情報Bにおけるプログラミング実践の報告

保福 やよい[†1] 兼宗 進[†2] 久野 靖[†3]

教科「情報」において、プログラミングを題材として取り上げる機会が増してきているが、そこに期待される教育効果や、その指導に際する問題点については、まだ十分に検討されているとは言えない状況である。筆者らは、「情報B」での実践において、興味深くプログラミングに取組ませるために、発見を伴う体験的な学習を行った。その内容と結果について報告する。結果として、生徒のモチベーションが向上し、意欲的にプログラミングに取組むなどの効果が観察された。

YAYOI HOFUKU,[†1] SUSUMU KANEMUNE[†2]
and YASUSHI KUNO[†3]

1. はじめに

30年以上前、「学ぶ」ことは、ものごとを暗記し、使えるようにすることだった。ものを知っていれば博識と言われ、尊敬された。一方、現代の私たちはパソコンや携帯電話、スマートフォンなどを使いこなし、日々大量の情報を扱う。情報は大量であり、すべてを知識として蓄えることは不可能である。必要な情報を的確に探し出し、そこから新しいものを産み出せる力が求められている。

しかし、実際の学校現場では、知識重視の教育が中心のままという現状がある。高校でも一斉授業による知識の伝達が効果的な授業方法と考えられている[1]。しかし、教科「情報」には、このような教え方はそぐわないと感じている。教科「情報」は新しさを失わない実験的な科目であり、今まで蓄えた知識と社会との関わりを実感し、自分で考える力を養い、知識を応用する力をつけることが可能である[2]。

筆者らは「情報社会の理解」、「問題の発見と解決に効果的な思考力」、「生徒の言語活動の充実」、「数学等の知識の活用」を実感できるような教材として、コンピュータ・サイエンス・アンプラグドの実習やプログラミングの作品を扱うことを考えた。

一方、プログラミングは高度な内容であり、それをどのような形で扱うのが適切か、どのような落とし穴があるかについて、授業実施前に十分に検討しておく必要がある。本稿では、教科「情報」の授業においてプログラミングを題材として実施た授業について、授業設計、授業実施、授業結果を報告する。

以下、第2節では、授業におけるプログラミングの実施に際してどのような課題があるかについて検討している。第3節では、今回の授業設計の土台となったコンピュータ・サイエンス・アンプラグドの考え方と、その利点をプログラミングの授業に取り入れる可能性について述べる。第4節では、生徒の興味を惹く教材の土台となったプログラミング言語について、第5節では授業設計と学習計画にについて、第6節ではプログラミングへの導入を意識したアンプラグドの実習について、第7節ではそれに引き続くプログラミングの実習について、設計、実施結果、評価を中心に説明する。最後に第8節ではまとめをおこなう。

2. ソフトウェアの仕組みを扱う難しさ

プログラミングを授業として実施するには「生徒の意欲・関心の問題」「達成感を得る難しさ」「教員の経験の問題」などの問題点が考えられる。本節ではこれらの問題点について考えていく。

2.1 生徒の意欲・関心の問題

初めて体験するものには、興味とともに、大きな不安を感じるのが普通である。ほとんどの高校生は、プログラミングの経験がないため、プログラミングに対する意識は、無関心か否定的なものになる。

平成21年度のプログラミングの授業に先立って、39人の生徒にプログラミングに対する印象を自由記載で書かせた。結果は、「難しいもの」と答えた生徒は24人、「プログラミングという言葉を初めて聞いた」と答えた生徒は12人であり、「楽しそうだ」と答えた生徒はわずか3人であった。このことからも、高校生にとってプログラミングは「よくわからないもの、難しいもの」という印象が強いことがわかる。

†1 神奈川県立相模向陽館高等学校 Sagami-Koyokan High School
†2 大阪電気通信大学 Osaka Electric-Communication University
†3 筑波大学 University of Tsukuba

2.2 達成感を得る難しさ

プログラミングの授業で難しいことのひとつは、生徒に達成感を持たせることである。教科書のプログラムを1行ずつ入力させて実行させることは簡単だが、それはキー入力の練習であり、プログラムの学習や、プログラミングの達成感には必ずしも結びつかない。

