

個別指導室におけるプログラミング必修科目の 学修支援

Learning Support for Compulsory Programming Subjects in the Programming Tutoring Room

吉岡紫*

丸山一貴†

YOSHIOKA, Yukari

MARUYAMA, Kazutaka

要旨

プログラミング科目の授業は、説明と演習を組み合わせた形態で授業を実施している場合が多く、演習時間は授業内容や例示されたプログラムの動作確認、課題を解く時間として想定している。しかし、情報学部情報学科の学生は、学習レベルのバラツキがあり、例示されたプログラムを使用した試行錯誤を行うことができない学生が存在する。これらの学生の学習場所として、個別指導室を設けている。

本論文では、個別指導室を利用している学生を3つのタイプに大別し、タイプ別の指導方法と、学生が苦手とする学習項目についての報告を行い、今後の授業運営の助けとすることを目的としている。学生を納得型、積極型、消極型の3つに分類し、それぞれの学習傾向と指導方針を整理した。苦手とする学習項目は、プログラミングの基本的な要素である「条件式」、「繰り返しによる和の計算」、「二重ループ」について取り上げる。具体的な利用者の事例として、個別指導室を利用した学生のうち2名にインタビューを行い、授業と個別指導室、個人学習をどのように組み合わせたかについて調査した内容を報告する。

1 はじめに

明星大学情報学部情報学科では、プログラミング科目を必修としている。授業資料・板書による説明と演習を組み合わせ、2時限を連続して使用する授業形態である。PC教室を使用して、すべての学生が同じ環境で演習することができ、演習時間は授業内容や例示されたプログラムの動作確認だけでなく、課題を解くことに使用することを想定している。しかし、学生の学習レベルにバラツキがあり、例示されたプログラムに変更を加え、どのように動作が変化するか、試行する学生は少ない。また、説明時間、演習時間に教員に質問することを苦手とする学生が一定数存在する。結果として、課題と演習を通じて学習内容を理解するという教員の意図が十分に反映されない。これを補うため、プログラミングの個別指導室を設けて、サポートを行っている。本論文では、2015年度から約5年にわたってプログラミング個別指導室で行ってきた学生指導の経験から、個別指導室を利用する学生を3つのタイプに大別し、学生が苦手とする学習項目とその特徴について述べ、大別したタイプごとの指導方法をまとめる。また、個別指導室を利用した学生のうち2名にインタビューを行い、授業と個別指導室、個人学習をどのように組み合わせたかについて調査した内容を報告する。本論文の目的は、これらの報告を通じて、今後の教員による授業運営と、個別指導室での指導について有益な情報を提供することである。以下、第2章でプログラミング個別指導室とその利用状況を紹介する。第3章ではプログラミング個別指導室利用学生の特徴について、第4章では指導を通して得られた知見について述べる。第5章で学生による個別指導室の活用例を、第6章で関連研究を、第7章でまとめを述べる。

2 プログラミング個別指導室

プログラミング個別指導室は、情報学部の必修科目であるプログラミング関連科目の学習場所として開室しており、主にプログラミングの初学者、プログラミングを苦手としている学生を対象としている。プログラミング関連科目の指導に加えて、実験科目のレポート作成支援、情報処理技術者試験の学習支援なども行ってい

*明星大学情報学部情報学科 †明星大学情報学部

る。本論文ではプログラミング必修科目の指導に限定して、指導内容や得られた知見について述べる。また、個別指導室において授業内容に沿った指導を実施するために、前期はプログラミングⅠ、後期は、プログラミング序論、プログラミングⅡの授業に参加している。2016年度、2017年度、2018年度は、プログラミング序論の授業を担当している。本論文には、授業における質問対応により得られた知見が含まれる。以降、プログラミング必修科目の授業内容の概要を紹介した上で、学年ごとの利用状況と、個別指導室における指導について述べる。

2.1 プログラミング必修科目の授業内容

情報学部の2014年度以降入学者向けのカリキュラムにおいて、必修のプログラミング科目は、プログラミング序論(1年後期)、プログラミングⅠ(2年前期)、プログラミングⅡ(2年後期)である。このカリキュラムでは2年次以降、2コース制を採用しており、プログラミングⅠの授業内容はコースによって若干異なり、プログラミングⅡは学習する言語が異なる。必修科目以外のプログラミングの授業は多数あるが、プログラミング個別指導室の基本的な指導対象者をプログラミング初学者、プログラミングを苦手と思う学生としているので、本論文では割愛する。

プログラミングの授業では、授業資料・板書による説明と演習を組み合わせている。学習する内容に合わせた例示プログラムを多数用意し、そのプログラムを利活用することで、演習課題を解くことができるように設計されている。

