授業実践

身の回りの生活におけるロボットの活用実践 ～micro:bitでつくるミニチュア配膳ロボット～

生徒全員が自ら手を動かしたくなるプログラミング授業を求めて

俊成学園中学校・高等学校 教諭　岡野　英樹

1．はじめに

令和４年度から高校で導入された「情報Ⅰ」の科目は，今年で４年目に入る。また，共通テストの「情報」も令和７年から始まった。やはり目玉となるのはプログラミング教育で，その教育方法や学習方法は多岐にわたる。そもそも文部科学省の情報の学習指導要領では，プログラミングの項目で，「社会や自然などにおける事象をモデル化する方法，シミュレーションを通してモデルを評価し改善する方法について理解すること。」[(1)]が求められており，プログラミングは共通テストのためだけに学ぶものではない。これに基づくプログラミング教育が求められる。

本校は男子校であり，高校１年生約260名を対象に「情報Ⅰ」の授業を行っている。全日制普通科の３学期制に基づき，中間試験と期末試験をそれぞれ２回，そして学年末試験を１回，計５回の考査を実施している。そして，プログラミングの授業は，２学期のはじめから，中間考査までの期間で行っている。まず，６コマを使いmicro:bitを用いたPythonの基礎学習を行ってから，その後，４コマ利用して企業と共同で開発したミニチュア配膳ロボットを使用したプログラミング授業を展開している。本報告では，プログラミング授業に向けて生徒はどのように準備していき，取り組むのか，また，来年度に向けてどのように振り返り教材を作り替えていくのかをまとめていく。

2．入力量を増やすためのタイピングスキル

まず，実習課題のスピードについていけるように，１学期はじめから１学期中間考査ぐらいまでは教科書の内容とタイピングを行っている。これまでタイピングを学習せずに授業を行ったこともあったが，授業の進度が遅くなる問題があった。１学期の最初にタイピングをする理由については，この後の授業で実習課題の量が増えても，生徒が対応できる速度を身につけさせるためである。

タイピングは，本校では「マイタイピング」[(2)]というWebサイトを活用して学習している。アカウント登録すれば，自分でタイピングさせたい文章や単語の問題を作ることができ，アカウント登録をしなくても，誰でも学習できるタイピングWebサイトである。以前は様々なタイピングソフトやタイピングサイトを使っていたが，キーを押す指が，自己流で正確な指ではなかったり，キーボードを見てしまったりする生徒もいるため，様々な教材研究の末にこちらにたどり着いた。

主に，タイピングは目標に準拠した評価をしている。まずは，ホームポジションキーを教え，キーボードの中段（asdfghjkl;）「中段マスター」を学習する。中段は当然母音が「あ」しか存在しないので，文章や単語はあまり登場しないが，それ以上にホームポジションキーのマスター及び正しい指使いに重点を置きたいので中段だけのゲームを設定している。中段で秒間2.0タイプ以上（スコアに直すと2000点以上）を目標にしている。2000点以上スコアを取ると，教員に知らせ，教員は手を隠す板を生徒の手の上の置き，そこで再度テストし，2000点以上とると中段は合格というルールである。次に，中段と上段（asdfghjkl;とqwertyuiop）で組み合わせたゲームを行い，同

様に2000点以上取ると，再び教員に知らせ，テストをする。これも合格すると，中段と上段と下段（asdfghjkl;とqwertyuiopとzxcvbnm）になり，これも合格すると，中段と上段と下段と最上段（ asdfghjkl; と qwertyuiop と zxcvbnm,. と 1234567890-）とレベルアップしていく課題になる。また，指導上の留意点として，「タイピングはイライラしてくるので，イライラしたら休憩してください。」と話すようにしている。休憩後に，新たな気持ちでタイピングすることで内田クレペリン検査のように休憩後パフォーマンスがあがったりすることを指導しながら体感している。また，新しいスキルの習得には，こまめな休憩が効果的であるという報告[3]がある。新しいスキル習得のパフォーマンスは練習中ではなく，休憩中にあがっているという。

