ドント方式による議席配分

情報処理学会・学会誌「情報処理」
2023年8月1日 13:14

岩崎英哉（明治大学）

今回取り扱うのは，2021年3月に公開された大学入学共通テスト「情報」のサンプル問題の第2問で，プログラミングに関する問題です．この問題は，比例代表選挙において各政党の得票数から具体的な当選者数（議席数）を決定する方法である「ドント方式」を題材としています．現在の日本の国政選挙における比例代表選挙は，衆議院では候補者に順位のある（ただし同順位の候補者を許す）拘束名簿式，参議院では候補者に順位のない非拘束名簿式が採用されており，いずれも少し複雑なシステムとなっています．本問では，候補者名簿の中身には触れずに当選者数だけを求めるという，最も単純な場合を扱っています．

ドント方式は，過去には2011年の大学入試センター試験における「情報関係基礎」でも題材とされたことがあります．2006年以降，「情報関係基礎」における情報の処理方法の論理的思考力・問題解決能力に関する問題は，プログラミングによる第3問と表計算による第4問の一方の選択となっており，2021年に大学入試センター試験が大学入学共通テストに衣替えしてからも，この問題構成は維持されています．2011年のドント方式の問題は第4問でしたので，表計算ソフトウェアを用いるものでした．

今回紹介するサンプル問題も情報関係基礎の表計算ソフトウェアによる問題も，話の流れはほぼ同じで，次のような展開になっています．

1. 政党の得票数に基づいて議席の定数を比例配分して各政党の当選者数を決定しようとすると，比例配分した値に小数点以下の端数が出て，当選者数の合計が定数と一致せずうまくいかないことを確認する．

2. 比例配分によらない方法として，ドント方式の考え方を説明する．
3. 各政党の名簿には十分な数の候補者がいるという前提のもとで，ドント方式により当選者数を求めるプログラム，あるいは，表計算操作を完成させる．
4. 候補者数が当選者数に足りない場合の考慮等が必要であることを指摘する．

「情報関係基礎」では，表計算ソフトウェアが提供する基本的な関数群の活用に主眼が置かれていましたが，今回紹介する問題ではこれをプログラムによって実現します．本問で用いるプログラムは，Pythonのような特定のプログラミング言語を用いるのではなく，大学入学共通テスト「情報」のための言語である「共通テスト用プログラム表記」[1]（以下略して単に「プログラム表記」と呼びます）を利用します．これは，情報関係基礎の第3問で用いられる「共通テスト手順記述標準言語（旧センター試験用手順記述標準言語）」[2]（「DNCL」と呼ばれます）とよく似た言語となっていますが，微妙な違いがいくつかあります．この点については，最後に少し触れたいと思います．なお，試作問題「情報」の概要（https://www.dnc.ac.jp/kyotsu/shiken_jouhou/r7ikou/r7mondai.html）に，プログラム表記が「例示」という形で示されてます．

前置きが長くなってしまいました．それでは問題を見ていきましょう．まず問題の導入部は次のようになっています．

第２問　次の文章を読み，後の問い(問１～３)に答えよ。

Mさんは，18歳になって選挙権が得られたのを機に，比例代表選挙の当選者を決定する仕組みに興味を持った。そこで各政党に配分する議席数（当選者数）を決める方法を，友人のKさんとプログラムを用いて検討してみることにした。

ここにあるように，本問の問題文はMさん，Kさん，後から登場する先生による対話形式になっています．最近は，対話形式を用いた問題文が，さまざまな科目で用いられているようです．

比例配分による方法

問1は，素朴な方法として，比例配分によって当選者数を決めようとしています．

問１　次の文章の空欄 ア ～ ウ に入れる最も適当なものを，後の解答群のうちから一つずつ選べ。同じものを繰り返し選んでもよい。

ここでは「同じものを繰り返し選んでもよい」と書かれています．回答する際の大前提ですので，見落さないようにしましょう．

次に，各党の得票数，議席数などの情報が提示されます．いずれも重要な情報ですね．

Mさん：表1に，最近行われた選挙結果のうち，ある地域のブロックについて，各政党の得票数を書いてみたよ。

表1　各政党の得票数

	A党	B党	C党	D党
得票数	1200	660	1440	180

Kさん：今回の議席数は6人だったね。得票の総数を議席数で割ると580人なので，これを基準得票数と呼ぶのがいいかな。平均して1議席が何票分の重みがあるかを表す数ということで。そうすると，各政党の得票数が何議席分に相当するかは，各政党の得票数をこの基準得票数で割れば求められるね。

ここでは「基準得票数」という用語が目立たずに定義されており，さらに，これを用いた比例配分による各政党の議席数の計算方法が，言葉で説明されています．この部分もしっかりと抑えておきたいところです．