表1にプログラミング必修科目の授業内容を示す。プログラミング序論が最初の導入となっており、プログラミングⅠがその復習とプログラミング言語の高度な機能の紹介、プログラミングⅡはより高度な内容という位置づけである。

2.2 利用状況

プログラミング個別指導室は、原則として週4日、9:00から17:00まで開室しており、学生は自由に入室できる。表2に、筆者が着任した2015年度以後、2018年度までの個別指導室の4年間の利用者数、リピート率、複数期利用率を示す。利用者数は、学生数であり、延べ人数ではない。個別指導室は、プログラミングの学習以外に実験科目のレポート作成支援、情報処理に関連する資格試験の学習支援を行っており、データはすべての利用者の合算データとなっている。

個別指導室は、2015年度、2016年度は、PCを6台使用して活動を行った。2016年度後期に利用者が増えたので、指導室のレイアウト変更を行い、2017年度、2018年度は、最大10台のPCを使用可能な状態にした。表3に、2015年度から2018年度までの各学期の利用状況を示す。

2015年度は着任初年度であり利用者の傾向が少し異なるが、2016年度から2018年度について、授業ごとの類似点を以下に述べる。

1年の後期に開講されるプログラミング序論は、プログラミングの基礎的な文法を学習する授業であり、授業

表1 プログラミング必修科目授業内容表

授業名	授業内容
プログラミング序論	標準入出力、変数・リテラル、代入・参照、算術演算子、選択的実行、条件式・比較演算子、論理演算子、繰り返し処理、配列
プログラミングⅠ	標準入出力、変数・リテラル、代入・参照、算術演算子、選択的実行、条件式・比較演算子、論理演算子、繰り返し処理、配列(多次元)、関数、変数のスコープ、ポインタ、ファイル操作
プログラミングⅡ	<ul style="list-style-type: none"> ・C言語(*) アドレスとポインタ、ポインタによる配列操作、文字列操作、ポインタを使用した関数、構造体・共用体、スタック・キュー、再帰、ファイル操作、ビット操作 ・Java言語(*) Javaの基礎、メソッド、オーバーロード、標準入力、例外、ファイル操作、オブジェクト指向、Listとソート、GUIプログラミング

(*) プログラミングⅡはコースごとに言語が異なる。

表 2 プログラミング個別指導室 利用状況

利用者数合計	リピート率	複数期利用率
384	70%	49%

中の説明をきちんと理解することで、演習問題、小テストの問題を解くことができる。よって、個別指導室の利用者は少ない。不明点だけ質問する傾向があるので、学生のリピート率は低い。ただし、2016年度と2017年度は、後期にITパスポート対策講座を個別指導室にて実施しているため、利用者数が多く、リピート率が高い。プログラミングⅠは、授業において出題される課題が難しくなるため、個別指導室の利用者が増加する。毎週、課題の学習をするために、繰り返し利用するようになる。プログラミングⅡのJava言語を学習しているコースでは、レポート課題、テストのやり直し課題を、個別指導室を利用して行うよう授業において指導しているので、利用者が増加し、繰り返し利用するようになる。プログラミングⅠで個別指導室を利用している学生は、プログラミングⅡにおいても利用する傾向がある。

利用者の平均滞在時間は1時間30分、つまり1時限の授業時間に近い数値となっている。授業の空き時間を使用して、個別指導室を利用している学生が多いと考える。

2.3 個別指導室における指導

プログラミング個別指導室では、複数の学生が一つの部屋で学習する。指導員が一人、常駐して複数の学生の対応をしている。授業と同じ環境で学習できるPCを用意しており、学生はそのPCを使用してプログラムを作成することができる。指導員が決まった内容を授業形式で説明するのではなく、基本的に学生からの質問に対して必要な説明を行う。質問の内容は雑多であるが、いろいろな学生が説明を聞くことができるようにホワイトボードを用いて説明している。同じ問題を考えている学生が、一緒に説明を聞くことができるよう声掛けを行っている。なぜなら、同じような疑問を持ち、同じように間違えたプログラムを作成している学生がいることを認識できるからである。間違えることに恐怖心を持つ学生にとって、個別指導室の状況は「自分だけができていない」という負の気持ちを軽減させる効果がある。詳細は、第3章に述べる。