このようなルールにすることで，生徒の能力に応じて個々で学習できるという点と，全員が同じ空間で行うことによって，お互いに競争心を持ち，意欲的に学習させることができる。自分以外の他の生徒が合格していくと，自分もやらざるを得ない状況になる。また，個人的には，中段と上段が秒間２タイプ打てるようになれば，タイピングはほぼマスターしたと考えている。したがって，生徒を中段と上段の指導までは注視しながら行っている。そして，夏休みの宿題にタイピング課題を出し，２学期に備えてプログラミングの準備をしてもらう。

３．プログラミング指導法の分類

プログラミングの指導及び教育に関しての議論は昔から行われている。Dehnadi[4]らはプログラムが書けるようになるかは，モデルを頭の中に作ることができるかによると指摘しており，学生のメンタルモデルの重要性を提唱している。

メンタルモデルとは，「人間が無自覚のうちに持っている，思い込みや価値観」である。ただ，これを構築するためには，経験を踏んでいくことが必要となる。その経験をプログラミング教育にあてはめるならば，「書く経験」「読む経験」「コードを分割して考える経験」「デバックする経験」「有名なアルゴリズムについて考える経験」であると考えている。筆者は，この経験を涵養するプログラミングの指導法に関して主に４つが存在し，それぞれメリットとデメリットがあると考えている。それをまとめたものが以下である（表１）。

１つ目の写経型は，「そのままプログラムコードを写す」方法である。０からコードを書かせることで，全体構造を把握できる反面，単純作業になりがちであり，学習者の思考（メンタルモデル）を働かせにくい。２つ目の穴埋め型の方法は，

表１　プログラミング指導方法の分類

	写経型	穴埋め型	摂動型	お題型
説明	そのままプログラムコードを写す	プログラムがすべて書かれてあり，主要な部分だけが穴埋めで構成されてそれを埋める	写経のプログラムに加え，微小の変化を加えるお題を出し，コードを書かせる	お題を与えて，個人またはグループで作らせる
メリット	コードを０から書かせることができ全体構造を把握できる	プログラムが書けなくても試験問題は解ける可能性がある	変数の変化や構文の意味を理解させようとすることができる	より実践的なプログラミング指導が可能になる
デメリット	単調な作業になりがちである	プログラムを０から書けない事が多い	生徒のつまずきを詳細に把握しないといけないため，微小の変化を加える問題を設定することが意外と難しい	初学者は手が止まりがちになる

「プログラムがすべて書かれてあり，主要な部分だけが穴埋めで構成されてそれを埋める」方法である。共通テストやプログラミングの資格試験と似たようなことを実習で行う。このような場合であると，プログラムが書けなくても試験問題は解ける可能性がある反面，プログラムの全体構造は深く理解しないまま進むので，プログラムを０から書けないことが多い。３つ目の摂動型は，亀井，古宮ら[5]が提唱している方法で，「写経のプログラムに加え，微小の変化を加える問題を出し，コードを書かせる」方法である。こちらは，変数の変化や構文の意味を理解させようとすることができる反面，筆者の経験上，教員側は生徒のつまずきを詳細に把握しないといけないため微小の変化を加える問題を設定することは意外に難しい。最後の４つ目のお題型は，「お題を与えて個人またはグループで解かせる」方法である。このような場合であると，より実践的なプログラミング指導が可能になる反面，初学者は手が止まりがちになる。本校のプログラミング教育はこの４つの指導法を織り交ぜながら行っている。

また，プログラミング学習は，画面に映るエディタ上に入力して，画面上で結果を表示するソフトウェア型の学習であることが多い。しかし，最初は興味があってもだんだんと興味がなくなってくることを筆者の大学時代の自分の経験や周りの意見から感じていた。そこで，よりプログラミングを楽しんで学習してもらえるようにハードウェアのmicro:bitを使い，Pythonの一種であるMicro Pythonでプログラムを記述する方法を採用した。

４．プログラミングPython基礎課題

プログラミングPython基礎課題は，合計６コマで編成される。①はmicro:bit配布と実行，②エラーと変数とコメントアウト，③LED自作と疑似乱数とif文，④if文じゃんけんプログラム，⑤while文とif文と初期化，⑥for文と配列である。関数とオブジェクト指向の説明に関しては７コマ目以降で行うミニチュア配膳ロボットで教えている。一般的なプログラミング学習の内容と同じであると思うが，エラーの見方やコードの一部を実行させないコメントアウトの使い方にも時間を割いて学習を行っている。

