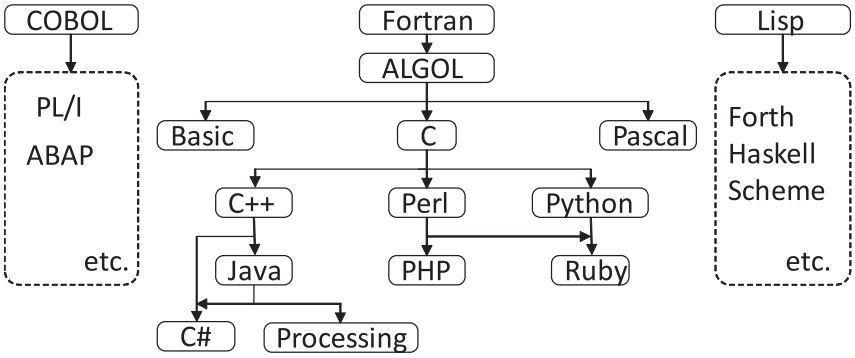

Fig. 1　Phylogenetic tree of computer program

「プログラミング駆け込み寺」は巨海らが運営していた「物理駆け込み寺」[3]に似ている．物理駆け込み寺では個々の学生に対し寺に来るよう呼び出しを行うもので，大きな成果を上げていた．それに対し当プログラミング駆け込み寺では，来るように促すものの呼び出しは行わなかった．あくまで本人の「向上したい」という意欲を重要視したためである．向上心のある学生に対するプログラム教育として須藤の Picture 塾[4]がある．これは入学当初から CG やプログラムに興味がある学生に対しその技術を教える活動である．学生はどちらかというと上位レベルの者が多く，そのレベルをさらに高める効果はあるものの，全体のボトムアップには繋がらない．当駆け込み寺では，Picture 塾に通う程強い向上心は無いものの，自力で卒業する意欲がある学生を対象とした．

本稿では当学科のプログラミング教育のカリキュラムと学生のプログラミング能力の現状を示したのち，駆け込み寺の活動について報告し持続的なプログラミング教育支援活動について考察する．

2　プログラム言語

読者の理解を助けるためここではプログラム言語について簡単な解説を行う．図 1 はネット[5]の情報を参考に以後の議論のため簡略化したコンピュータプログラム言語系統樹である．1950年代，異なる設計思想から Fortran，COBOL，Lisp という三つのコンピュータ言語が誕生しそれぞれ進化した．中でも Fortran から進化した ALGOL は「ALGOL 系」「ALGOL 風」と呼ばれる数多くの言語を輩出した．さらにその中の C 言語から派生した様々な言語は C-family と呼ばれ，2019年 9 月現在 Wikipedia に 68 の言語がリストアップされている[6]．C-family の中には C++，C#，Java，Python といった現在主流となっている様々な言語が含まれている．これら C-family に属するプログラミング言語は文法が非常に似ている．外部ファイルやモジュールの取り扱い（use, include, import等）や入出力（print, println, printf 等），配列やリストの扱いといった細かい違いはあるものの，大枠は同じである．

図 2 に 1 ～99 の整数のうち，7 で割り切れない数の和を実際に加算して求めるプログラムをいくつかの C-family 言語で書いた例を示す．プログラムの中核は図中太字で示した部分で，始め合計値（sum）を零に初期化した後，カウント（i）を 1 から 99 まで上げていき 7 で割り切れなければ加算し，最後に結果を表示する処理となっている．図より中