表 3 各学期の個別指導室の利用状況

学年	学生数	延べ人数	リピート率	平均滞在時間	学生数	延べ人数	リピート率	平均滞在時間
2015年度 前期				2015年度 後期				
4年以上	17	144	76%	1時間36分	12	229	83%	2時間05分
3年	16	57	31%	0時間47分	11	49	45%	1時間32分
2年	26	101	50%	1時間19分	29	68	48%	0時間58分
1年	3	3	0%	0時間10分	18	69	61%	1時間07分
2016年度 前期				2016年度 後期				
4年以上	13	70	77%	1時間47分	6	19	83%	1時間13分
3年	27	140	63%	1時間23分	14	103	79%	1時間38分
2年	26	127	69%	1時間21分	59	240	69%	1時間09分
1年	10	81	60%	1時間16分	25	139	88%	1時間15分
2017年度 前期				2017年度 後期				
4年以上	18	169	56%	1時間53分	12	79	75%	1時間34分
3年	28	286	86%	2時間13分	32	168	66%	1時間34分
2年	26	155	77%	1時間28分	57	288	65%	1時間33分
1年	31	66	48%	1時間23分	27	123	52%	1時間21分
2018年度 前期				2018年度 後期				
4年以上	25	90	76%	1時間31分	17	93	35%	2時間03分
3年	41	351	76%	1時間34分	29	207	66%	1時間50分
2年	29	245	66%	1時間30分	49	447	77%	1時間28分
1年	11	25	18%	1時間39分	8	30	50%	1時間01分

3 プログラミング個別指導室 利用学生の特徴

本章では、プログラミング個別指導室を利用する学生を3つのタイプに大別し、その特徴と指導方針について述べる。

3.1 特徴

個別指導室を利用する学生は、以下のタイプに大別することができる。

(1) 納得型

納得型の学生は、学習意欲が高く、理解したという納得感が得られるまで学習する。何かしら解決できない疑問点があり、その疑問を解決するために個別指導室を利用する。授業資料の整理やノートテイキングはできており、論理的に考える力はあるが、理解に時間がかかる。些細なことで思考が止まり、正しく理解してははずの事柄に疑問を持つことがある。多くの情報を与え、学生自身に必要な情報を取捨選択させることで理解を深めることができ、納得につながると考える。また、納得型の学生は、学習方法が確立しており、学習する場所の一つとして個別指導室を利用する場合がある。疑問が生じたときにすぐに解決することができるメリットがあるので、個別指導室で学習をする。

(2) 積極型

積極型の学生は、活発に質問をすることができ、一見、学習意欲が高いように見える。教科書や授業資料を参照して、例示されたプログラムを書き換えながら様々な試行錯誤を行うが、プログラムを書き換えた意図を説明できないことが多々ある。試行錯誤に根拠がなく、論理的な理解に繋がらない作業になっている場合がある。授業資料の整理やノートテイキングはできておらず、自分が不足している知識について、理解が進んでいない。自分自身の能力にある程度の自信があり、いろいろなことが上手くできていると誤解しているため、詳細な知識を修得するつもりがない。動作するプログラムを作成すること、演習問題を解くこと、とりあえず単位を修得することといった目先の課題解決だけが目的となっている。演習問題の答えとして、プログラムが正しいかどうかを質問することが多く、プログラムの動作を理解することには考えが及ばない傾向がある。このため、プログラムを作成する作業はできるが、すでに学習したことを活用できず、知識の定着が悪い。

(3) 消極型

消極型の学生は、プライドが高く、間違いを指摘されることを嫌い、他者の視線を気にする傾向が強い。結果として、プログラムを書き換えて試行錯誤する作業を行わない。教科書や授業資料を参照している様子はあるが、思考が進んでいない印象がある。とりあえず単位を修得したいという目的で来室するが、利用のきっかけが教員からの勧め、友人からの誘いなど、本人が主体的に考えていない場合が多い。教員から声をかけても、反応が少なく、何もしていない印象を受ける。教員の説明や資料がわかりにくいなど、他者に責任を転嫁するような発言が多く、自身の現状を客観的に判断する力が不足している。

3.2 指導方針

第3.1節において大別したタイプごとに指導方針を示す。

(1) 納得型

納得型の学生は、疑問点が明確になっており、本人にとって必要な情報が整理された状況で指導室を利用する。不鮮明なイメージを浮かべた状態で来室することが多いので、原理原則を説明した上で、図などを用いて詳細な処理を考えさせる。具体的なイメージと一緒に考えることで疑問点を解決する方法が有効である。ただし、取り組んでいる課題より前に扱った課題、すなわち知識の積み上げにおける前段階で、誤った理解やあいまいな理解をしているために、後段階で疑問点となっているようなケースがある。どの段階で混乱しているかについて、聞き取りを行うことが大切である。

(2) 積極型

短絡的にプログラムを作成しても、思い描いている処理ができないことを認識しているため、個別指導室を利用する。そこで、目先の問題解決ではなく、知識を蓄積できるようになることを目標として、指導を行っている。学生には、必ずノートを持ってくるように伝え、問題を解決するための思考過程を記述することを行わせている。疑問が発生した場合に、最初に自分で作成しているノートを確認する作業を行わせることで、詳細な情報をノートに記述しなければ、役に立たないことを認識させる。基本的に、会話をすることを得意としている学生が多いので、なるべく友達同士で問題点を考えるように指導している。大枠の説明を行い、その内容