授業スタイルとしては，教材はプリントB４判表裏で構成され，オリジナルで作っている。プリント裏側では，テスト対策部分として，様々な文法や使い方の説明をする。これを10分程度で終わらせ，残りの40分で表側のプログラミング課題を行っていく。筆者はその40分の間で机間巡視しながら，コードの未入力があれば指摘したり，解くヒントを個人個人分けて与えたりしている。プログラミング学習はある程度，理解のある人が隣についていた方が捗りやすいと考えているからである。また，①～⑥まで終わったら，プログラムの処理体系は順次処理，反復処理，分岐処理に分けられ，文法上で教えることはこれが大部分であると説明を付け加え，この３つを組み合わせでいけばどのようなプログラムも作成可能であることを教えている。

学習者のレベル差はどうしてもつきものになる。このような場合であっても，全員が意欲的に参加できる個別最適化学習の授業設計が望まれる。これを実現するために，課題を星の数に応じて３段階にレベル別に分けて提示する構成にしている。★は全員がやるべき課題（写経型），★★は挑戦してほしい課題（摂動型），★★★は課題ができすぎて時間を持て余してしまう生徒用の課題（お題型）である。できたところまでをタブレットでプログラムコードの写真を撮って提出させ，それぞれの課題について１点から３点で評価するという方法をとっている。

また，エディタの選定も重要になる。micro:bitの場合，micro:bit Python Editor[6]というWebサイトがとても有用で助かっている。エラーがあれば，micro:bitに転送せずとも赤丸でエラー付近を指摘してもらうことができ，初学者にはわかりやすいエディタとなっている。近年では，無料のエディタであってもこのようなエラーを指摘してくれるものが登場しているので，これを上手に活用する方法もある。また，ショートカットコマンド

を使い，簡単に＃を先頭に付けることができるコメントアウトが使えることも重要である。変数の変化など初学者はあまり理解できない。そのため，その変数の変化を途中検証したければ，変数の中身を出力するコードを書いて検証しなさいと指導している。そして，検証が終わり予想通りに実行されているとわかれば，その部分をCtrl＋／でコメントアウトし，その行の実行を無効化しなさいと指導している。具体的な使い方は後述にあるコード１（p.27）に記載している。

５．ミニチュア配膳ロボット

基礎課題が終わったところで実践的なプログラミングに移る。基礎課題で利用したmicro:bitを今度はロボット[7]に装着して４コマ使ってプログラミングを行う（図１）。

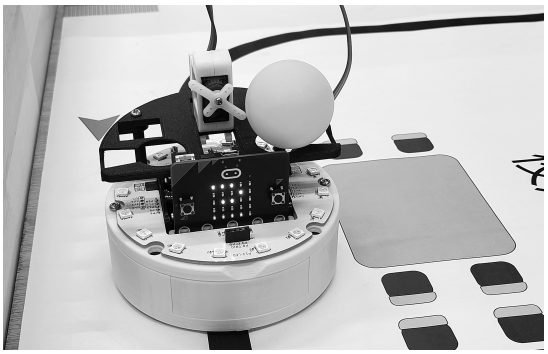

図１　ミニチュア配膳ロボット

ロボットというハードウェアを使うことで，壊れる可能性があるが，それ以上に子供たちの興味・関心が増すため，壊れた際にすぐ別のロボットを用意できるようにクラスの人数より３～４台多いロボットを発注している。

コードをすべて足し合わせると70行ほどのコードになるが，大きなコードの場合，上から順番に書いていくことはプログラマーでも行わない。そのため，ミニチュア配膳ロボットの動作行為を細かく分割し，指定された動作をクリアしていくことで，だんだんと配膳ロボットの基本的な動作行為に近づいていくような課題設定にしている。その課題設定が以下である（表２）。

STEP１からSTEP７までは写経型なのであまり難しくないが，プログラムを分解して書いていくことを体で理解してもらうためにあえて設定している。STEP８からSTEP11まではヒントや少しだけのサンプルコードを与え，自分でプログラミングを行っていく摂動型の課題となっている。そして，最後のSTEP12はヒントなしで行うお題型の課題となっている。学年全体でSTEP12をクリアできるのは例年１～２人程度となっている。ただ，このような難易度であっても，ハードウェアを使ったプログラミングは子供たちの興味・関心を引き出すには非常に有効である。自分が表現したプログラムが正しく動くのかを検証する楽しさがあるので，やんちゃな子であっても真剣に取り組む。STEP８以降が難しくてもなぜか続けて