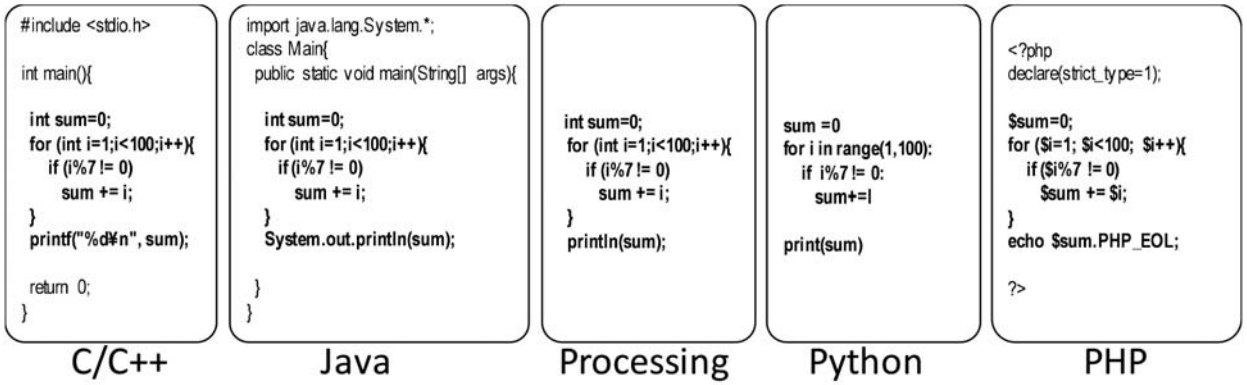

Fig. 2　Same problem solutions in di.erent language

核部分はほとんど同じ記述であることが見てとれる．特にCやJava，Processingに関しては画面表示以外は完全に一致しており，そのままコピーアンドペーストしても支障はない．もちろん多種多様な言語に分化したのはそれなりの理由があり，高度な処理になればなるほど記述に違いが出てくる．しかし初学者が学ぶ，変数，配列，条件式，ループ，サブルーチンコールといった基本部分ではほとんど違いが見えない．

前述のようにC-family言語は現在の主流言語であり，その知識は社会に出てから役に立つ．さらに言語間の差異はわずかであり，一つの言語を覚えると他の言語に流用できる．そこで当学科では卒業までにC-family言語を用いて，ある程度のプログラムが記述できること目標にしている．ただし全学年を通し一つの言語のみで教育することは，プログラムの本質と個々の言語仕様との見分けがつかなくなる虞がある．そのような配慮から当学科のプログラム授業（プログラミングⅠ～Ⅳ）では，ProcessingやC/C++といった異なる言語を用いている．なお近年はScratch[7]といったグラフィカルな言語が発達し，初等中等教育におけるプログラミング教室で活発に利用されている．プログラム教育の初期段階ではテキストベースのプログラムより，このようなグラフィカルな言語の方が効果的であるという報告もある[8]．しかし報告によるとその効果の差はわずかであり，その後C-family言語に移行すること，キーボードを用いたテキスト入力の練習といったプログラム以外のITスキルの向上といった観点から，最初からC-family言語を用いたテキストベースプログラムを教えている．

3　情報ネットワーク工学科におけるプログラム授業

はじめに述べたように当学科では1，2年の前後期，4セメスタに渡りプログラミングⅠ～Ⅳという授業を行っている．ここではそれらの授業の内容と目的を，各担当教官に取材した結果を述べる．各授業では，変数，関数，配列とともに，条件式（`if`文，`switch`文），ループ（`for`文，`while`文），及びサブルーチン（関数）といった基礎的なことを毎回初めから教えている．同じ内容を言語や問題設定を変え繰り返し演習することで，基礎を確実に理解してもらう運用である．なお，以前はプログラムⅡ以降は習熟度に応じてクラス分けを行い，出来る者には高度な内容を，出来ない者には基礎的な内容を教えていたが，授業の難易度（学生にとっての負荷）や評価に対する不平等感が強く廃止され，現在は全員が等しく同じ授業・課題を受けている．

プログラミングⅠ（1年前期）　入学して初めて行うプログラムの授業であることから，プログラムとはどういうものかという概念から教えている．演習はProcessingを用いて円や四角などの図形を描画するもので，プログラムができることは凄くて楽しいと感じてもらえることを目指している．