が理解できる学生がいた場合は、その学生を他の学生に説明する担当者としている。学生同士で不明点を洗い出し、話し合うことで、より理解が深まると感じている。目標は、納得型の学生のような思考ができるようになることである。

(3) 消極型

消極型の学生は、間違えることに対する恐怖心が強いため、作業ができない。個別指導室では、プログラミングを苦手としている多くの学生が学習しており、同じような疑問を持ち、様々な間違ったプログラムを作成している。その状況を見ることで、間違えることに対する恐怖心が減ることが、大切であると考えられる。また、例示されたプログラムが自分の考えた通りに動作しないこと、授業資料が読めないことに対しては、自己の責任と考えない傾向がある。正しく動作するプログラムを少しずつ開示し、少しずつ期待する動作に近づけることを繰り返し行わせる。プログラムをどのように修正するべきかを、学生本人に考えさせ、例示されたプログラムを使用した試行錯誤ができるように指導する。目標は、積極型の学生のように、自分がどのように考えているかを説明しながら質問ができるようになることである。

4 指導における知見

本章では、学生にありがちな誤ったプログラム、指導内容、方法について特徴的と考えられる項目を3つ取り上げ、詳細に述べる。

4.1 条件式の組み立て方

(1) ありがちなプログラム

$a < x < b$ のような、複数の条件を論理積などで結合する場合、合理的な条件式や if 文を記述できないケースが多い。図 1 のプログラムは、「 x が 0 以上、100 以下で y が 50 以上 90 以下の範囲を外れる場合に out of range と表示する」という問題文の場合に作られたプログラムの一例である。本来は否定の ! やド・モルガンの法則を用いることが期待された問題だが、否定を用いず空の then パートと else パートを組み合わせて実現している。

繰り返しの場合、継続する条件を記述するよりも、繰り返しを終了する条件の方が理解しやすいためか、永久ループと break 文の組み合わせで実現するケースが見られる。図 2 は「キーボードから 0 が入力されるまで繰り返す」という問題に対する学生の解答の一例である。本来は、「キーボードからの入力が 0 ではない場合に繰り返す」という条件になる。繰り返す条件を記述できないため、図 2 に示すような永久ループと break 文を使用したプログラムを作成する。

どちらのケースも、条件の否定が考えられないことが、このようなプログラムを作成する原因の一つと考えられる。

(2) 指導方法

タイプごとの指導方針を以下に示す。

・納得型の学生

具体的な数値と図による可視化を用いて、条件を整理させることが有効である。指導員がホワイトボードに数値の区間を示す数値線やベン図を描き、いくつかの数値がどの領域に所属するかを考えさせる。その後、いくつかの条件式に具体的な数値を当てはめて、期待通りの真理値が得られるか試行錯誤させる。問題文に記載してある文面通りの条件以外に、期待通りの真理値になる条件が複数あることを理解させる。具体的な数値から、条件を自由に考えさせることが大切である。

```
if ( (0 <= x && x <= 100) && ( 50 <= y && y <= 90) ){
    // then part に処理が書かれていない
} else {
    printf("out of range\n");
}
```

図 1 then パートが空のプログラム

```

n=-1;
while(1){                                // 永久ループを使用
    scanf("%d",&n);
    if(n==0) break;                       // やめる条件を記述
                                           //何かしらの処理がある
}

```

図 2 永久ループと break 文

・積極型の学生
 納得型と同様に、具体的な数値と図による可視化を用いる方法が有効である。図 1 の場合であれば、x が 10 の場合、y が 100 の場合など、具体的な値を示しながら、動作を考えさせる。その後、数値の区間を示す数値線を作成させて、いくつかの数値がどのような領域に所属するかを考えさせる。学生が思いついていない条件をいくつか示し、異なる視点を持たせることが大切である。

・消極型の学生
 消極型の学生には、動作する条件を増やしていく方法が有効である。具体的な数値をいくつか例示して、どのような動作になるか考えさせる。論理演算子を使わず、単一の比較演算子で表現できる条件式を使って、まずは動くプログラムを実装させる。その上で、他の具体的な数値も扱えるように比較演算子を論理演算子で追加して、徐々に完成形に近づける。成功する条件を増やすことで、複雑な処理を組み立てることができるということを学習させる。