筆者のひとりは、以前の勤務校でJavaによるプログラミングを扱った。生徒は7人で、授業内容はプリントに載っているコードを入力し、改変するというものである。授業では、入力ミスによるコンパイルエラーが多発し、教員はトラブルシューティングに追われた。生徒も、プリントに載っているコードをそのまま入力するのが精いっぱいだった。生徒と教員の双方に無力感だけが残った授業であった。

2.3 教員の経験の問題

情報を担当する教員は、プログラミングの経験をほとんど持たないことが普通である。そして、校務や生徒指導で忙しい人が多く、新しいことを吸収し、実践するゆとりがない。その結果、「コンピュータは情報検索やワープロ、表計算、プレゼンテーションができればよい」と考えている教員も多い。授業の中でプログラミングを実践するためには、教えやすい言語が必要である。

授業をチームティーチングで行う場合には、副担当教員もプログラミングを扱うことになる。副担当教員は、授業時間数の少ない一般教科の教員が割り当てられることが多い。ワープロ、表計算、プレゼンテーションであれば一般教科の教員でも授業をサポートできるが、プログラミングを扱うことは難しい。そこで、教員にとっても理解しやすい言語が条件になる。

2.4 改善のための方策

前述の3種類の課題を改善するためには、次の条件を満たす必要がある。

- 生徒がプログラミングに興味を持てるようにする。
- 生徒が自分でプログラムをアレンジして達成感を持つことを可能にする。
- 専門知識を持たない教員でも授業を可能にする。

そのために、生徒と教員の双方にとって、易しく楽しみながら学べる教材があれば指導ができる講師陣が増え、工夫のある達成感の高い授業になると考えた。

そこで、コンピュータ・サイエンス・アンプラグド[3]（以下、アンプラグド）によって、ゲームを通じてプログラミングのイメージと必要性を実感させた後、初心者にも親しみやすい教育用プログラミング言語ドリトル[4]（以下、ドリトル）を使い、実習を行うことにした。

3. アンプラグド

プログラミングに対する苦手意識を取り去り、プログラミングを直感的に理解させるために、アンプラグドを活用した。

アンプラグドは、ニュージーランド カンタベリー大学のTim Bell博士がコンピュータを使わずに情報科学を体験的に学ぶ手法として1990年代に開発した[5]。2007年には日本でも翻訳が出版[3]されたため、情報の授業で扱えるようになった。現在は中学校[6]、高校[7][8]、大学などの授業[9][10]、小学生向けのイベント[11]等で数多く実践されている学習方法である。

アンプラグドは次のような特徴を持っている[12]。生徒たちが主体的に学習することができる。

- ゲームで楽しみながら学ぶ。
- 体を動かし体験を通じて学ぶ。
- グループで協力して問題解決。

友達と一緒にゲームやパズルを解くなかで，互いにコミュニケーションを取り楽しく学ぶことができる。また，楽しみながら学ぶ中で，好奇心を大いに刺激され，自分で考える中で様々な「発見」をする。アンプラグドで学ぶ中でブラックボックスであった情報科学の本質を理解することができる。

4. ドリトル

ドリトル[4][13]は、共著者である久野と兼宗が設計した教育用言語である。ドリトルはJavaで記述されており、Javaの実行環境（JRE）があれば動作する。小学校、中学校、高校、大学等[14]の授業で実践が報告されている。ドリトルには次の特徴がある。

- 日本語の文字による記述—プログラムは英語、日本語、韓国語で表記することができる。母語で書くのでプログラムの中身を理解しやすい。
- エラー対応—数字やアルファベットなどは全角・半角の区別をしない。そのためエラー出現の頻度が少なく、エラー時のコメントも親切であり、生徒がエラーを自力で解決することが可能である。
- 簡単な記述—ドリトルではグラフィックス、ゲーム、音楽、ネットワーク、ロボット制御などをたやすく扱うことができ、短い行数で高度な内容を記述できる。わずか10行で簡単なゲームの記述ができる。

表 1　学習計画　全 10 時間

アンプラグド	第 12 章「プログラミング言語」	1 時間
ドリトル	プログラム体験	1 時間
	ドリトルの基礎	2 時間
	自由課題制作	5 時間
	発表	1 時間

