

「プログル情報」を活用したプログラミングの授業実践

岐阜県立大垣南高等学校 教諭
高木 貴光

1. はじめに

「プログラミングの授業」というと、教材の準備や環境構築、指導法などで苦労するという印象がある先生方も多いのではないのでしょうか。本稿では、授業準備や展開に悩む先生方の参考に少しでもなればと思い、本校におけるプログラミングの授業実践(単元の展開例)を紹介します。

2. マイクロビットを使った導入

まずは、マイクロビット(micro:bit)を活用し、生徒にプログラミングの楽しさを伝える実践を行いました。マイクロビットは、センサーやボタン、LEDを搭載したプログラミング学習に最適な教材です。

画面内や紙面上だけで完結する従来のプログラミング学習では、生徒が達成感を得にくいという課題がありました。しかし、実物を動かして光らせたり、音を出したりする体験を通じて、プログラミングの楽しさをダイレクトに演出できたと感じています。まずは「楽しい」という成功体験を提供し、「プログラミングは難しい」という先入観を払拭することが重要だと考えています。

生徒からは「懐かしい」「中学校の時に使ったことがある」といった感想が寄せられ、中学校での学習基盤をもとに、高校でも指導をスムーズに展開できる環境が整いつつあると実感しました。

実習では、まず「MakeCode」を用いてLEDの点灯パターンを作成することから始めました。これは、プログラミングにおける「Hello World(最初の初歩的なプログラム)」に相当する導入です。生徒は、各自がデザインしたパターンを実機に転送し、実際にパターンがLEDに表示されるだけで大きな歓声を上げていました。

次に、マイクロビットのスピーカーを活用して音を鳴らす実習を行いました。PC室が一斉に電子音で賑やかになりましたが、これこそが生徒が主体的

かつ楽しんで取り組んでいる姿の表れであると感じます。「ボタンを押すと特定の音が鳴る」「本体を振るとランダムに数値が出力される」といったサンプルプログラムを通じ、多様な動作を体験させました。

3. 「プログル情報」を使って本格的なプログラミングへ

プログラミング実習の実施にあたり、ソフトウェアのインストール、授業計画の策定、教材の準備、課題の進捗確認など、教員の負担は多岐にわたります。

オンライン教材である「プログル情報」は、一部の高度な実習には対応していないものの、プログラミングの入門期における教材としては極めて十分な機能を備えています。指導内容も充実しており、「レッスン1」では基本的なコードの記述方法をスモールステップで学ぶことができます。続く「レッスン2」ではWeb APIを利用した天気予報情報の取得、「レッスン3」ではデータベースの活用といった実践的なカリキュラムが組まれています。さらに、課題の解答例がPDFで公開されているため、プログラミング指導に苦手意識を持つ教員でも、安心して授業を展開できる点が非常に優れています。

4. レッスン課題の作り方

本校の実践においては、生徒の習熟度を考慮して教材の「レッスン1」のみを共通実施とし、「レッスン2」以降は教員が自作した独自の課題に取り組む形をとっています。本教材には「問題集管理機能」が備わっているため、この機能と自作した課題を用いることで、学習ステップを教員側で再構成し、習得状況に最適化したオリジナル教材を、順序立てて提供することが可能となっています。

具体的な実践例として「おみくじプログラム」の作成が挙げられます。この実習では、次のようにプロ

グラムを段階的に改良(パワーアップ)させていく手法をとっています。

- ① 乱数を生成して整数の数字を出力する。
- ② 1次元配列を用いて、①で出力した数字を添え字として、対応するメッセージを抽出する。
- ③ 2次元配列へと拡張してパラメータ(運勢やラッキーアイテムなど)を増やす。
- ④ おみくじの出現確率を調整する。

ここには、「最初から完成されたコードを目指すのではなく、段階的な試行錯誤を通じて学ばせる」という指導上のこだわりがあります。作成の過程で「この機能を拡張したい」「無駄な記述を整理したい」「予期せぬエラーへの対処に困った」といった経験をさせることで、次の学習課題や深い理解へと繋げています。

5. 個別指導へのプロセス

従来のプログラミング実習では、最終的な実行結果のみで評価せざるを得なかったり、机間巡回時に偶然見つけた生徒にのみ指導が行われたりしがちでした。そのため、1対多の環境で全員のコードをリアルタイムに把握することは極めて困難でした。

こうした課題に対し、「プログレ情報」の提出機能(管理画面での解答確認機能)は非常に有効です。教員は、生徒が提出したプログラムを手元の端末から即座に確認できます。そのため、エラーで躓いている生徒や、早期に課題を達成した生徒に対し、状況に応じた的確なチェックや個別指導が可能となりました。さらに、多くの生徒が共通して間違えている箇所を発見した際には、一度全体の作業を止め、クラス全体で共有して考える時間を設けるなど、集団指導と個別指導を柔軟に組み合わせた効率的な授業運営を実現しています。

ここで、次の課題を受けて生徒が記述したコードから、思考プロセスを見取った具体例を紹介します。

出題した課題

底辺が5、高さが8の三角形の面積を表示するプログラムを作成せよ。

画面上の実行結果はすべて同じ「20」となるが、プログラムの内部構造を「プログレ情報」の管理画面で確認すると、生徒の理解度に応じてケース①～③のような違いが見られた。これらに対し、個別に次のような声かけや指導を行っている。

ケース①

```
1 print((5 * 8) / 2)
```

見取り：変数の概念を全く使用せず、単なる電卓の計算式として記述している。

指導アプローチ：「計算は合っているけれど、もし底辺が『10』に変わったら、また式を書き直さなきゃいけないよね。数値を何度も変えられるように『変数』という箱を使ってみよう」と促す。

ケース②

```
1 A = (5 * 8) / 2
2 print(A)
```

見取り：面積を格納する変数(A)は使えているが、底辺と高さが定数(固定値)のみである。

指導アプローチ：「面積を変数にしたのは素晴らしい！じゃあ、底辺の『5』と高さの『8』も、それぞれ別の変数(箱)に入れて、その変数同士を掛け合わせる式に直せるかな？」とステップアップを導く。

ケース③

```
1 ans = (5 * 8) / 2
2 print(5 * 8 / 2)
```

見取り：変数(ans)を定義したものの、出力(print)の段階でそれを利用せず、再び定数で再計算させている。

指導アプローチ：「せっかく ans という変数の中に計算結果を保存したから、print の中身は数字の計算じゃなくて、その変数の名前(ans)をそのまま使ってみよう」と、変数の参照について気づかせる。

6. おわりに

プログラミングの授業は何かと大変かと思いますが、少しでも日々の授業づくりのヒントになればと思い、本校の実践を紹介させていただきました。

最近では、AIがプログラムを自動生成してくれる時代ですが、基礎が理解できていなければ、それを活用することはできませんし、AIが誤ったコードを出力することもあります。だからこそ、たとえコードそのものを完璧に書けなくても、「どのような手順や仕組みで動かすか」を論理的に考えられる力を育てること。それこそが、これからのプログラミング教育において最も大切にしたいことだと考えています。