さて，いよいよプログラムを書く部分に入っていきます．まずプログラム中で用いる配列が説明されています．

Mさん：その考え方に沿って政党ごとの当選者数を計算するプログラムを書いてみよう。まず，プログラムの中で扱うデータを図1と図2にまとめてみたよ。配列 Tomei には各政党の党名を，配列 Tokuhyo には各政党の得票数を格納することにしよう。政党の数は4つなので，各配列の添字は0から3だね。

i	0	1	2	3
Tomei	A党	B党	C党	D党

図1　各政党名が格納されている配列

i	0	1	2	3
Tokuhyo	1200	660	1440	180

図2　得票数が格納されている配列

Mさん：では，これらのデータを使って，各政党の当選者数を求める図3のプログラムを書いてみよう。実行したら図4の結果が表示されたよ。

党名を保持する配列はTomei，得票数を保持する配列はTokuhyoとなっています．このように，配列名や変数名はその内容を表す名前が付けられることが多く，答を考える際のヒントにもなっています．またここでは，配列の添字が0から始まるという情報も与えています．これは，プログラム表記の仕様をよく知らない人でも問題を解けるようにという親切心でしょうか．

ちなみに図1の中の「A党」などは政党名を表す文字列なので，本当は「"A党"」のように「"」で囲って書くのが，より正しいのでしょう．さらに，図1，図2の配列名の上には「i」と書かれています．これは配列の添字を表す変数と推測しますが，問題文や図のどこにも現れておらず謎ですね．もしかしたら，iの下に書かれている配列名のところは「Tomei[i]」のように添字付きで書くべきだったのかもしれません（2014年の「情報関係

基礎」第3問の配列の図は，そのような書き方になっていました）．本問はサンプル問題なので仕方ありませんが，本番の問題ではこのようなところもしっかり校正していただきたいところです．

さて，いよいよ穴埋めするプログラムの登場です．穴に対する選択肢も合わせて示しておきましょう．

```
(01) Tomei = ["A党","B党","C党","D党"]
(02) Tokuhyo = [1200,660,1440,180]
(03) sousuu = 0
(04) giseki = 6
(05) mを0から [ア] まで1ずつ増やしながら繰り返す:
(06)     sousuu = sousuu + Tokuhyo[m]
(07) kizyunsuu = sousuu / giseki
(08) 表示する("基準得票数：",kizyunsuu )
(09) 表示する("比例配分")
(10) mを0から [ア] まで1ずつ増やしながら繰り返す:
(11)     表示する(Tomei[m],"：",[イ] / [ウ] )
```

図3　得票に比例した各政党の当選者数を求めるプログラム

[ア]～[ウ]の解答群

⓪ 0　① 1　② 2　③ 3　④ 4　⑤ 5　⑥ 6　⑦ Tomei[m]
⑧ Tokuhyo[m]　⑨ sousuu　ⓐ giseki　ⓑ kizyunsuu

（01）と（02）は，ふたつの配列に対する初期設定です．配列値の設定は各括弧 [と] の中，に値をカンマで区切って並べて書くことができることになっています．配列の添字は0から始まりますので，この2行によって図1と図2に示すように配列が設定されます．このような，配列値を並べることによる配列の初期設定については，本記事の最後で触れたいと思います．

（05）～（06）では，mを変化させながら変数sousuuを次々と更新しています．これにより，得票数の合計をsousuuに求めていることはすぐに分かります．mを配列Tokuhyoの添字として使っており，【ア】はその上限値ですので，答は「3」となります．この繰り返しの後，（07）で得票数の合計sousuuを議席数gisekiで割って基準得票数を求め，これを変数kizyunsuuに設定しています．

（10）～（11）では，各政党の得票数が何議席分に相当するかを求めています．計算のしかたは，表1のすぐ下のKさんの発言「各政党の得票数をこの基準得票数で割れば求められる」とありますので，これをもとに考えればよいですね．【イ】は各政党の得票数ですので「Tokuhyo[m]」，【ウ】は基準得票数ですので「kizyunsuu」となります．

問題文は次のように続いています．

Kさん：得票数に比例して配分すると小数点のある人数になってしまうね。小数点以下の数はどう考えようか。例えば，A党は2.068966だから２人が当選するのかな。

Mさん：なるほど。切り捨てで計算すると，A党は２人，B党は１人，C党は２人，D党は０人になるね。あれ？ 当選者数の合計は５人で，６人に足りないよ。

Kさん：切り捨ての代わりに四捨五入したらどうだろう。

Mさん：そうだね。ただ，この場合はどの政党も小数点以下が0.５未満だから，切り捨てた場合と変わらないな。だからといって小数点以下を切り上げると，当選者数が合計で９人になるから３人も多くなってしまう。

Kさん：このままでは上手くいかないなぁ。先生に聞いてみよう。

基準得票数：580
比例配分
A党：2.068966
B党：1.137931
C党：2.482759
D党：0.310345

図４　各政党の当選者数の表示

上のプログラムにより各政党の得票数が何議席分に相当するかが分かりましたが，端数が出てきてしまいました．比例配分しているのですから端数が出るのは当然ですが，端数を切り捨てても四捨五入しても切り上げてもうまくいかないことをKさんとMさんは確認しました．