5. 授業設計

プログラミングの授業設計の要素として次のことを考えた。

- 生徒も教員も楽しめる授業内容
- 体験や発見を伴う授業内容
- 活発なコミュニケーション
- プログラミングの本質を理解

プログラミングに対して生徒の意欲・関心を引き出すことができれば、ほとんど成功と言える。そのためには、生徒も教員も楽しめる内容であること、さらに、楽しい授業のためには、知識を与えられるだけでなく、生徒自らがさまざまなことを発見することが必要であると考えた。そして、生徒同士が教え合ったり、互いの作品に刺激を受けお互い高め合うこと、また生徒が気楽に教員に質問できる環境にあればさらによい。そういった中でプログラミングの本質を理解してもらうような授業設計ができればこのうえない。

上記の要素を取り入れるために、プログラミングの授業の導入として、アンプラグド第 12 章「プログラミング言語」を行い、その後ドリトルでプログラミング体験、自由作品制作へと進んだ。全体の学習時間は 10 時間である。表 1 に学習計画を示す。次節からはそれぞれの学習内容について報告する。

6. アンプラグド第 12 章「プログラミング言語」による導入授業

6.1 実習の内容

アンプラグド第 12 章「プログラミング言語」を次のような手順で行なった。

(1)　2 人 1 組になり 図 1 の絵 (A、B) をそれぞれ一方にだけ配り、他方には見せないように指示する。

(2)　渡された絵を 図 2 のワークシートのマス目にできるだけ短い文章で表現する。

(3)　文章が出来上がったら、ワークシートを隣と交換し文章をもとに絵を描く。

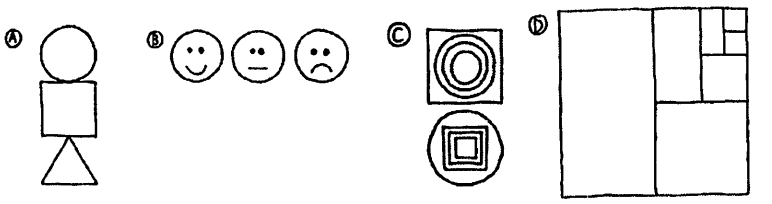


図 1　扱った絵 (A、B、C、D)

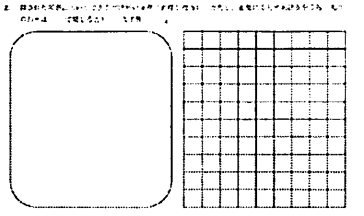


図 2　ワークシート

(4)　絵を描き終わったら隣の人にワークシートを返し、元の絵と比較する。

(5)　文章を発表する。

(6)　図 1 の絵 (C、D) についても同様のことを行う。交換相手を変えるのもよい。

ABCD と課題が進むにつれ難易度が高くなる。生徒は文章を書き、制限時間内まで推敲し、簡潔に表す方法がないか試行錯誤し続ける。生徒の文章の例を示す。

- 「丸を描き、その中に目と口だけをかいて、口のみを変えながら笑った顔とふきげんな顔とその中間のような顔を描く」(B の例)
- 「正方形があり、縦に 2 等分しその右半分に横棒をひき、上下に正方形をつくる。同じことを上の内部で 2 回行う」(D の例)

生徒は制限時間を目一杯使って、文字をぎりぎりまで削っていく。その中でさまざまな発見がある。推敲をこらした文章を書いただけに、自分の文章が正確に元の絵に復元できるかについて大変興味深そうだった。人によって文章の解釈が違うというのを視覚的に表すよい教材であった。

6.2 振り返り

実習の終わりに次のような質問をした。

- あなたは、どのようなことに気をつけて言葉にしましたか
- 絵を伝えるときに必要な要素はなんですか
- 相手はあなたの思う通りに絵に直してくれましたか
- 言葉を短くするためにどのような工夫をしましたか

これらの質問に対して、「簡潔にわかりやすく書くこと」「実行して調べること」「繰り返しを使うこと」「形、位置関係、大きさ、数など必要な情報を与えること」「全体をまず説明し、細部に入ること」「相手の立場にたつこと」などが大切な点として挙げられた。

プログラミングとは、コンピュータに私たちの実行したいことをコンピュータの言葉を使って伝えることである。この実習は、文章を書く方がプログラムを書く人、絵を描く方がコンピュータの役を演じているとも考えられる。従って、これらの大切な点は、文章を的確に表現するだけでなく、プログラミングの大切な考え方としても役立つのではないか。