プログラミングⅡ（1年後期）　引き続きProcessingを用いてグラフやアニメーションの描画を行う．マウスの位置に応じてインタラクティブに表示を変える処理を通じ，データ構造という概念が必要だという感覚を養うこと，ある程度の長さのプログラムを作るのに慣れることを目指している．

プログラミングⅢ（2年前期）　全体を二班に分けProcessing以外の言語としてC言語およびC++言語を前半後半に分けて学ぶ．このうちC言語の授業ではプログラム内におけるデータの動きを理解するため，あえてグラフィカルな要素を排除し，素数を求めたり数値積分を行ったりするコマンドラインのプログラムを作る．一方C++言語の授業ではオブジェクト指向の考え方（クラスやインスタンスの概念）に触れるためGDIライブラリを用いて簡単な画像処理を行う．

プログラミングⅣ（2年後期）　全体を三班に分けプログラムに対する多角的な視点を持ってもらうことを目指す．一つ目の班ではWEBブラウザ上でProcessingが動作するp5.jsを用い，スマートフォンでも動作するプログラムを作る．二つ目の班では本来整数型しか扱えないコンピュータ内で実数型（IEEE754）をどのように扱っているか知るためC言語による実装を行う．三つ目の班ではopenFrameworks（C++言語）を用いて，最新のメディアアート構築のさわりを体験してもらう．

その他　以前より教官の間では，プログラミングⅠ～Ⅳだけではプログラムの実力が付いてないことが問題視されていた．そのため学年全体を10人程度の班に分け，各担当教官が固有の演習を行う工学基礎セミナという授業において，プログラミングの補講という形を取ることもある．その内容は班によって違うが，簡単なCの演習問題を数多く解か

せるもの，少し難しい課題を数ヵ月かけて解かせるもの，Raspberry PI や Arduino を用いて電子回路を制御するものなど様々である．

4　情報ネットワーク工学科学生のプログラム能力

当学科における学部生のプログラミング能力の現状を調べるため，2019 年 9 月にプログラミングⅡ（1 年生），Ⅳ（2 年生）の初回授業で能力測定を実施した．測定は図 3 に示した 8 問の試験である．このうち 1 ～ 4 は記述式，5 ～ 8 は選択式の問題である．紙面の都合上，実際の設問および選択肢の内容は割愛する．図のように設問自体は C-family 言語（ここでは Processing を想定）の変数と配列，`if` 文と `for` 文の基本的な部分について問うものである．ケアレスミスの可能性を考慮しても 6 点は取れないと習熟したとは言い難い．

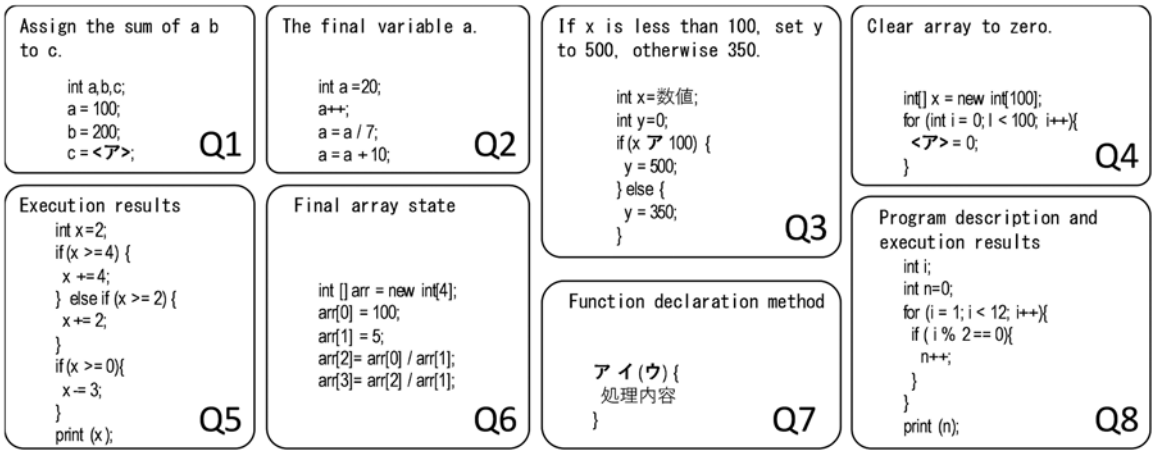

Fig. 3　Programming ability test (Q 1-4: Description, Q 5-8: Selection)