表２　ミニチュア配膳ロボットのSTEP別課題

STEP１．	コードはコメントアウトを使い行為を分解しておく。
STEP２．	モジュールを読み込み，まっすぐ走る。
STEP３．	黒い線まで走っていき，黒い線になったらとまる。
STEP４．	STEP３のプログラムをコメントアウトし，ライントレースする関数を書いて，ライントレースする。
STEP５．	テーブルをカウントしていくプログラムを書く。
STEP６．	テーブル番号のセットをするプログラムを書く。
STEP７．	もし，配膳テーブルと一緒なら，とまり，配膳する。（完成）
STEP８．	（発展編）動き始めたら，緑に光り，キッチンについたら消える。
STEP９．	（発展編）動き始めたら，緑に光り，配膳後は赤く光り，キッチンについたら消える。
STEP10.	（発展編）動き始めたら，緑に点滅し，配膳後は赤く点滅し，キッチンについたら消える。
STEP11.	（発展編）キッチンについたら，時計回りに緑の光の玉が移動し続け，緑の光を一周させる。
STEP12.	（発展編）動き始めたら，時計回りに緑の光の玉が移動しつづけ，配膳後は反時計回りに赤い光の玉が移動しつづけ，キッチンについたら消える。

しまう。周りの友達が成功しているところを見ると競争意識がわき，自分の取り組む意欲も向上するように見受けられる。何度も何度も失敗しても書き続け，成功したら「先生！先生！先生！」と嬉しそうに報告しに来てくれる。実際に配膳ロボットを始めると，授業が始まる前からプログラミングに取り組む生徒が増えた。また，授業終了のチャイムになると，「もう終わり？つまんな」と言われ，とても授業に集中している姿が目に映った。

このようにロボットを使ったプログラミング教育は興味・関心を引き出すには非常に有効ではあるが，指導上の留意点も多い。まず，ハードウェアなので故障の心配がある。42台購入して，うまく設定しないと動かないものも２，３台ほどある。生徒人数以上に万が一に備えて少し多く購入しておかなければならない。そして，初年度に指導を行う場合，想定外のトラブルもある。しかしこの点に関しては，次年度以降慣れる。本校にはもう一人情報を教える専任教諭がいるが，２年目以降は慣れてロボットの動作行為を見てすぐプログラムのコードの修正箇所を指摘できるようになっている。

６．ふり返りと次年度への変更

一昨年度は表２のSTEP８は設定しておらず，STEP９をSTEP８の課題にしていたが，そのSTEP８で手がとまってしまう生徒が多かった。そこで，より段階的にSTEPを進められるように昨年度は，一昨年度のSTEP８をSTEP９とし，間に新たに表２のようにSTEP８を設定した（図２）。

図２　STEP別課題の挿入イメージ

そして，筆者が担当したクラスで，最終到達STEPの割合を一昨年度と昨年度で比較を行った（図３）。

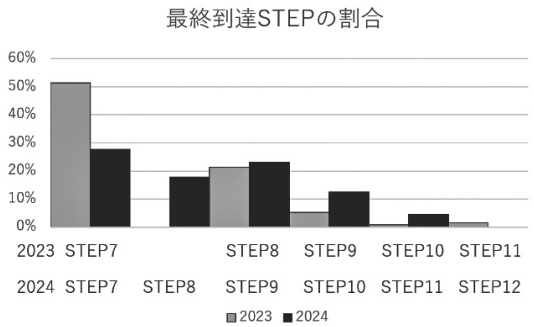

図３　最終到達STEPの割合の比較

グラフから見てわかるように，一昨年度はSTEP７で停まっていた約51％の生徒を昨年度は28％と少なくすることができ，STEP８以降に誘導できている。さらに昨年度のSTEP９以降も一昨年度と同じ内容にも関わらず，徐々に増えていることがわかる。難易度が高いプログラム課題の場合，間に中間課題を設定して段階を踏ませた課題を用意しておくと，うまくいくことが多いと考えている。

８．おわりに

筆者はプログラミング指導で生徒の手が止まってしまう場合は，２つの対策を主にしている。１つ目は，上記に挙げた「コードの分解」，そして，２つ目は，「例文コードの提示」である。