ドント方式の考え方

さて，いよいよ先生の登場です．導入部は次のようになっています．

問２　次の文章の空欄［エ］～［ス］に入れる最も適当なものを，後の解答群のうちから一つずつ選べ。同じものを繰り返し選んでもよい。

Mさん：先生，比例代表選挙では各政党の当選者数はどうやって決まるのですか？ 当選者数が整数なので，割合だけだと上手くいかなかったのです。

「情報関係基礎」の過去の問題を見ると，本格的なプログラミングの設問に入る前に，問題の前提や解法等を理解しているかを確認する問いが出される傾向にあります．おそらく教科「情報」の問題でも，同じ傾向が続くと予想されます．問題の前提，解法の説明部分の理解が曖昧のままでいると，先に進んでも分からなかったり間違えたりする可能性が高いので，多少の時間をかけてもしっかりと理解しておくことが重要でしょう．

本問もこの例にならって，まず先生がドント方式はどのようなものであるかを説明しています．

先　生：様々な方法があるけど，日本では各政党の得票数を１，２，３，…と整数で割った商の大きい順に定められた議席を配分していく方法を採用しているよ。この例だと表２のように，❶から❻の順に議席が各政党に割り当てられるんだ。C党が❶の議席を取っているけど，このとき，何の数値を比較したか分かるかな。

表２　各政党の得票数と整数で割った商

	A党	B党	C党	D党
得票数	1200	660	1440	180
1で割った商	❷1200	❹ 660	❶1440	180
2で割った商	❺ 600	330	❸ 720	90
3で割った商	400	220	❻ 480	60
4で割った商	300	165	360	45

Mさん：１で割った商です。A党から順に 1200，660，1440，180 ですね。
先　生：そうだね。ではA党が❷の議席を取るとき，何の数値を比較したのだろうか。
Mさん：C党は１議席目を取ったので，1440 を２で割った商である 720 を比較します。A党から順に 1200，660，720，180 ですね。この中で数値が大きいA党が議席を取ります。なるほど，妥当な方法ですね。
Kさん：この考え方で手順を考えてみようよ。

ここの説明にあるとおり，ドント方式では，各政党の数値を比較して，最大値を与える政党から次の当選者を選びます．この数値は，得票数を1で割った商，2で割った商，3で割った商，……とし，当選者として選ばれた場合には，比較する数値を次に進めます．問題文では2議席目までしか説明が載っていませんが，最後の6議席目が決定するまでの過程を下に示しましょう．毎回の比較での最大値を太字で示しました．太字に対応する政党から当選者を出すこととなります．

A:1200 B:660 C:**1440** D:180 ⟶ A:**1200** B:660 C:720 D:180 ⟶
A:600 B:660 C:**720** D:180 ⟶ A:600 B:**660** C:480 D:180 ⟶
A:**600** B:330 C:480 D:180 ⟶ A:400 B:330 C:**480** D:180

次に，この方式をプログラムとして実現するために用いる配列を2つ用意します．

Kさん：この考え方で手順を考えてみようよ。
先　生：まずは候補者が十分足りるという条件で手順を考えてみるのがいいですよ。
Kさん：各政党に割り当てる議席を決めるために，比較する数値を格納する配列 Hikaku がいるね。
Mさん：各政党に配分する議席数（当選者数）を格納する配列 Tosen も必要だね。最初は議席の配分が行われていないから，初期値は全部 0 にしておくね。

i	0	1	2	3
Hikaku				

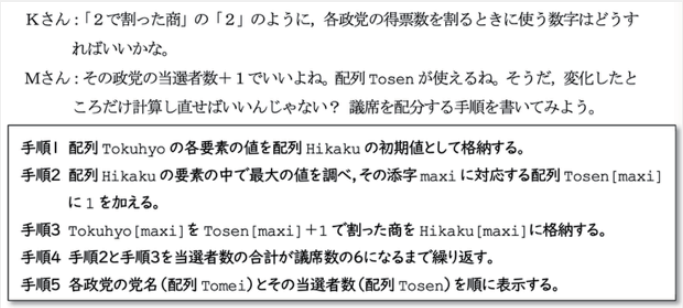
図５　整数で割った値を格納する配列

i	0	1	2	3
Tosen	0	0	0	0

図６　当選者数を格納する配列

各党に関して，毎回比較する数値を保持する配列Hikakuと，決まった当選者数を保持する配列Tosenを用意することとしました．ちなみに，図5のキャプションは間違っているわけではありませんが，直上の会話文に合わせて「比較する数値を格納する配列」とする方がよいでしょう．

さて，配列Hikakuには比較する数値が入りますが，それが最大値の場合は，その数値をひとつ先に進め，次の整数で割った商を設定しなければなりません．ここで「次の整数」はどうすれば知ることができるのでしょうか．