プログラミングⅡは 121 名，Ⅳは 83 名が試験を受けた．結果を度数折れ線（ヒストグラムを折れ線で表現したもの）で表現したものを図 4 に示す．ただし両者の違いを見比べるため縦軸は受験者の数で割った百分率で表現している．驚くことに両者の得点分布は 0 ～ 2 点，3 ～ 5 点，6 ～ 8 点の割合がほぼ 15：30：55 であり，ほとんど違いがないことが見て取れる．昨年と今年においてプログラミングⅠの指導内容を変えてないことから

1．プログラミングⅡおよびⅢの教育効果がない
2．学年間における学習意欲，もしくは基礎学力が異なる

のいずれかが原因としか考えられない．このことを検証するには，複数年に渡って継続して能力測定を行う必要がある．

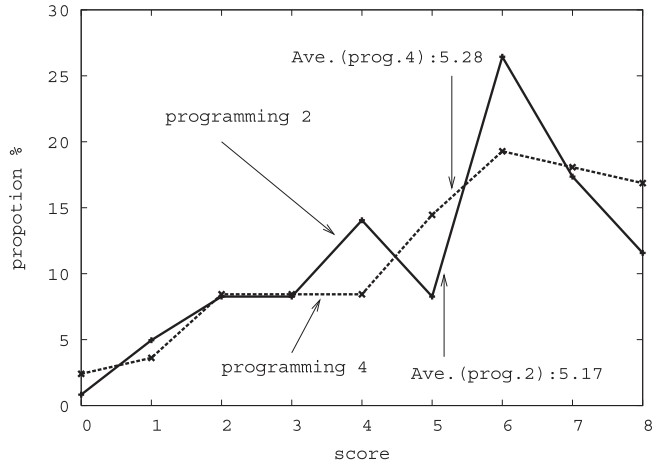

Fig. 4　Histogram of programming ability test between programming II and IV

Table 1　Correct answer rate for each question

QuestionNum.	1	2	3	4	5	6	7	8
Programming II	76%	78%	69%	20%	68%	86%	55%	65%
Programming IV	77%	73%	82%	30%	67%	83%	51%	76%

表 1 に各設問の正答率を示す．設問 1 に関しては両者とも正答率に変わりはない．また誤答のうち半数近くは a+b の計算を実際に人が行い 300 と書いていた．これについては設問を正しく読んでないと言えなくもないが判断に迷う所である．設問 2 に関しては 30 という誤答が多かった．おそらくこれは途中演算 a++ や a = a/7 で変数 a の値が変わると感じてないのかもしれない．他にも 10 や 14 といった誤答も多かったがどういう考えでこの値に至ったか推定できてない．またこの問題はプログラミングⅡの方が正解率が高いのも興味深い．設問 3 に関しては < という正解に対し <= という誤答が多かった．問題文は日本語で「100 より小さい場合は…」となっており，正しく「未満」の意味であるが「以下」と捉えられてもおかしくはない．もう少し問題文を工夫すべきだったかもしれない．なお <= も正答とした場合の正解率はⅡが 77 %がⅣが 88 %となる．設問 4 の誤答は若干 int [i] という記述が多いようであったが様々なパターンがあり一概に言えない．この問の正解率の悪さから，多くの者は配列について十分理解できてないことが見て取れる．続いて設問 5 ～ 8 に関しては選択問題のため誤答から傾向をあまり読み取ることができなかったが，設問 7 に関し「ア：戻り値の型，イ：関数名，ウ：引数の記述」という正解に対し「ア：関数名，イ：void，ウ：条件式」を選択した者がⅡは 25 %，Ⅳは 32 %もいた．これは 3 割近くの者が関数を作成できないことを意味する．

