「情報Ⅰ」プログラミング指導あれこれ

秋田県立秋田西高等学校 教諭
長岐 孝一

1. はじめに

2022年度から開始された学習指導要領に基づいて「情報Ⅰ」の指導が始まって3年目となり，今年度は初めて大学入学共通テストにおいて「情報Ⅰ」が実施される。2022年11月に情報Ⅰの試作問題[1)]が発表されており，多くの先生方が，その内容を参考にしながら「情報Ⅰ」の日々の教材研究を進めていることだと思う。

その試作問題において，学習指導要領の「情報Ⅰ」の4分野のうち「コンピュータとプログラミング」の配点が他の分野に比べて高いことからも，特に「コンピュータとプログラミング」の分野の指導の充実が求められている。各学校では，情報機器の設置環境等にあわせて，扱うプログラミング言語，内容，方法を工夫されて，授業をなされていると思う。

私も他の先生方のプログラミングに関する授業内容を見聞きしながら自分なりの授業を展開してきたところである。そこで，恐縮ながら私の授業実践の中からいくつかご紹介させていただきたい。

2. 本校の学習環境

本校は秋田県の中央部に位置し，1学年160人の普通科高校である。国公立大学へは約30人，私立大学へは約50人進学している。「情報Ⅰ」は高校2年生で2単位履修しており，大学入学共通テストで「情報Ⅰ」を受験する生徒が過半数であることを想定して，指導計画を立てている。

1人1台端末としてChromebookで，Google Colaboratoryを利用してPythonでプログラミングの指導を行っている。

3. 付箋紙を用いたアルゴリズムの学び

プログラムのアルゴリズムや流れを学ぶ際に，フローチャートを用いることが多いが，フローチャートの書き方を一通り学び，中身を自分で書くことができるようになるためには，ある程度の授業時間数が必要になる。できるだけ簡単に，手を動かしながら学ぶことを目指して，付箋紙を用いたアルゴリズムの学びを実践してみたので紹介したい。

いくつかの課題を与えて，付箋紙を利用してアルゴリズムを作成する。写真だとわかりにくいかもしれないが，入力文，実行文，表示文，条件分岐文，繰り返し文などにわけて，付箋紙の色や大きさで機能を分類している。

写真1は入力した整数が偶数か奇数かを表示する課題，写真2は1時間に2倍に増える細菌の5時間後の数を求める課題である。

なお，インデントを意識して分岐構造，反復構造内の実行文は少しずらして貼ることにする。

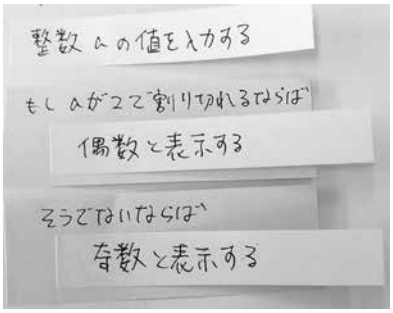

写真1　入力した整数が偶数か奇数かを表示する課題のアルゴリズム

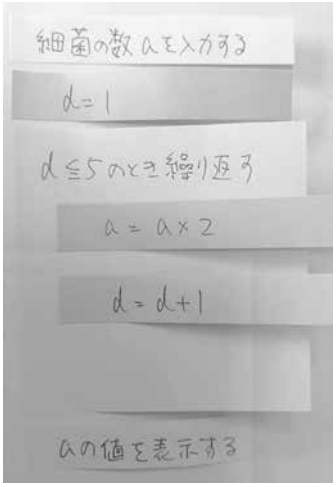

写真2　1時間に2倍に増える細菌の5時間後の数を求める課題のアルゴリズム

実際に作業しながら，短時間で分岐構造，反復構造を理解･定着させることができたと思う。ただし，授業クラスが多いと付箋紙の準備が少し大変だったのと，そのアルゴリズムが実際に正しいかどうか生徒自身が検証できないのが課題である。