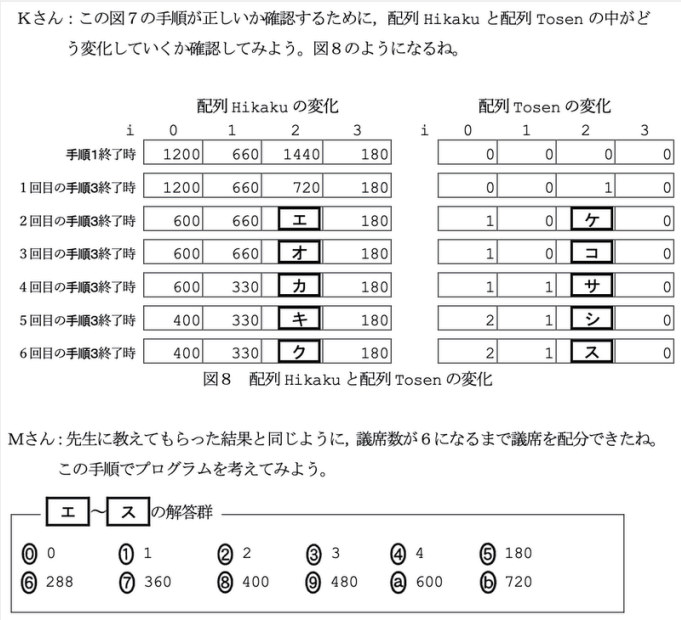
Kさん：「2で割った商」の「2」のように，各政党の得票数を割るときに使う数字はどうすればいいかな。

Mさん：その政党の当選者数＋1でいいよね。配列 Tosen が使えるね。そうだ，変化したところだけ計算し直せばいいんじゃない？ 議席を配分する手順を書いてみよう。

手順1 配列 Tokuhyo の各要素の値を配列 Hikaku の初期値として格納する。
手順2 配列 Hikaku の要素の中で最大の値を調べ，その添字 maxi に対応する配列 Tosen[maxi] に 1 を加える。
手順3 Tokuhyo[maxi] を Tosen[maxi]＋1 で割った商を Hikaku[maxi] に格納する。
手順4 手順2と手順3を当選者数の合計が議席数の6になるまで繰り返す。
手順5 各政党の党名（配列 Tomei）とその当選者数（配列 Tosen）を順に表示する。

図7　手順を書き出した文章

ここには，「次の整数」は「当選者＋1」にすればよいという重要な情報が書かれています．さらに，各党の当選者数を求める手順が文章で与えられました．この手順の動きを，以下で順を追って確認しています．



Kさん：この図7の手順が正しいか確認するために，配列 Hikaku と配列 Tosen の中がどう変化していくか確認してみよう。図8のようになるね。

	配列 Hikaku の変化				配列 Tosen の変化			
i	0	1	2	3	0	1	2	3
手順1終了時	1200	660	1440	180	0	0	0	0
1回目の手順3終了時	1200	660	720	180	0	0	1	0
2回目の手順3終了時	600	660	エ	180	1	0	ケ	0
3回目の手順3終了時	600	660	オ	180	1	0	コ	0
4回目の手順3終了時	600	330	カ	180	1	1	サ	0
5回目の手順3終了時	400	330	キ	180	2	1	シ	0
6回目の手順3終了時	400	330	ク	180	2	1	ス	0

図8　配列 Hikaku と配列 Tosen の変化

Mさん：先生に教えてもらった結果と同じように，議席数が6になるまで議席を配分できたね。この手順でプログラムを考えてみよう。

エ ～ ス の解答群

⓪ 0　① 1　② 2　③ 3　④ 4　⑤ 180
⑥ 288　⑦ 360　⑧ 400　⑨ 480　ⓐ 600　ⓑ 720

図8の配列Hikakuの変化は，先に説明した6議席目までが決定する過程そのものです．その過程を再掲します．

A:1200 B:660 C:**1440** D:180 ⟶ A:**1200** B:660 C:720 D:180 ⟶
A:600 B:660 C:**720** D:180 ⟶ A:600 B:**660** C:480 D:180 ⟶

A:**600** B:330 C:480 D:180 ⟶ A:400 B:330 C:**480** D:180

これを空欄を対応させると，（最後の【ク】に相当するところは上の過程には書かれていませんが，）【エ】は720，【オ】は480，【カ】も480，【キ】も480，【ク】は360であることは分かりますね．【オ】の480は元々の得票数1440を3で割った商ですが，これを直前の値720を2で割ったものにしないように注意してください．配列Tosenの方は，Hikakuの値が変化したときに限って1ずつ増えることに注意すれば，【ケ】は1，【コ】は2，【サ】も2，【シ】も2，【ス】は3となります．問2の問題文の「同じものを繰り返し選んでもよい」というただし書きが重要であることが分かります．