下位レベルの者が具体的にどのようなコードを記述するか知るために，ある年のプログラミングⅢの期末試験の解答を抜粋してもらった（個人保護の観点から筆跡が出ぬようテキストデータに起こしてもらった）．プログラミングⅢは原則，授業中の課題を提出すれば合格となるもので，期末試験は課題を出さなかったどちらかと言うと下位レベルの学生が受ける．「1000～2000の整数のうち 3 もしくは 8 の倍数を小さい順に重複なく表示する」という問題の解答例を図 5 に示す．この問題は FizzBuzz 問題[9]をアレンジしたもので

1．for 文が正しく書けるか
2．if 文が正しく書けるか
3．剰余演算子 % の機能を理解しているか

あたりが鍵となる．なお printf 関数の使い方に関しては，プログラムの本質ではないという立場から問題文にその使い方を明記している．また図中黒字部分はあらかじめ問題文に記載しており，赤字部分が学生の解答である．

図より解答者 1 を除いて for 文が理解できてないことが分かる．また解答者 2，3，4 は if 文と for 文の区別が付いてないことも見て取れる．さらに <= とかかなければいけない比較演算子に対し，≦と記述する者も多い．文字‘≦’は 1 バイト文字に含まれてなく，1 バイト文字で記述するプログラムコードでは表現できない．これは実際にコーディングすればすぐ気づくことであり，明らかに今まで実際に手を動かしてコーディングを行ってきてなかったといえる．

5　プログラミング駆け込み寺の活動と評価

5．1　駆け込み寺創設理念

以上のように半数近い学生はプログラムがあまり，もしくはまったく身についてない．このままでは以後の授業につ

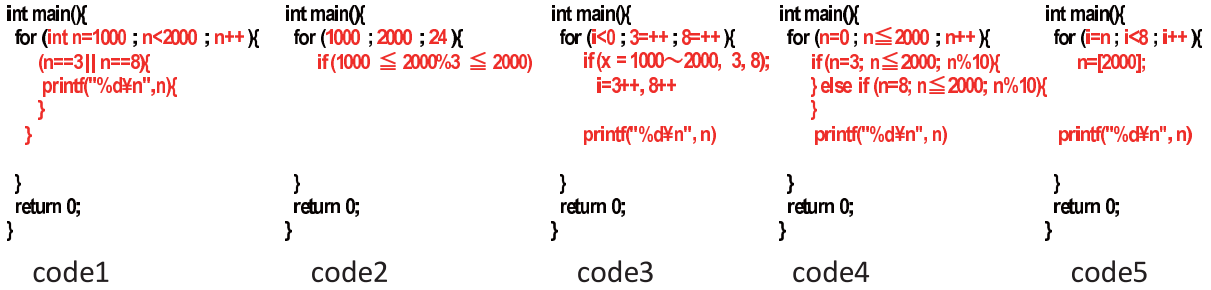

Fig. 5　Sample answers for programming examinations in information class

Fig. 6　Photograph of last resort for computer programming

いて来れないと考えられる．そこでそれに危機感を覚えた学生が自主的にプログラムを学び直す場として「プログラミング駆け込み寺」を創設した．寺の目的はプログラムの基礎：

1．プログラムがどこから始まりどの順番で動くかの理解
2．変数や配列の概念と宣言方法
3．if文，for文，while文の書き方
4．条件式，代入式の書き方
5．関数の概念と宣言方法

といったことを修得してもらうことである．ただし，３章で述べた各授業で出された課題が分からないといったプログラム全般に関する悩み事も引き受けている．寺の開催は週一回，三時間程，ラーニングコモンズと呼ぶ多目的自習教室で行った（図６）．