１つ目のコードの分解に関して，その課題の全体のコード量を10とした場合，３の量のレベルの課題，６の量のレベルの課題などを作っていき，段階を踏ませる課題を間に追加することが多い。つまり，少し書いたら実行，少し書いたら実行というように，コードを膨らまして書いていくような課題を設定することが重要であると考えている。

２つ目の例文コードの提示に関して，ヒントを与えたり，例文コードを与えたり，そこから考えさせるような課題に作り替えていたり，テスト前に正解のコードを見せたりするようにして理解の調整を図っている。

例えば，一昨年度，プログラミングPython基礎課題の⑥for文と配列の項目において，指定された配列の要素に対して，それが素数か合成数か

コード１　配列の要素のそれぞれの数は素数か合成数かを調べる一般的なコード

```
(1)     # Imports go at the top
(2)     from microbit import *
(3)
(4)     Num = [5, 7, 9, 11, 35]
(5)     for i in range(0, 5):
(6)         # display.scroll(Num[i])
(7)         prime = True
(8)         for j in range(2,Num[i]):
(9)             # display.scroll(j)
(10)            if Num[i] % j == 0:
(11)                prime = False
(12)                break
(13)        if prime:
(14)            display.scroll(str(Num[i]) + '=p')
(15)        else:
(16)            display.scroll(str(Num[i]) + '=c')
```

※micro:bitのdisplay.scroll()はPythonのprint()と同じような意味である。

を判定する問題を設定していた（コード１）。この場合，★には配列の要素をただ表示させる問題を写経型（１行目から６行目まで）で出し，★★★に素数か合成数かを判定させるお題型の問題を出していた（１行目から16行目まで）。しかし，やはり多くの生徒が止まってしまった。そこで昨年度から中間課題を設定し，★★の摂動型の問題を作った。for文の中にfor文を作る訓練をさせるために，「5, 2, 3, 4, 7, 2, 3, 4, 5, 6, 9, 2, 3, 4, ‥‥」というように配列の要素を表示させたあと，２からその数－１までを表示させるような途中経過の問題を設定した（１行目から９行目までだが，７行目は書かなくてもよい）。一部の生徒は手が動いたが，これでもfor文の引数の中に，変数や配列変数を指定できるという細かい文法作法を知らない生徒がいるため，今年度はさらに例文コードの提示を行う予定である。九九の１の段から５の段までのサンプルコードと，for文の引数に，具体的な数字だけではなく，変数や配列変数を書くことも可能であるサンプルコードを載せるように作り替えたのでそれらを授業で展開していこうと試みている。

このように，プログラミングの授業および，振り返りをまとめていったが，本校の授業プリント[8]をご覧いただく場合，「配膳ロボット　プログラミング教育」で検索していただければ，情報が載っているのでそちらも適宜参考にしていただければ幸いである。

９．謝辞

本取組を進めるにあたり，株式会社スイッチエデュケーションの宗村様，木原様および本校情報科教諭の萩原知明様にこの場を借りて感謝申し上げます。

10．参考文献

(1) 文部科学省，「高等学校学習指導要領（平成30年告示）解説 総則編」.
(2) 中段マスター―マイタイピング―
https://typing.twi1.me/game/29641（2024年12月20日参照）.
(3) ER Buch, “Consolidation of human skill linked to waking hippocampo-neocortical replay,” Cell Reports, Volume 35, Issue 10, June 8, 2021.
(4) Saeed Dehnadi, Richard Bornat, “The Camel Has Two Humps,” 2006.
(5) 亀井亮汰，古宮誠一，「写経型学習の欠点を補う摂動を用いた理解度確認問題生成手法―二項演算子の事例に基づく有効性評価」，日本ソフトウェア科学会第36回大会，2019年.
(6) micro:bit Python Editor
https://python.microbit.org/v/3（2024年12月20日参照）.
(7) micro:bit用ロボットベース
https://switch-education.com/products/microbit-robobase-linesensor/（2024年12月20日参照）.
(8) 佼成学園高等学校, 情報Iを見据えたプログラミング教育～学習者の独自設計を可能にさせたミニチュア配膳ロボットのプログラミング教材の開発と実践と評価～
https://www2.kosei.ac.jp/port/robot.html（2024年12月20日参照）.