ドント方式のプログラム

さて，いよいよ上の手順をプログラムとして実現します．

問3　次の文章の空欄［セ］～［テ］に入れる最も適当なものを，後の解答群のうちから一つずつ選べ。

Mさん：図9のプログラムを作ってみたよ。商を整数で求めるところは小数点以下を切り捨てる「**切り捨て**」という関数を使ったよ。

Kさん：実行したら図10のように正しく政党名と当選者数が得られたね。

```
(01) Tomei = ["A党","B党","C党","D党"]
(02) Tokuhyo = [1200,660,1440,180]
(03) Tosen = [0,0,0,0]
(04) tosenkei = 0
(05) giseki = 6
(06) m を 0 から［ア］まで 1 ずつ増やしながら繰り返す:
(07)     Hikaku[m] = Tokuhyo[m]
(08) ［セ］< giseki の間繰り返す:
(09)     max = 0
(10)     i を 0 から［ア］まで 1 ずつ増やしながら繰り返す:
(11)         もし max < Hikaku[i]ならば:
(12)             ［ソ］
(13)             maxi = i
(14)     Tosen[maxi] = Tosen[maxi] + 1
(15)     tosenkei = tosenkei + 1
(16)     Hikaku[maxi] = 切り捨て(［タ］/［チ］)
(17) k を 0 から［ア］まで 1 ずつ増やしながら繰り返す:
(18)     表示する(Tomei[k],"：",Tosen[k],"名")
```

図9　各政党の当選者数を求めるプログラム

［セ］，［タ］・［チ］の解答群

⓪ max	① maxi	② tosenkei
③ Tokuhyo[maxi]	④ Tokuhyo[maxi] + 1	⑤ Tokuhyo[max]
⑥ Tosen[maxi]	⑦ Tosen[maxi + 1]	⑧ (Tosen[maxi] + 1)

［ソ］の解答群

⓪ max = max + 1	① max = Tokuhyo[i]	② max = Hikaku[i]
③ Hikaku[i] = max	④ Tokuhyo[i] = max	⑤ Tokuhyo[i] = Hikaku[i]

図7の手順を図9のプログラムと対応させると，次のようになります．

手順1 ⟺ (06)～(07)
手順2 ⟺ (09)～(14)
手順3 ⟺ (16)
手順4 ⟺ (08)～(16)
手順5 ⟺ (17)～(18)

tosenkeiという変数が登場しましたが，これはその名が表す通り，「今までに決定した当選者の合計数」を保持します．このことは，当選者が決まり，(14)～(15)行目で当選者の政党の当選者数を1増やすと同時にtosenkeiの値も1増やしていることからも見てとることができます．【セ】は手順4にある「当選者の合計が議席数の6になるまで繰り返す」に該当する部分ですので，「tosenkei」であることが分かります．

次に【ソ】を考えます．これは各政党の現在の数値の中の最大値を求めるところです．変数maxには最大値の暫定的な値を保持し，比較対象の数値Hikaku[i]がそれより大きければ暫定値maxを更新する，となっています．また，最大値を与える政党に対応する添字をmaxiに設定します．よって【ソ】は「max = Hikaku[i]」となります．このように，繰り返しを用いて配列の最大値を求める方法は，一種の常套句ともいえるでしょう．

最後に(16)の【タ】【チ】ですが，手順3に対応することから，正解は【タ】が「Tokuhyo[maxi]」，【チ】が「(Tosen[maxi] + 1)」であることが分かりますね．この選択肢（8番）は両側に括弧がついているのが奇妙に見えるかもしれませんが，除算「/」は加算「+」よりも結合の優先度が高いので，括弧をつけなければなりません．もしも選択肢の中に括弧のない「Tosen[maxi] + 1」があれば，間違ってこちらを選ぶ受験生も多数現れるでしょうが，そのような意地悪な選択肢はありませんでした．【タ】については，直前の比較対象の数値をもとにして新しい比較対象の数値を計算するものと勘違いして，「Hikaku[maxi]」が正解だろうと思う受験生も現れそうですが，この選択肢はないので，勘違いしていても軌道修正がはかれそうです．ちなみに，商の整数を求めるのにここでは「切り捨て」という関数を用いていますが，整数の除算で商の整数を求める演算子に「÷」というのがありますので，(16)行目の右辺は「【タ】÷【チ】」としてもよかったのでしょう．

候補者数が足りない場合

今までは，はじめの方の先生の発言「候補者が十分足りるという条件」のもとで考えていました．しかし現実には，ドント方式で決定される当選者数に満たない人数しか候補者名簿に登録していない政党が存在することがあるかもしれません．実際日本の国政選挙においても，このような候補者不足が起こったことがあります．本問は最後に，候補者が足りない場合にも対応できるように，プログラムを修正します．