4.2 繰り返しによる和の計算

(1) ありがちなプログラム
 1 から n までの整数の和 sum を計算するような処理を、学習済みである繰り返しで実現することができない。「sum=1+2+3+...+n;」であることと、値が決まっていない n までの計算式を記述できないことは理解できるが、すでに学習済みである繰り返しとインクリメントの考え方を活用して、和を計算する処理を導き出すことが難しい。繰り返し処理を一つのパターンと認識しているため、繰り返し処理のループ制御変数の変化や繰り返しごとに変化する値について理解をしていない。また、「i++;」と記述するインクリメントを、i が 1 ずつ増加すると理解しており、「i=i+1;」の式の動作として理解できている学生は少ない。

(2) 指導方法
 タイプごとの指導方針を以下に示す。

・納得型の学生
 プログラミングでは、どのように和の計算を考えるか、詳細に説明することが有効である。納得型の学生は、完成形のプログラムを作成した上で、詳細な動作を知りたいと考えている。図 3 に示す和の計算の考え方を説明する。図 3 の左の例に示すプログラムを記述し、「=」を使用した代入式を利用することで、sum に和の値が代入できることを説明する。次に、i の値は 1 ずつ増加しているのので、図 3 の中央の例に示すようにインクリメ

sum = 0;	sum = 0;	sum = 0;
i = 1;	i = 1;	for(i=0; i<n; i++){
sum = sum + i;	sum = sum + i;	sum = sum + i;
i = 2;	i = i + 1;	}
sum = sum + i;	sum = sum + i;	
i = 3;	i = i + 1;	
sum = sum + i;	sum = sum + i;	
i = 4;	i = i + 1;	

図 3 和の計算の考え方

```

for( i = 0; i<10; i++){
    for(j = 10; j < 100; j=j+10){
        a[i]=j;           // ここは、10回しか動作しないと考える
    }
}

```

図 4 不要な繰り返し文

ントという考え方で代替できることを説明する。1 ずつ増加する処理は、繰り返し処理の中にすでに記述してある `i++`; で代用できないかヒントを与える。また、同じ処理が複数記述されているので、図 3 の右の例に示す繰り返し処理を使うプログラムを記述できるように導く。

・積極型の学生

納得型と同様に、プログラミングでは、どのように和の計算を考えるか、詳細に説明することが有効である。積極型の学生は、図 3 の左の例を記述すると、その場でプログラムを書き換えようと手を動かす傾向がある。見たことがある繰り返しの処理の雛形の中に、「`sum = sum + i;`」を記述することで、意味を理解することなく正しいプログラムを作成できる可能性が高い。そこで、プログラムを入力させる前に、図 3 の左の例をノートに記述させ、図 3 の中央の例になるように、学生本人にどの部分が繰り返しになるかを考えさせ、ノートに記述させる。その後、図 3 の右側のプログラムを考えさせてから、実際のプログラムを作成させることが大切である。

・消極型の学生

基本的な繰り返し文の動作を理解させてから、和の計算について説明する必要がある。繰り返し文の動作をきちんと理解できていない可能性が高いので、1 から `n` まで `i` をインクリメントさせて、表示するプログラムを作成させて、動作を確認させる。「`i++;`」と「`i=i+1;`」が同じ動作であること、繰り返し処理がどのような動作であるかを理解させることが大切である。また、繰り返し処理の中に、別の変数を利用した、値が変化する処理を加え、その変数がどのように変化するかについて考えさせる。どのように処理を組み合わせるべきかを考えることを苦手としているので、確実に動作する例示プログラムをいくつか作らせることが大切である。その後、積極型と同様の指導を行う。

4.3 二重ループの理解と利用

(1) ありがちなプログラム

二重ループの動作を誤解しているケースが多い。これは、繰り返しを用いた処理を慣用句のようなパターンとしてしか理解していないことが原因と考えられる。第 4.2 節でも述べた通り、繰り返し処理のループ制御変数の変化や繰り返しごとに値について理解をしていないからである。例えば、整数型の要素数 10 個の配列の中に 10、20、30、…、100 までの数値を格納するような場合に、図 4 に示すようなプログラムを記述する学生が多い。このプログラムは配列の 1 要素に繰り返し代入しており、また、最終的な各要素の値も正しくない。配列への代入も、10 から 100 までという規則的な数値の生成も、既習の題材では繰り返しにより実現されていたことから、学生はこれらを単純に組み合わせてしまうものと考えられる。発想することそのものは問題ではなく、この二重ループが期待通り動作するか、検証できないことが問題である。

図 5 に示すような、底辺の大きさをキーボードから入力させ、直角三角形を描画するプログラムは、二重ループの練習としてよく用いられる題材であるが、これを苦手とする学生も多い。いずれのケースも、二重ループの動作について、正しく理解していないことが原因の一つと考えられる。二重ループの中で、2 つのループ制御変

```

size: 4
*
**
***
****