アンプラグドによる導入授業により、生徒と副担当教員は、プログラミングに肯定的な考えを持った状態で次の実習に臨むことができた。

7. ドリトルによるプログラミング授業

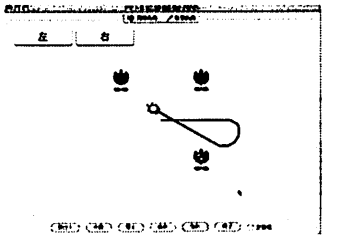
7.1 プログラム体験

実際のプログラミングを体験させるため、**図3**の「宝物拾いゲーム[*1]」のプログラムを1行ずつ編集画面に入力し、その都度実行させた。実行すると、図3のように、カメが動いた跡を残しながらチューリップを消していく。カメは左右のボタンでコントロールする。わずか10行のプログラムで簡単なゲームができ、1行入力して実行するたびに発見がある。

生徒は、1行1行の命令が何を意味しているのかの説明はなくても、プログラムを理解していたようだ。特に説明しなくても、競うように、チューリップの数を増やし、かめたのスピードを変えていた。短い体験ではあったが、コンピュータゲームのようなソフトウェアがどのように作られ、プログラムとは何をするものかを、楽しく理解できていた。

7.2 宝物拾いゲームの感想

「宝物拾いゲームを入力し実行したことで、感じたことを自由に書いてください」という質問に対して、1クラス生徒31名の回答結果を得た。**表2**に結果を示す。「宝物拾いゲー




```
かめた＝タートル！作る。
左ボタン＝ボタン！"左" 作る。
左ボタン：動作＝「かめた！30　左回り」。
右ボタン＝ボタン！"右" 作る。
右ボタン：動作＝「かめた！30　右回り」。
タイマー！作る「かめた！10　歩く」実行。
タートル！作る "tulip.png" 変身する　ペンなし　100 100 位置。
タートル！作る "tulip.png" 変身する　ペンなし　100 -100 位置。
タートル！作る "tulip.png" 変身する　ペンなし　-100 100 位置。
かめた：衝突＝「　！相手！　相手！消える」。
```

図3 宝物拾いゲーム

表2 宝物拾いゲームの感想（選択。複数回答あり）

楽しい	19名
自分でもできる	7名
簡単	4名
日本語で入力	3名
難しかった	1名

表3 宝物拾いゲームの感想（自由記述）

プログラミングの楽しさ、どんな風にでも変えることが可能でとても興味のあるものだった。
何もない所からものを作り出す楽しさ。とにかく面白い!!
脳が命令を出して私たちが字を書けるように、パソコンでかめたに「歩く」という動きをさせることが出来て面白かった。
「走る」と入力しても命令できなくて、命令に登録されていない言葉はできないということがわかった。

*1 カメ太の日記「1時間で学ぶソフトウェアの仕組み」 http://kamemune.eplang.jp/diary/2008-11-06-1.html

ム」の実施前には、「難しい」24名、「プログラミングという言葉を初めて聞いた」12名、「楽しそうだ」3名であったことを考えると、大きな変化があった。表3に、生徒の自由記述の感想を示す。生徒が、自分たちでプログラムを作れるという体験に興奮している様子が伝わってくる。

プログラミングとはどんなものか、ドリトルで何ができるかを事前に体験させることは生徒のモチベーションを高め、全体像を見せることで今後の作品作りに役立った。

7.3 ドリトルの基礎

宝物拾いゲームに出てきた、ボタン、色、変身、アニメーションなどについての練習問題を交えながら自由課題制作に無理なく取組めるよう基礎がためをした。「宝物拾いゲーム」ではなんとなく理解していた命令も、ここでしっかりと自分のものになった。ここで作った課題を発展させ、自由課題に仕上げていった生徒もいた。ここで気をつけるのは、あまり複雑なことに入り込まないことだ。複雑なことに拒否反応を示す生徒もいる。「よくできた」「楽しい」というイメージを壊さないよう、自由課題につないでいくように気をつけた。

7.4 自由課題

ゲーム、アニメーション、アニメーション＋音楽などそれぞれテーマを決め、事前に仕様書を提出し5時間の予定で自由課題に取組んだ。新型インフルエンザ流行のため、学年閉鎖や出席停止者も多く授業が円滑に進まなかったが、補講を行うことで補った。自由課題の内容は9割近くがアニメーション（音楽付きも含む）、1割がゲーム作成であった。