先　生：できたようだね。各政党の当選者数は求められたけど，政党によっては候補者が足りない場合もあるから，その場合にも対応してみよう。図11のように各政党の候補者数を格納する配列 Koho を追加してみたらどうだろう。例えば，C党の候補者が足りなくなるように設定してみよう。

A党：2名
B党：1名
C党：3名
D党：0名

図10　各政党の当選者数の表示

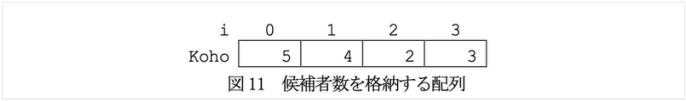


図 11　候補者数を格納する配列

ここで新たな配列Kohoが導入されました．ここには各政党の候補者の数が保持されています．候補者が足りなくなったかどうかは，この配列を利用すれば判定することができます．

Mさん：候補者が足りなくなったらどういう処理をすれば良いのですか？
先　生：比較した得票で次に大きい得票数の政党が繰り上がって議席を取るんだよ。
Mさん：なるほど。では，図9の(11)行目の条件文を次のように修正すればいいですね。当選していない候補者はどこかの政党には必ずいるという前提だけど。

(11)　もし max < Hikaku[i]　ツ　テ　ならば:

ツ の解答群
⓪ and　① or　② not

テ の解答群
⓪ Koho[i] >= Tosen[i] + 1　① Koho[i] < Tosen[i] + 1
② Koho[i] >= Tosen[i]　③ Koho[i] < Tosen[i]

比較する数値を保持する配列Hikakuにおいて最大値となっても，その最大値の政党に候補者が残っていなければ，その政党からは当選者の出しようがありません．したがって，最大値となる政党を求める(10)～(13)において，候補者が残っているという条件を付加する必要があります．そのために(11)を上のように変更します．「Hikaku[i]が最大値の暫定的な値maxより大きい」ことと「候補者が残っている」ことが両方とも成立しなければなりませんので，【ツ】は「かつ」を表す「and」になります．【テ】では「候補者が残っている」ことを表現します．今までの当選者数はTosen[i]に，候補者数はKoho[i]にあります．新たな当選者が出ると当選者数はTosen[i] + 1になりますので，候補者数はそれ以上でなければなりません．したがって【テ】は「Koho[i] >= Tosen[i] + 1」となります．この空欄には，「+ 1」を忘れて「Koho[i] >= Tosen[i]」を選んでしまう受験生が少なからず出そうです．

問題は以上で終わりです．最後に，このプログラムにはまだ問題点があることを確認しています．

Kさん：先生，候補者が不足するほかに，考えるべきことはありますか？
先　生:例えば,配列Hikakuの値が同じになった政党の数が残りの議席の数より多い場合，このプログラムでは添字の小さい政党に議席が割り当てられてしまうので不公平だね。実際には，この場合はくじ引きで議席を割り当てるようだよ。

共通テスト用プログラム表記について

そもそも「情報関係基礎」は，なぜ現実のプログラミング言語ではなくDNCLを用いてきたのでしょうか．その理由は「特定のプログラミング言語を出題に用いることによる不公平を避けるため」[3] であり，この点について「情報関係基礎」の1999年および2000年の問題作成部会の見解（情報関係基礎アーカイブ https://sites.google.com/a.ipsj.or.jp/ipsjjn/resources/JHK から閲覧可能）は「プログラミングに相当する処理手順が自然言語によって記述された出題方法は（中略） 適切との評価を得ている」とまとめています．ここで重要なことは，実用的な言語によってプログラミングを学びきちんとした知識を獲得した受験生は，予備的な知識の必要なくプログラムが理解できるようにDNCLは設計されている，ということです．この考え方は，教科「情報」の問題で使われるプログラム表記にも継承されているはずでしょう．

DNCLにせよプログラム表記にせよ，これを「プログラミング言語」として捉え，高校の授業でもこれらを教える必要があると誤解している人も少なくないと聞きます．DNCLやプログラム表記を授業で教えてもかまいません（もちろん教えなくても何ら問題はありません）が，それだけにとどまっていてはいけません．高校の授業で真に重要なことは，世の中で広く利用されているプログラミング言語を用いて，実際に役に立つプログラムが動くことを体感しながら学習を進めることではないでしょうか．

以上をふまえた上で，DNCLとプログラム表記を対比させて考えていきたいと思います．

DNCLとプログラム表記の違い

教科「情報」のプログラムで用いるとされているプログラム表記は，「情報関係基礎」で用いられているDNCLと見た目はよく似ていますが，微妙な違いがあります．両者の主な違いを，**表1**にまとめました．