```

図 5 直角三角形

```

size: 4          int i,n;
*****          printf("size:");scanf("%d",&n);    // 縦の大きさを入力させる
*****          for(i=0; i<n; i++){                // 縦の数を制御
*****              printf("*****\n");            // 横の大きさは5個固定
*****          }

tate: 4          int i,j,n,m;
yoko:5          printf("tate:");scanf("%d",&n);
*****          printf("yoko:");scanf("%d",&m);
*****          for(i=0; i<n; i++){                // ここで縦の数を制御
*****              for(j=0; j<m; j++){printf("*");} // ここで横の数を制御
*****          }
*****          printf("\n");

```

図 6 長方形を描く

数の値を表示して確認させても、値の変化と実際の動作の関連性が理解できないように見受けられる。二重ループの中で繰り返される処理は、内側の繰り返し処理の { } の中のみ繰り返すと理解している学生も多い。また、繰り返し処理の中で実行される処理をすべて説明する（ステップ実行の全過程を説明する）ような問題を出題すると、ほとんどの学生が正しく説明することができない。

(2) 指導方法

タイプごとの指導方針を以下に示す。

・ 納得型の学生

二重ループで使用しているそれぞれの変数がどのように変化しているかを理解させることが大切である。値の変化が可視化できるよう * を使用した描画を利用して考えさせる。図 6 の上半分に示すように、最初に printf() 文で "*****" を表示し、それをキーボードからの入力した数値の回数だけ表示することで、四角形を描くプログラムを考える。キーボードから入力した値が、描画のどの処理に影響するかを考えさせる。4 と入力すれば、縦が 4 個、横が 5 個の四角形が表示できる。次に、図 6 の下半分に示すように、横に表示している * の数をキーボードから入力した数値の数に変更する。外側の繰り返しの処理が縦の大きさ、内側の繰り返しが横の大きさを示していることを理解させる。その後で直角三角形について考える。図 7 に示すように、ノートに * を使用した直角三角形を描かせ、行と表示する * の数の関係について考えさせる。* の数の変化について、どのようにすべきか気が付くように、行の番号と * の数値を明記するようにしている。

・ 積極型の学生

納得型と同様に、二重ループで使用しているそれぞれの変数がどのように変化しているかを理解させることが大切である。積極型の学生は、詳細な処理を理解しているわけではないので、"*****" を繰り返すことで四角形を描いた後、"*****" を繰り返して描くことを考えさせる。"*" を複数回表示する繰り返し処理は、見たことがある繰り返し処理を流用して描くことができるが、改行の処理を加えることができない傾向がある。"*****" には、改行が 1 回のみであること、繰り返し処理が終了した後に実行することが必要であることを説明する。その後、"*" を横に複数回描く繰り返し処理を、どのように繰り返すことで四角形になるかを考えさせる。また、ノートに * を使用した四角形を描かせ、キーボードから入力した数値が変わることで、プログラムの中の数値が変化するかを考えさせる。その後、納得型の学生と同様に直角三角形の指導を行う。

```

1  *           // 1行目に1個
2  **          // 2行目に2個
3  ***         // 3行目に3個
4  ****        // 4行目に4個

```

図 7 直角三角形のノートへの記述

・消極型の学生

繰り返す動作を理解させることが大切である。まず、printf()文を使用した"*****"を表示するプログラムを動作させ、どこを繰り返すことで四角形になるか考えさせる。次に、"*****"を、繰り返しを使用して描く方法を考えさせる。この時に、縦に"*"を複数表示する繰り返しと、横に複数表示して改行を1回表示する処理を例示して、違いについて考えさせる。横に"*"を複数描き、最後に改行を表示する処理をかたまりとして認識させ、どの部分を繰り返すことで四角形になるかを考えさせる。その後、キーボードから縦と横の数を入力し、四角形の大きさを変化させるには、どこを変えれば良いかを考えさせる。繰り返し処理の動作を理解させるために、ヒントを与えながら学生本人に試行錯誤を行わせる。その後、積極型の学生と同様に直角三角形の指導を行う。

4.4 考察

消極型の学生は、答えやヒントを待つ姿勢が強いので、まずは小規模のプログラムで手を動かすよう導く必要がある。正しく動作するプログラムを作るためには、例示されたプログラムを利用すること、プログラムを少しずつ書き換える作業、つまり試行錯誤が必要であることを経験させる。また、間違えることはプログラムを作る上で大切であることを経験させ、手を動かすことに対する恐怖心をなくすことが必要である。加えて、一人では解決できない場合には、個別指導室において解決できることを認識させる。