５．２　駆け込み寺の運営

駆け込み寺を訪れた者は，まず訪れた理由をインタビューされることから始まる．その内容は大きく二つに分かれ，授業で出された課題が分からないという者と，全体的にプログラムが分からないので力をつけたいという者がいる．前者も受け入れてはいるが，当寺のメインターゲットは後者である．

プログラムが分からないと来た者は，まず5.1節で述べたプログラムの基礎が理解できているか順を追って質問し，

```
Int func(int  a, int b){
 if (a != b){
  if (a > b){
   return a;
  } else {
   return b;
  }
 } else {
  return a+b;
 }

void setup(){
 int c=_____;
 printf("%d¥n",c);
}
```

問 1. 下線部分が func (25, 100) だった場合に，
printf 文で出力される数値はいくつか

問 2. 下線部分が func (200, 70) だった場合に，
printf 文で出力される数値はいくつか

問 3. 下線部分が func (50, 50) だった場合に，
printf 文で出力される数値はいくつか

Fig. 7　Example of pro.ciency test

Table 2　Established student

ID	grade	Resu lt of ability test (Fig. 3)								
		Q 1	Q 2	Q 3	Q 4	Q 5	Q 6	Q 7	Q 8	score
A	2	×	✓	✓	×	✓	✓	×	✓	5
B	1	✓	✓	✓	×	✓	✓	✓	×	6
C	1	✓	✓	✓	✓	✓	✓	×	✓	7
D	1	×	✓	✓	×	✓	✓	✓	✓	6
E	1	✓	✓	✓	×	✓	✓	✓	✓	7

どこで躓いているのかを確認する．その後躓いていた部分を重点的に説明し，最後に演習問題を解いて理解できたかどうかを確認，実感してもらう流れとした．二回目以降の場合，質問や説明のプロセスは省略し，演習と解説を繰り返す．演習問題は，自信をつける意味もあり難しい問題とせず，文献[10]等を参考にできるだけ平易な，しかし本質を突くような問題とした．その例を図 7 に示す．

一方で前者：授業で出された課題が分からなくて訪れる者も全体の三割程居た．課題の意味がなくなるため，解答そのものを示すことはせず考え方のみを伝えるよう努めた．ここで授業の課題は一見難しそうに見えるものの，授業中に板書等でヒントや解法そのものを示していることが多い．学生は板書を書き写さず単にスマートフォンで撮影して安心し，記録した事実すら忘れていることがある．問題文とそれを出した教官の性格も考え，課題が出る前に提示されたものがあるはずと，何か示されたものがないか聞きだすところから始まった．なお，教官が出したヒントの内容は容易に想像できるが，それを提示するのは質問者の役割というスタンスを厳守した．

5．3　駆け込み寺の効果

駆け込み寺は 2018 年 11 月～2019 年 1 月，および 2019 年 5 月～ 7 月に渡ってのべ 6 カ月間行った．その間の来訪者はのべ 80 人くらいであったが，そのほとんどはリピータであり，実質定着した者は 1 年 4 人，2 年 4 人の 8 人くらいであった（2018 年度の学年）．主催者としてその者達が当寺を通し，どのくらいプログラミング能力が向上したか興味深いものの，残念ながら定量的な評価を行っていない．指導に重点を置いたため評価に時間が取れなかったことに加え，図 7 のようなテストで評価することは意味がないと考えたためである．このような問題を通して寺の運営を行っており，いわば試験対策をしているようなものである．対策された試験では実際の能力を測ることができない．