表1　DNCL とプログラム表記の主な違い

項目	DNCL	プログラム表記
代入	代入先 ← 値	代入先 = 値
算術演算	乗算 ×	乗算 *
比較演算	等しい =，等しくない ≠，以上 ≧，以下 ≦	等しい ==，等しくない !=，以上 >=，以下 <=
論理演算	かつ，または，でない	and，or，not
表示	…と…を表示する	表示する (…，"と"，…)
条件分岐	もし … ならば ｜ … を実行し，そうでなければ ｜ … を実行する	もし … ならば: ｜ … そうでなければ: ｜…
順次繰返し	…を…から…まで…ずつ増やしながら， ｜ … を繰り返す	…を…から…まで…ずつ増やしながら繰り返す: ｜…
条件繰返し	…の間， ｜ … を繰り返す	…の間繰り返す: ｜…

そもそもDNCLは，日本語が分かり，高校までの数学の記法と少数のプログラミングの基本概念（変数，配列，代入，繰り返しや条件分岐など） を理解していさえすれば，DNCLそのものの仕様や構文を厳密には知らなくても，プログラムを「見れば意味が分かる」ように設計されています．そのため，プログラム中の式は数学における記法とほぼ同じように記述します．たとえば乗算は「＊」でなくて「×」を，等号付きの比較は「<=」でなくて「≦」を，等しくないことは「!=」でなくて「≠」を使います．論理演算も日本語の「かつ」などを使います．対してプログラム表記の方は，より現実のプログラミング言語の表記に近づき，アスキー文字による「＊」，「<=」，「!=」，「and」などを用います．変数への代入操作も「=」を用いて記述するようになりました．出力については，DNCLでは「・・・を表示する」と日本語で記述していたものを，プログラ

ム表記では「表示する（・・・）」のような関数呼出しで記述します．条件分岐や繰返しのような制御構文も，行末に「:」をつけ，DNCLでは必要であった制御構文を終わらせる最後の行（「を繰り返す」など）が不要になるなど，よりPythonに近くなりました．これらの違いから分かるように，DNCLは「見れば意味が分かる」だったのが，プログラム表記は言語仕様や構文の知識を前提とする方向にシフトしたようです．その意味でDNCLとプログラム表記は「似て非なる」言語と言えるでしょう．

配列の扱い

このように，プログラム表記は言語仕様や構文の知識が受験生にあることを前提としているように思えますが，このことが問題にも影響していることを示す典型的な例を，配列の扱いに見ることができます．

DNCLもプログラム表記も両方とも配列の添字は0から始まり，場合によっては1以上の添字の範囲だけを使うことがあるとなっています．また配列への値の設定は要素値を括弧で並べた一括代入が可能で，DNCLでは「A ← {7,8,9}」と，プログラム表記では「A = [7,8,9]」と書くことができます．いずれにおいても，こうするとA[0]には7が，A[1]には8が，A[2]には9が設定されます．

2006年以降2023年までの「情報関係基礎」の本試験第3問（すべてDNCLを利用）の問題について配列の使われ方を調べました．その結果を**表2**に示します．この表の初期設定の欄は，問題を解く前提として配列要素の初期値が設定されている必要がある場合，その初期値の設定方法を次のいずれかで示しています．

- 図表 ― 本問のように，図あるいは表を用いて示している．
- 問題文 ― 問題文中に文章として書かれている．
- 逐一代入 ― 「A[1] ← 7」のような添字を指定した代入を必要個数並べている．

表2 「情報関係基礎」第3問における配列の使われ方

年	問題の内容	次元	添字の範囲	初期設定
2006	生命体	1次元，2次元	1以上	図表
2007	ブロックゲーム	2次元	1以上	図表
2008	並べかえ	1次元	1以上	逐一代入
2009	素因数分解	1次元	2以上	―
2010	漢数字	1次元	1以上	逐一代入
2011	試験点数の集計	1次元	0以上，1以上	図表
2012	天秤	1次元	0以上	逐一代入
2013	利益最大	1次元	1以上	図表
2014	太鼓	1次元	1以上	図表
2015	盤上ゲーム	2次元	0以上，1以上	図表
2016	箱詰め	1次元	1以上	図表
2017	三角形の個数	1次元，2次元	1以上	図表
2018	迷路	2次元	1以上	図表
2019	グループ分け	1次元，2次元	1以上	図表
2020	宝さがし	1次元	1以上	問題文
2021	すごろく	1次元，2次元	0以上，1以上	図表
2022	あみだくじ	1次元	1以上	逐一代入
2023	ロープ飛び移り	1次元	1以上	図表

この表を見て分かるとおり，配列の添字は0から始まるというDNCLの仕様にかかわらず，ほとんどの問題では添字を1から使っており，0から使うような例は数えるほどしかありませんでした．初期設定に関しては，一括代入を用いたプログラムは皆無であり，ほとんどが添字を明示した図表や逐一代入を利用しています．図表や逐一代入を用いれば，配列の添字はいくつから始まるかという言語仕様の知識を必要としません．使う添字が図表やプログラム中に陽に示されているので，「見れば意味が分かる」プログラムになります．