積極型の学生は、ゴールを意識せず、理解ではなく正解を出すことを目指して作業するので、迷走する上に積み重ねもできない。しかし、計算機に向かって作業することには抵抗がないので、紙ベースの作業（可視化を含む）を取り入れて、作業の意味や効果を意識させ、理解することでゴールに近づけることを経験させることが必要と考える。

納得型の学生は、学習に対する姿勢や方法が確立しており、個人ごとに学習方法に工夫や思い入れがあるので、その方法に合わせる必要がある。また、学生自身に必要な情報を取捨選択する能力があるので、王道の考え方だけでなく、いろいろな考え方やアルゴリズムを説明し、学生が納得する考え方やイメージを取り入れられるようにサポートすることが大切と考える。

5 学生による活用例

本章では、個別指導室の利用頻度が高かった納得型の学生2名に対してインタビューを行い、個別指導室での学習を、授業と個人での学習へどのように活用したかについて、本人の同意に基づいて具体的な事例を紹介する。

5.1 事例1

本学生は、基本的な学習習慣があり、授業資料、教科書、参考図書を使用した学習により、自習により知識を定着させることができる。必要に応じて予習をし、自分が納得するまで復習することを習慣としている。

個別指導室の利用のきっかけは、プログラミングⅡに入ってから、プログラムを漠然としか理解していないことに気づいたこと、担当教員の話から詳細な理解が必要であることを認識したためである。プログラミングⅡでは、明星 LMS*に公開されている資料を授業前に印刷して授業に参加していた。資料に必要な情報を書き足し、資料に書き込めない量の情報や、自分で調査した内容はルーズリーフに記述して、資料と一緒にファイリングして情報を整理する方法で学習していた。

授業内容を整理できるまで質問しないことを決めており、基本的に授業時間は自分の力で理解を深める時間としていた。そのため、疑問点や不明点があっても、担当教員やTAに質問をすることはなかった。週末を利用して復習を行い、疑問点を整理した上で個別指導室を活用した。個別指導室においては、具体的な疑問点を質問することが多く、授業のサンプルを使った詳細な説明が有益であると感じていた。

5.2 事例2

本学生は、基本的な学習習慣があり、自習により知識の定着をすることができるが、プログラミングに対して苦手意識のある学生である。基本的に復習重視の学習方法であり、自分の疑問点を整理するために、ノートを作成する。授業資料が公開されている授業でも、ノートを作りこむことで、自分の理解を整理する学習方法である。

個別指導室は、プログラミング序論の授業内課題を理解するために利用を開始し、その後のすべてのプログラミングの授業において利用していた。授業中の理解を補うこと、演習課題の疑問点をなくすことを目的としている。疑問点は、その都度解決したい気持ちがあるので、授業中であれば担当教員やSAに質問している。自分が

*LMS (Learning Management System) : 学習管理システム

納得するまで理解をしたいという気持ちがあるので、繰り返し質問し、知識を積み上げ、少しずつ理解を深めていく。現在学習している項目から過去の学習項目に関する疑問が発生することが多く、その疑問を解決するために個別指導室を利用している。また、個別指導室は、自分のペースで学習できる場所、疑問が生じた際にその場で解決できる学習場所として、定期的に利用している。

5.3 考察

2名の学生には、授業資料やノートを自分のやり方に合わせて手元に用意し、それを活用することができる共通点がある。二人ともデジタルデータとして記録する方法、例えば、プログラムの中に説明を加えるような方法ではなく、高校生までの学習方法として確立しているノートを作成する方法を採用している。事例1の学生は、自分が納得できるまで学習することが必要であると考え、授業で得られた内容を掘り下げることを行っている。個別指導室を多くの知識を得られる場所として活用している。事例2の学生は、一つの項目を繰り返し学習することで、知識を定着させる学習方法を採用している。学習することにより増える疑問点を、その都度解決できる場所として、個別指導室を活用している。

それぞれ自分なりの学習スタイルが確立しており、授業時間ではサポートしにくい部分を個別指導室で補えたと考えられる。

6 関連研究

6.1 プログラミング初学者のつまずきの分析

プログラミング教育に関する先行研究は数多くあるが、学生つまずきとその対策という観点では、岡本ら[1]による取り組みが強く関連している。岡本らはプログラミング初学者が学習過程でつまずく要因について分析している。現在のプログラミング学習に関する教材はサンプルプログラムの模倣が中心であるとしており、(1)プログラムの記述と実行、(2)実行結果の認知、(3)結果の抽象化による理解という過程で学習が行われる。情報学科のプログラミング関連科目も演習形式を取っており、学生が取り組むこれらの過程を授業時間内に教員およびTA・SAにより支援している。また、これらの過程におけるつまずきの要因には以下の2つがあるとしている。

