共通テスト試作問題「情報Ⅰ」を題材としたプログラミング教育

法政大学 国際文化学部 教授
重定 如彦

1. はじめに

2022 年 11 月に，2025 年度試験から実施される共通テスト「情報Ⅰ」の試作問題が公表された[1)]。そこで，本稿では試作問題の中でプログラミングをテーマとする第 3 問を題材としたプログラミング教育についての私見を述べたいと思う。

2. 第 3 問の概要と勉強法

第 3 問は，「与えられた金額を硬貨のみで支払う際に，支払いと釣銭で使用する硬貨の枚数の最小値を求めるプログラムを作成する」という問題である。問題文には，答えを求めるための計算の手順（アルゴリズム）が説明されており，その説明に従ってプログラムを完成させるという流れになっている。

第 3 問は，プログラミングの基礎であり重要な要素である，変数，式と代入，配列，条件分岐，繰り返し，関数呼び出しなどが盛り込まれた良問であるが，マークシート方式の問題であるためどうしてもプログラミングではなく，日本語の問題として解くという，受験上のテクニックが使えてしまう設問が含まれている。例えば，問 3 の図 2 にある「 サ を シ から 99 まで 1 ずつ増やしながら繰り返す」の シ の解答群は，0，1，99，100 となっているが，「99 まで 1 ずつ増やしながら」という文から，日本語の意味的に 99 と 100 が入らないことは容易に推測できてしまう。

しかし，このような受験上のテクニックを用いて問題を解いても，プログラミングの力を身に付けることができないのは言うまでもないだろう。また，選択肢から選ぶという方式は，プログラミングに必要な論理的な思考力を測ることはできても，実際に一からプログラムを記述する力を測ることは困難であろう。プログラミング教育の目的の 1 つが受験の合格ではなく，様々な問題に対して自分でプログラムを記述して解決する力の習得であることを考えれば，第 3 問を参考にして勉強を行う際に，穴埋め選択式の問題をそのまま解くのではなく，この問題からプログラムのコードの部分をすべて削除し，説明されているアルゴリズムからプログラムを自力ですべて記述することが有効ではないだろうか。

3. 多様な記述方法の学習

第 3 問のような，与えられたアルゴリズムをプログラムの形に正しく翻訳する力を身に付けるためには，様々なアルゴリズムに対するプログラミングを数多くこなすことが非常に重要である。しかし，授業時間の制約上，どうしても取り上げることができる題材が限られてしまう。そこで，一つの題材に対して，異なるアルゴリズムやプログラムの記述方法を紹介することを提案したい。具体的には，第 3 問を題材に授業を行う際，自分の思うようにプログラムを記述させてみるのはどうだろうか。おそらく，配列を使わずに条件分岐を使って i の値に応じた硬貨の額を計算するプログラムを記述する生徒や，硬貨の種類を値段の大きいものから順に配列で表現し，配列の先頭の要素から順番に繰り返しを使って処理を行う生徒が出てくるだろう。多様なプログラムの記述を得ることができれば，それらを比較することで，同じアルゴリズムであっても様々な方法で記述できることや，それぞれの利点や欠点を実感できる。その際，文法的または論理的に間違ったプログラムが混じる可能性が高いが，それらの間違ったプログラムは，初心者が起こしやすい文法エラーや論理エラーの具体例として利用することもできる。

なお，その際に論理エラーのあるプログラムを見逃さないように気を付ける必要がある。論理式内の比較演算子の > と >= を間違ってしまうような簡単なものであれば比較的発見しやすいが，初心者にありがちな例として，経験者にとって思いもかけない

ような間違った論理でプログラムを記述する場合がある。また，例えば変数 x が 1 以上 3 以下を判定する論理式を「1 <= x <= 3」と記述[2]するなど，文法の理解不足によって論理エラーが発生する場合もある。