試験対策を行ったというバイアスの可能性も踏まえつつ駆け込み寺に定着した学生が図 4 のどのレイヤに属するか調査した．結果を表 2 に示す．5 名しかデータが無いのは当時 2 年生だった 3 名はプログラミングⅣに合格したため，試験を受けなかったためである．表より定着した学生は全員，それなりにプログラムが分かっているレイヤに属していることが見て取れる．定着前の能力を計測してないため，定着前後の能力値を比較しこれが寺の効果であることを客観的に証明することはできないが，特にＡ君，Ｂ君は寺に訪れだした当初は変数への代入すら理解していなかったことから，寺の効果であると信じている．実際に連絡先を知っていた 2 年生二名に参加したいきさつや解決したことなどの感想を聞いてみた．以下にその回答を示す．

5．3．1　Ａ君の場合

参加した理由　プログラミングⅢ，Ⅳの単位を落としたため．
参加してみて　マンツーマンで分かりやすく教えてくれたので理解できた．無事（Ⅲの）単位が取れた．
今後も続けてほしいか　Yes，自分のように授業について行けなかった者はきっといるから．

5．3．2　Ｆ君の場合

参加した理由　授業中という限られた時間では質問しても納得できるまで教えてもらえず，日に日に疑問が増えていき，ついていけないほどとなったため．
参加してみて　少人数なので質問もしやすく，細かく丁寧に教えてもらえた結果疑問が解決できた．
今後も続けてほしいか　Yes，授業で教えてもらう時よりも時間をかけて理解できるよう教えてもらえるので後輩にも勧めたい．

6　今後の取り組みについて

6.1　駆け込み寺について

前章で述べたようにインタビューの結果はおおむね好評で，今後も駆け込み寺を続けて欲しいという意見であった．しかし第一筆者は現在大学院二年であり今期で修了する．そのため持続して寺を運営することはできない．これを解決する方法の一つとして，駆け込み寺を Picture 塾[4]のように組織化し，現役の大学院生が担当となって教えていく方法がある．教官と比べ大学院生は学生との年齢差が少なく気楽に訪れることができる．しかし寺を運営するのに十分なプログラミング能力と熱意，コミュニケーション力を持った者が毎年排出されるか定かでない．可能な限り存続してほしいがそれに頼るのは危険である．

そこで現在駆け込み寺の代わりとなるような WEB ページが作れないか検討中である．プログラミングⅠ～Ⅳの TA として学生を指導してきた感覚では，プログラミング能力をなんとかしたいという需要はたしかにある．しかしその大半はなんとかしたいと思う気持ちより，授業の無い自由時間に駆け込み寺に行ってみることが面倒だという気持ちの方が大きい．寺でルーチンワーク化されたノウハウを再現できる WEB ページを作ることで，より多くの者が自主的にプログラミング能力を向上させられるかもしれない．「○○言語入門」と銘打つ WEB ページは多数存在する．中には筆者らより何倍もプログラミング言語に精通した者が作ったページもあり，そのような中新たなページを作成することは無駄なようにも思える．しかし，

- 5.1節で述べたプログラムの基礎部分のみ解説することで，本来の言語が持つ全仕様を解説した量の圧迫感をなくせる
- 演習問題もプログラムの基礎部分に限定することで，ポインタ等高度な問題に遭遇しなくなる
- 当学の学生に対しポータル的な扱いをすることで，どこまで，どういう表現で説明しているか一元的に管理できる

といったメリットがあるのではともいえる．

2019 年 9 月現在，Google Document を利用し駆け込み寺で使用した資料を共有する WEB ページを製作中であり（図 8），今年度中にこのページを公開し寺機能の代替が可能かどうか検討していく．各資料を Google Document で共有することで，製作者側は今までの資料をわざわざ書き直す必要がなくなり整備のハードルを下げることができる．また利用者側が自分のドキュメントとして取り込みノート代わりに書き込むことができる．さらに指導者と共有すれば遠隔で指導を受けることも可能となる．

Fig. 8　A snapshot of the current web page of introductory programming

6．2　プログラミング能力計測

ここまで述べてきたように，駆け込み寺（5. 3節）はもちろんプログラミングⅠ~Ⅳ（４章）の効果を客観的に知るためには定期的に学生のプログラミング能力を計測する試験を行う必要がある．そのためには適切な試験問題を用意する必要がある．プログラミング能力を測る方法として基本情報技術者試験のプログラム問題や，TOPSIC[11]などがあるが，難易度が高すぎる．プログラム能力には