7.5 発表と相互評価

実行画面を1人ずつプロジェクタに写し、簡単な紹介を各自で行った後、相互評価を行った。相互評価の方法は、クラスを2つの班に分け、班内の作品について「わかりやすい内容であるか」、「面白いか」「総合的によいか」という3つの観点について「大変よい：4」「まあまあよい：3」「あまりよくない：2」「よくない：1」の4段階で評価させた。ユーザインタフェースを意識するようにということは作品製作中から伝えていたことであり、「わかりやすい内容であるか」という項目を設けた。

発表に向けて準備する段階で、生徒のモチベーションの向上が見られ、発表中は自分の作品を誇らしく紹介することでの達成感を再確認し、様々な作品の紹介を楽しみ、プログラミングへの興味が増したようだった。作品を鑑賞する中で、プログラム画面を見ている生徒もいた。

相互評価の平均は、「わかりやすいか」が3.5、「面白いか」が3.2、「総合的によいか」が3.4となった。生徒は、周りの作品を好意的に捉えていたことがわかる。

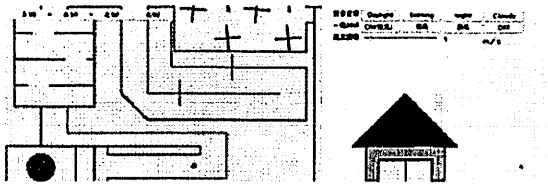


図4 生徒作品の例

7.6 生徒作品の例

図4に、生徒の作品例を示す。左の作品は迷路である。上部の4つのボタンはカメの角度を右左30度、右左90度変えることができる。カメは数ヶ所にある赤い壁を消しながら進んでいく。最初は簡単に進むことができるが、ゴール近くにはいくつもの風車が回っていて、カメをはね飛ばすので、難易度が高くなっていく。このようなゲーム性に富んだ迷路ではあるが、授業で教えたことだけを上手に組み合わせて作っているのに驚く。55行のプログラムの中には、衝突命令、ボタン、タイマーなどがきちんと配置され無駄が少ない。

図4の右の作品は風力発電と題して風車を実現した。風車の速さは「微風」「強風」「暴風」というボタンまたは「風速調整スライダー」で変えることができる。また、天気に関連付けた4つのボタンで背景色を変えることができる。「風速調整スライダー」では風速を数値で表現できるなど、教えた内容を越えた、難易度の高いものを記述している。この生徒は理科や環境教育に興味を持っていて、自分の思う内容がプログラミングで実現できて非常に満足そうであった。

7.7 授業アンケート

授業がすべて終わった後に「情報社会への理解」「プログラムの理解」「プログラムへの興味、関心」「授業への取組み状況」「今後の学習」を調べるために以下のようなアンケートを実施した。アンケートの項目は37項目あり、1クラス31名より回収することができた。それぞれの質問には「よくできた」「まあまあできた」「あまりできない」「できない」の4段階の選択肢を設けた。質問項目の内容で分類してアンケートを記載する。

7.7.1 情報社会への理解

表4に「情報社会への理解」に関するアンケート結果を示す。プログラムを実際に組んでみることで、便利なものの裏側にはコンピュータがあり、その1つ1つのプログラムは人間が作成したものという理解ができたと考えられる。さらに、人間が作ったものだから、

表 4 「情報社会への理解」に関するアンケート結果

質問	よくできた	まあまあできた	あまりできない	できない
Suica やオンラインショッピングなどは人間がプログラムを作ってできたシステムで故障することもあり得るということは理解できましたか	22	8	1	0
オンラインゲームは人間がプログラムしたゲームで、アイテムはデータであり、消えてしまったり、うまく動かないこともあることは理解できましたか	26	5	0	0
炊飯器やエアコンの中にはマイコンが入っており、人間が作ったプログラムで制御されていることが想像できますか	10	14	6	1

間違いがあるかもしれない、ということに気づいた生徒が多い。

7.7.2 プログラムの理解

表 5 に「プログラムの理解」に関するアンケート結果を示す。自由課題に取組むことで、基本的なプログラムの考え方を理解できた生徒が多い。命令がうまくいかない、そんな時には命令を小分けにして考えていくと、今まで使ったことのある命令の組合せでうまくいくことが多い。また、どうしてもうまくいかない時は、違ったやり方を試してみることもよい。答えは 1 つではない。物事を手順に分けて考えることはプログラミングのみならず、すべての問題解決の基本である。教科「情報」の中で、プログラミングの授業を通して体験的に学ぶことが可能である。