- (a) 学習内容の本質的な難しさが要因となる場合
- (b) 学習内容に付随する要素に起因する場合

要因(a)の例としてfor文が挙げられている。類似の概念をまだ学んでいないことや、for文を構成する要素が多く複雑であることが難しさの要因である。岡本らは、教材の題材選定や課題作成にあたって視覚的顕在化と呼ぶ考え方を反映することにより、学習者が学ぶべき内容を理解しやすくする手助けができるとしている。

情報学科のプログラミング関連科目でも、グラフィカルな出力による視覚的顕在化を活用した教材はあるが、納得型の学生でも過程(2)で満足してしまい、最も重要な過程(3)が不十分なままという印象を受ける。積極型の学生は、過程(1)は取り組めるが、過程(2)の認知が不十分である。テキスト処理を行うプログラムでは過程(1)と(2)における適切なプリントデバッグが有効だが、授業中の指導だけで取り組める学生は少ない。第3章で述べた個別指導室における指導方針は、これらを踏まえたものとなっている。

要因(b)の例としては学習支援環境の不備が挙げられており、コンパイラのエラーメッセージが分かりにくいことや授業資料の説明不足により、作成したプログラムと実行結果の対応関係が理解できないことが要因となっている。岡本らは、コンパイル手順やエラー発見の方法に関する手順を別冊子にまとめ、事例を充実させることが有効だとしている。

個別指導室の利用学生では、消極型の学生が過程(1)でこの要因によりつまずきやすい。授業内の演習の時間で教員やTAによりサポートしているが、質問することができずサポートを受ける機会を失っている。積極型の学生は質問することによってつまずきを解消してはいるが、場当たりに対処するため、類似の問題が発生したときに自力で解決できないことが多い。情報学科の授業資料でも、学習支援環境に関する補足情報を別冊子として学習内容の説明と分離することで、この問題が改善する可能性があると考えられる。

6.2 プログラミング入門教育の改善

第6.1節の要因(a)を改善する方法として、長ら[2]は1つの教材に多くの学習内容が盛り込まれていないかを確認するツールを提案している。個別指導室の利用学生に対しては、こうした学習内容を段階的に理解することを意図した課題設定が重要である。

要因(b)を改善する方法として、入門用のプログラミング言語[3]や手続き的なプログラミングの学習環境[4]といった先行研究がある。情報学科では、2019年度カリキュラムのプログラミング演習1でWebベースのPython学習環境[5]を導入しており、学習支援環境に関する問題を削減する取り組みを行っている。

7 まとめ

本論文では、2015年度から約5年にわたってプログラミング個別指導室で行ってきた学生指導の経験から、個別指導室を利用する学生を3つのタイプに大別し、学生が苦手とする学習項目とその特徴について述べ、大別したタイプごとの指導方法を述べた。また、納得型の学生2名にインタビューを行い、授業と個別指導室、個人学習をどのように組み合わせたかについて調査を行った。

個別指導室で学習することにより、プログラミングに対する負の気持ちが軽減することは、個別指導室の効果の一つと考える。また、本学部の学生は、第6.1節で述べた「(1)プログラムの記述と実行」がプログラミングの目的となっている場合があり、「(2) 実行結果の認知」、「(3) 結果の抽象化による理解」という過程に進むには、ノートを使用して自身の学習内容を記録して振り返る方法が有効であると考えられる。個別指導室において、学生が自身の学習方法を確立させ、知識が定着できるようになるよう、今後も学修支援を行っていきたい。

参考文献

- [1] 岡本雅子, 喜多一. プログラミングの「写経型学習」における初学者のつまずきの類型化とその考察. 滋賀大学教育実践研究指導センター紀要, Vol. 22, pp. 49-53, 2014.
- [2] 長慎也, 保福やよい, 西田知博, 兼宗進. De-gapper —プログラミング初学者の段階的な理解を支援するツール. 情報処理学会論文誌, Vol. 55, No. 1, 2014.
- [3] 兼宗進, 御手洗理英, 中谷多哉子, 福井眞吾, 久野靖. 学校教育用オブジェクト指向言語「ドリトル」の設計と実装. 情報処理学会論文誌プログラミング, Vol. 42, No. SIG11(PRO12), pp. 78-90, 2001.
- [4] 中村亮太, 西田知博, 松浦敏雄. プログラミング入門教育用学習環境 PEN. 情報処理学会研究報告コンピュータと教育, Vol. 2005, No. 104 (2005-CE-081), pp. 65-71, 2005.
- [5] 長慎也, 長島和平, 間辺広樹, 兼宗進, 並木美太郎. オンラインプログラミング環境 Bit Arrow におけるPython 処理系. 情報処理学会情報教育シンポジウム論文集, pp. 122-129, 2019.