- 文法に則って書く力
- 記述されたプログラムを読み解く力
- 与えられたタスクを実現するコードを組み立てる力

の三つがある．5. 1節で述べた範囲でこれらの能力を評価するためには，例えばFizzBuzz問題や配列の数値の平均を求めるようなプログラムを「`for`文が書けているか」「配列の添え字は適切か」等適当なルーブリックで厳密に評価していくのが望ましいが工数がかかる．今回用いた図３のような形式は評価は簡単である．しかし，例えば`a+b`と答えて欲しいところを300と答えられるなど誤解を生みやすい設問があったり，文法問題や解読問題ばかりでコードを組み立てる力を問うてなかった．これらの問題を解決するような設問は注意深く考えればきっとできると考えており，今後そういったものを作り上げていかなくてはならないであろう．

7　おわりに

当学当学科のコンピュータプログラミング能力の修学を助けるため「プログラミング駆け込み寺」を創設・運営した．当寺は自ら上達する意思のある者を救済することを目的としたため，特に強制力は発動しなかった．それでも６カ月の期間中にのべ80人近い学生が寺を訪れ，多少なりともプログラミング能力のボトムアップを実現できた．本稿執筆時（2019年10月）も引き続き運営している．

当寺の必然性を知るため，１年生，２年生を対象としプログラミング能力テストを実施した．その結果，プログラムを記述できる知識があると言える者，それなりにある者，おそらくまったく理解してないと思われる者はおおよそ４:２:１くらいの割合で分布していることが分かった．当寺は本来この全体の1/7程いる下位レベルの者を引き上げるのが目的であるが，修学意欲が乏しくなかなか実現できない．プログラミングができることにあこがれを持つようなイベントが必要かもしれない．

第一筆者は今年度で修了する．後輩には引き続き駆け込み寺を運営してもらうことを期待する．その助けになればと現在WEBページを製作している．今後，それらの資産を生かしてもらえたなら幸いである．

謝　辞

当論文を執筆するにあたり，情報ネットワーク工学科プログラミングⅠ～Ⅳ担当の先生方である吉田教授，佐塚准教授，小路口准教授，足立講師，をはじめ当学科学生のみなさんに取材させていただきました．感謝いたします．

文　献

(1)　堀越，実態調査に基づく一般情報教育としてのプログラミング教育の検討，筑波学院大学紀要，14，pp. 87-100，2019
(2)　小松，プログラミング教育の問題と対策，文京学院大学経営学部経営論文集，25，pp. 83-104，2015
(3)　巨海ほか，久留米工大における物理学初年次教育の試み(「物理駆け込み寺」の実践報告)，久留米工業大学研究報告38，pp. 33-41，2016
(4)　須藤，授業時間外の学びを支援する「Picture塾」の取組と成果，久留米工業大学研究報告40，pp. 69-74，2018
(5)　Computer Languages History, https://www.levenez.com/lang/
(6)　List of C-family programming languages, https://en.wikipedia.org/wiki/List of C-family programming languages
(7)　Scratch - Imagine, Program, Share, https://scratch.mit.edu/
(8)　D. Saito et. al., Comparison of Text-Based and Visual-Based Programming Input Methods for First-Time Learners, Journal of Information Technology Education: Research，16，pp. 209-226，2017
(9)　J. Atwood, Why Can't Programmers.. Program?, https://blog.codinghorror.com/why-cant-programmers-program/

⑽ 高橋，日本でいちばんわかりやすいプログラミングのドリル，日本能率協会マネジメントセンター，ISBN978-4820726425，2018
⑾ プログラミングスキル判定サービス TOPSIC，https://products.sint.co.jp/topsic