プログラミング自体は平易ではないという理解の中、体験的に多くのことを発見することでほとんどの生徒が「できた・わかった」という実感を得ている。

7.7.3 プログラムの興味・関心

表 6 に「プログラムの興味・関心」に関するアンケート結果を示す。多くの生徒が、他の人の作品に刺激を受け、自分で作品を作ることを楽しいと考えていることがわかる。自分の作品を上質のものに仕上げていく過程の中で、さまざまな発見をする。たとえば、「ふわふわ飛ぶ」「ボールを投げる」というアニメーションを入れたがる生徒が何人かいた。そこで必要な関数を紹介すると大変喜んで自分の作品に取り入れていたのが印象的だった。数学では単なる問題の 1 つだった、サインカーブや放物線が実際に使える知識になり、自分の作品の中で使うことで、生きたものになり数学の必要性も感じることができる。

プログラミングで作品を作ることは、絵を描くことや、曲を作ること、文章を書くことと似ているが、発見したり、他者から受ける刺激については、これらを越える創造的な取組み

表 5 「プログラムの理解」に関するアンケート結果

質問	よくできた	まあまあできた	あまりできない	できない
プログラムを書く時の手順を考えることができましたか?	12	15	4	0
プログラムの手順を楽しく考えることができましたか	17	13	1	0
物事を手順に分けて考えることが重要であるとわかりましたか	16	15	0	0
プログラムは上から順に実行することが学べましたか	21	9	1	0
プログラムの 1 行 1 行がどのような働きをするのか考えることができましたか	21	9	1	0
プログラムの基本的な命令の内容を理解することができましたか?	12	17	2	0
条件分岐を使う必要があることを理解しましたか	23	6	2	0
アニメーションをするとき、タイマーを使う必要があることを理解しましたか	26	5	0	0
プログラミングの授業は理解しやすいですか	4	15	11	1
できた・わかったという実感がありましたか	16	15	0	0
自分でプログラムを書くことができましたか	19	8	4	0
自分のプログラムを理解することができましたか	16	15	0	0
人のプログラムを見て、簡単なアドバイスをすることができましたか	4	14	13	0
マニュアルを調べて自分のプログラムで使うことができましたか	15	11	4	1

表 6 「プログラムの興味・関心」に関するアンケート結果

質問	よくできた	まあまあできた	あまりできない	できない
プログラムの作成に関心が持てましたか	13	15	3	0
プログラミングの授業は楽しいと思いますか	11	16	4	0
自分の作成したプログラムの動作結果を見るのは楽しいですか	15	11	4	1
自分のことがすごいと思った瞬間がありましたか	4	10	16	1
他人の作ったプログラムがすごいと思い、自分も頑張ろうと思いましたか	20	11	0	0
プログラミングの授業では好奇心を刺激されましたか	14	14	3	0
プログラミングの授業は充実していましたか?	15	12	4	0

表 7 「授業への取組み状況」に関するアンケート結果

質問	よくできた	まあまあできた	あまりできない	できない
思い通りのプログラムができましたか	6	17	7	1
プログラム作成の時間は十分でしたか	7	11	12	1
補講に出席しましたか？	20	7	0	4
思い通りのプログラムができるまで頑張りましたか	18	11	2	0
授業中、生徒・教員とのコミュニケーションはありますか？	21	10	0	0
休まずに出席しようという意欲が起こる授業でしたか	21	10	0	0
授業での自分の参加態度は積極的でしたか	14	15	2	0

表 8 「今後の学習について」に関するアンケート結果

質問	よくできた	まあまあできた	あまりできない	できない
プログラミングを学習することは重要だと思いますか	11	14	6	0
現在の時点で、プログラミングの知識・技術は身についていると思いますか	2	11	14	4
もっとプログラミングの勉強をしようと思いますか	6	13	11	1
授業で学習したことをもとにして、自分で工夫し勉強してみようと思いますか	4	15	11	1
プログラムを学ぶことは将来に役立つと思いますか	7	15	9	0