4. プログラミングにおける協働的な学び

プログラミングを授業の演習として取り組む場合，その進行具合は個人差が大きく，また，個人で黙々と取り組む傾向がある。できれば，プログラミングの学習においても，生徒同士がお互いのプログラムを比較し合ったり，ミスを見つけたりする協働的な学習を授業に取り入れたい。

そこで次のような課題を授業で扱ってみた。

「1 から 50 までの素数を求めよう」という課題に対し，クラスを 3 グループに分けて，それぞれ課題 1 ～ 3 のプログラムを作成させる。

課題 1　配列変数 a に 0 ～ 50 の値を入れる。

課題 2　2 の倍数，3 の倍数，…の配列変数を 0 に置き換える。

課題 3　配列変数で 0 以外の数を表示する。

それぞれのコードの例は以下の通り。あらかじめ，課題 2 と 3 では必要な配列変数 a のデータを渡しておく。

課題 1	課題 2	課題 3

a=[]
i=0
for i in range(0,51):
  a.append(i)
	i=2
	while i<26:
	  j=2
	  while i*j<51:
	  a[i*j]=0
	  j=j+1
	  i=i+1
		i=2
		while i<51:
		  if a[i]!=0:
		  print(a[i])
		  i=i+1

それぞれのコードがきちんと実行できることを確認後，課題 1 ～ 3 に取り組んだ生徒がそれぞれ一人ずつとなるように 3 人のグループをつくり，1 つのプログラムに統合させる。

プログラミングにおけるジグソー学習のようなものである。もちろん，もっと効率的なコードを書くこともできるし，素数を直接求めるコマンドがあるので模範的なコードとはならないが，あえてプログラムを 3 つに分けることにより，協働で 1 つのプログラムを完成させるようにした。

簡単なコードであるが，分岐構造，反復構造，順次構造が必要になるので，プログラミングの基本としての定着度は高まった。

5. 大学入学共通テストと Python

大学入学共通テストでは，疑似言語を使用してプログラムを記述することになっている（以下では試作問題[1]で用いられたものを疑似言語とよぶ）。

Pyhton と疑似言語の記述のしかたに若干異なる部分があるので，ここで追記したいと思う。

Python
a=[0,1,2,3] for i=0 in range(0,3,1):   print(a[i])

疑似言語
a=[0,1,2,3] i を 0 から 3 まで 1 ずつ増やしながら繰り返す : └ 表示する (a[i])

Python のほうを実行すると 0 ～ 2 が，疑似言語のほうを実行すると 0 ～ 3 が表示される。これは，Python の「range」が 0 以上，3 未満を表すからである。こうした細かな違いを理解する必要がある。

本校のように授業で Python を学んだ高校生は，大学入学共通テストの準備として，疑似言語でのプログラミングにもある程度慣れておく必要がある。大学入学共通テストに向けたプログラムの指導の注意点である。

6. おわりに

生成 AI を用いれば，課題解決のためのプログラムのコードを簡単に得られるようになった。私自身，生成 AI で得られたコードを参考にすることも多く，コードを新しく書くことよりも，コードを読んで修正する機会が多くなった。文部科学省の「初等中等教育段階における生成 AI の利用に関する暫定的なガイドライン[2]」を参考にしつつ，プログラミングの指導において生成 AI をどのように扱うか，今後大きな課題である。

参考文献

1）大学入試センター，「令和 7 年度試験の問題作成の方向性，試作問題等」，https://www.dnc.ac.jp/kyotsu/shiken_jouhou/r7/r7_kentoujoukyou/r7mondai.html（アクセス日：2024 年 9 月 24 日）

2）文部科学省，「初等中等教育段階における生成 AI の利用に関する暫定的なガイドライン」，https://www.mext.go.jp/a_menu/other/mext_02412.html（アクセス日：2024 年 9 月 24 日）