初心者が最初に間違ったことを覚えてしまうと癖となってしまう可能性が高いので，論理エラーは可能な限り発見し指摘しておいたほうが良い。残念ながら，論理エラーを必ず発見できるような方法は存在しないが，第 3 問の最後でも述べられているような，「様々な値(条件)に対してプログラムを実行し，すべて正しく計算できているかを確認すること」は，論理エラーを発見するための有効な手法の一つであり，この作業の重要性は説明しておくべきである。第 3 問の問題を複数の生徒に個別にプログラムを作成させた場合は，全員に様々な金額で枚数を計算してもらい，その計算結果を突き合わせて確認するという手法をとると良いだろう(全員が間違っている可能性もあるので，先生の結果と突き合わせても良い)。

4. 第 3 問を利用したアルゴリズムの確認

プログラムの論理エラーの原因の一つに，アルゴリズムそのものが間違っているというものがある。第 3 問の問 2 で説明されている関数「枚数」のアルゴリズムには暗黙の条件が隠されているが，そのことに気がつく生徒はかなり少ないのではないだろうか？このアルゴリズムが正しく動作する十分条件[3]は，硬貨を金額の小さい順に並べたときに，一つ大きい硬貨の金額がその前の硬貨の金額の整数倍になっていることである。数学的な証明は省略するが，そのことは直観からも明らかであり，日本の硬貨はその条件を満たしているので，このアルゴリズムは第 3 問の場合は正しく動作する。しかし，この十分条件を満たさない場合は正しく動作しないことがある。例えば硬貨の種類が 1，5，10，20，25 円の 5 種類の場合に，このアルゴリズムで 40 円の最小枚数を計算すると，25 円が 1 枚，10 円が 1 枚，5 円が 1 枚の計 3 枚となるが，正しい最小枚数は 20 円が 2 枚の計 2 枚である。思いついたアルゴリズムが本当に正しいかを検証する必要があるということの実例として，この第 3 問の問題設定を上記のような硬貨の種類に変えた場合で解かせることで生徒に実感させてみるのはどうだろうか。また，余裕があれば，どのような場合でも最小枚数を数えることができるアルゴリズムを考えさせてみるのも良いだろう。

5. バグに対する心構え

ここまでいくつかエラーの話を取り上げたが，プログラミングの初心者が躓く理由の中で，個人的に最も大きなものはデバッグの難しさにあると考えている。また，筆者が担当するプログラミングの授業での雰囲気から，バグの発生を恥ずかしいことや悪いことのように思っている学生が多いと感じている。

プログラミングをある程度経験したことがある人なら誰でもわかる事だが，そもそも一度でバグのないプログラムを完成させることは現実的ではない。そのことを生徒に伝える一つの方法として，先生自らバグを発生させてみせるというのはどうだろうか。実際に授業でプログラムを記述して説明する際にバグを発生させることで，先生でも間違えることがあるのだということを生徒に示すことができる。なお，筆者は，授業で意図せずにバグを発生させてしまい，数分間悩んだ挙句，学生にバグの場所を指摘されたことが何度かあるが，おそらくその学生にとっての大きな成功体験になったのではないだろうか。デバッグの技術の向上は，様々なバグを実際に体験し，それを適切に修正することが非常に重要である。そのため，初心者のうちは，バグが発生した場合は逆に上達の機会に恵まれてラッキーだと思ったほうが良いと学生に伝えるように心がけている。

最後に手前味噌になってしまうが，筆者が大学でのプログラミングの授業で得た経験を元に執筆した教科書を参考文献で紹介する[4]。本稿で紹介したように，取り扱う各題材に対して様々なアルゴリズム，記述方法，発生する可能性の高い論理エラーとデバッグなどを特に重要視した内容になっているので，興味がある方は参考にしていただければ幸いである。

参考文献・注釈

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 試作問題『情報Ⅰ』」 https://www.dnc.ac.jp/kyotsu/shiken_jouhou/r7ikou/r7mondai.html（アクセス日：2023 年 2 月 24 日）

2) この記述が正しいプログラミング言語も存在する。

3) 必要条件ではない点に注意。

4) 重定如彦，「学生のための JavaScript」，東京電機大学出版局，https://www.tdupress.jp/book/b600190.html（アクセス日：2023 年 2 月 24 日）