が可能になる。

7.7.4 授業への取組み状況

表 7 に「授業への取組み状況」に関するアンケート結果を示す。プログラムの授業では、生徒同士、そして教員との間に、活発な交流があったことがわかる。副担当教員も、生徒の頑張りを認め、作品の出来に素直に驚き、一緒に喜ぶことで生徒のやる気が増していた。プログラムの授業では、静かにキーボードの音だけがひびくケースもあるが、この授業においては楽しみながら、生徒が積極的に取り組む授業になった。

7.7.5 今後の学習について

表 8 に「今後の学習について」に関するアンケート結果を示す。自分の作品を仕上げるために、でき上がるまで何度でもプログラムを書き直したり、考え方を修正したりして粘り強く取組んでいることがわかる。

7.7.6 生徒の学習過程と達成感

絵を描くこと、曲を作ること、文章を書くことなどと比べてプログラミングが大きく違うのは、何度もエラーに見舞われて小さな失敗を繰り返し体験することだ。授業中に観察されたエラーの種類を表 9 に示す。学習の初期段階では、構文エラーが多く、プログラムに必要な「。」、「！」の欠損や変数名の間違いによるものが多かった。当初は教員の助けを借りることもあったが、次第に自分や友達同士で解決できるようになり、作品制作の後半では構文エラーによる質問がほとんどなくなった。これは、日常使う自然言語と、プログラムで使う人工言語の違いを学習したと考えている。

表 9 作品製作中に発生したエラー

エラーの種類	割合 (%)
構文エラー	60%
実行時エラー	25%
論理エラー	15%

次に多いのは実行時エラーであった。プログラム実行中に止まるこのエラーは、変数名の間違いによることが多かった。実行時エラーは長いプログラムを書いている中で発生するため、自力解決が難しい生徒もいた。

作品制作も終盤になってくると、「プログラム的には正しいが、結果が自分の意図した動作にならない」という論理エラーによる質問を多く受けるようになった。ある程度学習が進むと、生徒はそれぞれ自分のプログラムの論理を考えるという段階に進んで、かなりのエラーを自力で解決するようになった。その結果、完成させたときも自分の考える力によって完成させたという達成感を持つことができた。

教員に対してほとんど助けを求めず、自力ないしは友人の助けを借りてエラー解決をしていた生徒も 1/4 ほどいた。

あらゆる教科の中で、ここまで贅沢に失敗を体験させてくれる教材はない。小さいエラーを何度も体験し、解決してくことで、自信がつき、もう少し大きいエラーが出会っても、乗り越えられるのだ。そうして仕上げた作品だけに、愛着が湧き、深い充実感を感じていた。また、作品を仕上げるまでにさまざまな発見がある。授業で教えたことだけを使うのではなく、必要があれば自分で調べ解決することを通して、自発的な学習を行うことができた。

高校生に適しているプログラミング環境は、生徒の想像力を刺激し、自分でもできると思わせることが大切であるだけでなく、適度に失敗させる開発環境が適していると感じている。失敗を乗り越えた先には、大きな喜びがあり、問題解決能力も身につく。そういう意味では適度な難易度の教材が必要である。

8. おわりに

第2節で解決すべき課題として「生徒の意欲・関心の問題」「達成感を得る難しさ」「教員の経験の問題」を挙げた。それぞれの課題について振り返る。今回の授業設計と学習計画は教科「情報」の授業で有効であり、当初の課題を解決することができた。

- 生徒の意欲・関心 — 生徒たちは、授業前にプログラミングについての関心をほとんど持っていなかった。しかし、アンプラグド・ドリトルの授業を終えた後、「プログラムの作成に関心が持てましたか」「プログラミングの授業では好奇心を刺激されましたか」という質問に対して90％近い生徒が肯定的に答えている。さらに、「もっとプログラミングの勉強をしようと思っていますか」という質問に対して60％近くの生徒が肯定的に答えている。これにより、授業を通して高い「生徒の意欲・関心」が生まれたと考えている。
- 達成感 — 以前行った、プログラムコードを写し実行するだけの授業では、生徒はほとんど達成感を得ることができなかった。しかし、アンプラグド・ドリトルの授業を終えた後、「プログラミングの授業は充実していましたか」「他人の作ったプログラムがすごいと思い、自分も頑張ろうと思いましたか」の質問に対して90％以上の生徒が肯定的に答えている。これにより、授業を通じて強い「達成感」が生まれたと考えている。
- 教員の経験 — 今回行った授業と使用した教材は、広く高校の授業で利用することが可能である。特に、アンプラグドの導入授業（1時間）とドリトルによるプログラム体験（1時間）は、2時間だけで生徒の意欲・関心を高め、高い達成感を与えながら、ソフトウェアの仕組みを理解させることが可能である。そして、この範囲であれば、教員は特段のプログラミングに関する専門知識を必要としないことは大きな利点である。実際、今回の副担当の教員はプログラミングの経験はまったく持たなかったが、アンプラグドの授業を通じて生徒と同様にプログラミングに対して肯定感を持ち、さらにプログラム体験でプログラムの概要を理解し、今までの情報の授業よりも積極的に、生徒と授業で関わることができた。