一方で，プログラム表記を用いる教科「情報」の問題はまだ数が少なく，大学入試センターによる公式のものは次の3つしか存在しません．

- 試作問題（検討用イメージ） の第5問（2020年）
- 今回紹介しているサンプル問題の第2問（2021年）
- 試作問題の第3問（2022年）

この3問について，表2と同様の事項を調べた結果を**表3**に示します．

表3　プログラム表記を用いる問題における配列の使われ方

問題	問題の内容	次元	添字の範囲	初期設定
検討用イメージ	シーザー暗号	1次元	0以上	図表，一括代入
サンプル問題	ドント方式	1次元	0以上	図表，一括代入
試作問題	上手なお金の払い方	1次元	0以上	一括代入

いかがでしょうか．プログラム表記を用いたすべての問題が，一括代入により添字を0から使っていることが分かります．問題数は3問と少ないのですが，表2で示した「情報関係基礎」の問題群とは明らかに異なる傾向を示しているのです．この結果を見て，筆者もかなり驚きました．一括代入を利用すれば添字は必然的に0から始まらざるを得ず，必要以上に言語仕様に依存した問題になってしまっているという印象を受けます．「情報関係基礎」は「見れば意味が分かる」というDNCLの特長を活かし，配列の添字の始まりを問題に応じて柔軟に使い分けていました．プログラム表記でも，一括代入を用いずに逐一代入を用いれば，添字の始まりは自由に設定できるのですが，今までの出題を見る限りにおいては，そのような柔軟性がなくなりつつあるように思います．プログラム表記は現実のプログラミング言語に近づいたという，DNCLとの「微妙な違い」が，問題の傾向の大きな差となって現れているのでしょうか．

似て非なる言語の影響

DNCLとプログラム表記は「似て非なる言語」であるため，世の中には両者を混同する人が多数出ることを筆者は懸念しています．再度整理しておくと，次のようになります．

- 現在の大学入試共通テストの「情報関係基礎」で用いられている言語＝共通テスト手順記述標準言語（略称DNCL）
- 2025年からの大学入試共通テストの「情報I」で用いられる予定の言語＝共通テスト用プログラム表記（本稿では「プログラム表記」と略記）

しかしこれらを混同して，2025年からの「情報」で使われる言語は共通テスト手順記述標準言語（DNCL）である，と思い違いをしている人が多数現れることを，筆者は懸念しています．しかし，ある意味それは仕方のないことかもしれません．というのは，2021年の文献[4]　のサンプル問題（本稿で解説したドント方式の問題）に関して触れている部分で，「この問題の中で使用しているプログラミング言語は，（中略）大学入試センター独自の日本語表記の疑似言語（以下DNCL）としている」と述べているように「DNCL」という言葉を使っているからです（ただし，続く部分では「このサンプル問題では，（中略）一部

表記を改めたDNCLを使用している」と断わっています）．その後，2023年の文献[1)]では，「共通テスト用プログラム表記」と呼び「DNCL」という言葉は使わなくなりました（本稿はこれにならいました）．しかし，もしも上の懸念で述べたことが本当に起こっているならば，その思い違いを払拭するのは容易なことではないでしょう．しかしそのための努力をする必要があると考えます．

「情報I」の試験対策として，「情報関係基礎」のDNCLによる過去問を解いて学習する受験生も多いと思います．もしかしたら，過去問を中心に勉強して思い違いをしたまま受験する受験生は，共通テスト用プログラム表記という「似て非なる言語」を見て面喰らうかもしれません．せっかく導入が決まった「情報」の共通テストの本番で，思い違いによる混乱が起こらないこと願わずにはいられません．

参考文献

1）水野修治：令和7年度大学入学共通テスト『情報』の実施に向けて～問題作成方針に関する検討の方向性と試作問題～，情報処理，Vol.64, No.2, pp.74-77 (Feb. 2023).
https://doi.org/10.20729/00223448
2）大学入試センター：共通テスト手順記述標準言語（DNCL）の説明（2022）．
3）西田知博，川合　慧：大学入試センター試験とプログラミング言語，情報処理，Vol.50, No.10, pp.1013-1016 (Oct. 2009).
http://id.nii.ac.jp/1001/00067213/
4）水野修治：大学入学共通テスト新科目「情報」～これまでの経緯とサンプル問題～，情報処理，Vol.62, No.7, pp.326-330 (July 2021).
https://doi.org/10.20729/00211554

（2023年6月16日受付）
（2023年8月1日note公開）

■**岩崎英哉**（正会員）
1983年東京大学工学部計数工学科卒業．1988年東京大学大学院工学系研究科情報工学専攻博士課程修了．同年同大学工学部助手．1993年同大学教育用計算機センター助教授．その後，東京農工大学工学部助教授，東京大学大学院工学系研究科助教授，電気通信大学助教授，教授を経て，2022年より明治大学理工学部教授．電気通信大学名誉教授．工学博士．プログラミング言語，システムソフトウェアなどの研究に従事．