アンプラグドによる「プログラミング言語」の導入授業では、生徒から「最初に全体を説明してから、細部を説明するのがよい」という意見がでた。これはそのまま、プログラミングの授業にあてはまる。

今回は最初に「宝物拾いゲーム」でプログラム体験をさせた。まず全体像を見せてから、次に細部の説明に入った。このことで、生徒は「できること」がある程度想像でき、自分の作品制作にスムーズに入っていくことができた。この方法は「教えすぎる」ことを避けることができる。「教えすぎる」と詰め込まれたような気がするため、生徒は無意識に受身になり、能動的に学ばなくなってしまう。今回の授業では、「教える」より「発見する」ことを重視することで、生徒のモチベーションを高め「達成感の高い授業」を実現できた。

筆者の1人は本稿の授業実践後に転勤し、現在は昼間の多部制定時制高校に勤務している。現在の高校では学習面、生活面で課題を抱える生徒が多い。来年度開講予定の情報Bの授業では、異なる環境にある生徒にとっても今回の授業設計と学習計画が有効であるかどうかを試し、効果的な学習方法を探っていきたい。

参考文献

1) 小泉力一, 佐藤義弘:「全国アンケート調査で見る情報科教育の現状」. 情報処理, Vol.50, No.10, pp1005-1008, 2009.
2) 中野由章, 和田勉:「新学習指導要領とこれからの情報教育」. 情報処理, Vol.50, No.10, pp996-1004, 2009.
3) 兼宗進監訳:「コンピュータを使わない情報教育 アンプラグドコンピュータサイエンス」. イーテキスト研究所, 2007. http://dolittle.eplang.jp/?unplugged
4) プログラミング言語「ドリトル」: http://dolittle.eplang.jp/
5) Tim Bell: Computer Science Unplugged. http://csunplugged.org
6) 井戸坂幸男, 西田知博, 兼宗進, 久野靖: 中学校における CS アンプラグドの授業提案. 情報処理学会 コンピュータと教育 (CE) 研究報告, No.98, 2009.
7) 保福やよい, 井戸坂幸男, 兼宗進, 久野靖: 高校情報 B における CS アンプラグドの活用. SSS2008, 2008.
8) 間辺広樹, 並木美太郎, 兼宗進:「高校の文化祭における「CS アンプラグド企画」の実践報告と課題」. 情報処理学会 コンピュータと教育 (CE) 研究報告, No.103, 2010.
9) 和田勉:「アンプラグドコンピュータサイエンスと板書講義を併用した大学でのアルゴリズムの授業」. 情報処理学会 コンピュータと教育 (CE) 研究報告, No.100, 2010.
10) 兼宗進, 佐藤義弘:「情報科教育法での CS アンプラグドの利用」. 情報処理学会 コンピュータと教育 (CE) 研究報告, No.103, 2010.
11) 井戸坂幸男, 青木浩幸, 兼宗進, 久野靖: コンピュータサイエンスアンプラグドの小学生向け実践の取り組み. SSS2008, 2008.
12) 西田知博, 兼宗進:「コンピュータ科学を楽しく学ぶ」, 情報処理, Vol.50, No.10, pp980-985, 2009.
13) 兼宗進, 阿部和広, 原田康徳:「プログラミングが好きになる言語環境」. 情報処理, Vol.50, No.10, pp986-995, 2009.
14) 間辺広樹, 並木美太郎, 兼宗進:「障害者職業能力開発校における情報教育の取り組み」. SSS2008, 2008.