「情報」大学入学共通テスト対策

共通テスト手順記述標準言語

問題集

～DNCL 擬似プログラミング言語～

2022年１月２５日版 ※随時追加予定

この問題集の目的

プログラミングは自分自身で「考える」ことで力がつくので
解説を見る前に必ず自力で問題を解いてください。

いずれも『情報Ⅰ動画教科書』で講座提供中！

PythonやJavaScriptなどでプログラミングの基礎を習得した後
大学入学共通テスト（情報関係基礎・サンプル問題など）演習を
行ってもかなり難しく感じると思います。
本問題集は、共通テストの過去問演習に入る前に、
DNCL（共通テスト手順記述標準言語）の基礎を習得することを
目的に作成しています。

ご利用にあたって

情報教育の底上げが目的なので、資料を修正して、
学校・塾（営利目的含む）の授業等で利用して頂いて問題ありません。
私への連絡不要ですが、利用する際には、
YouTubeチャンネル・情報Ⅰ動画教科書・IT用語動画辞典を
紹介してもらえると嬉しいです。

【突破ロドットコム YouTubeチャンネル】
https://www.youtube.com/c/toppakou

【情報Ⅰ動画教科書】
https://toppakou.com/infol/

【IT用語動画辞典】
https://toppakou.com/ITWORD/

問１

次のプログラムを実行した場合の
出力結果を答えてください。

```
x ← 2
y ← 3
もし x ≦ 2 ならば
    x ← x + 2
    y ← y - 1
を実行する
“x=” と x と “ y=” y を表示する
```

出力結果

◆解説動画◆
https://youtu.be/BF6JLjoufq0

問２

次の空欄部分を埋めてください。

配列名：Tokuten

要素番号（添字）	1	2	3	4	5	6
	34	56	43	55	87	76

Tokuten[3]とした場合、
取得できる値は、［　　］である。

２番目の５６の数字を
取り出したい場合は、
［　　］と指定する。

Tokuten[1]+Tokuten[5]は
［　　］である。

◆解説動画◆
https://youtu.be/Xbj6KHKqrw4

問３

次のプログラムを実行した場合の
出力結果を答えてください。

```
x ← 2
y ← 3
もし x ＜ 2 ならば
    x ← x + 2
    y ← y - 1
を実行する
“x=” と x と “ y=” y を表示する
```

出力結果

◆解説動画◆
https://youtu.be/mNAzHBRVyzw

問４

次の空欄部分を埋めてください。

配列名：Tokuten

要素番号（添字）

0	1	2	3	4	5
34	56	43	55	87	76

Tokuten[3]とした場合、
取得できる値は、［　　］である。

２番目の５６の数字を
取り出したい場合は、
［　　　　］と指定する。

Tokuten[1]+Tokuten[5]は
［　　］である。

◆解説動画◆
https://youtu.be/8OlEBYWTgeI

問５

次の空欄部分を埋めてください。

二次元配列名：Gyoretu

要素番号（添字）

	0	1	2	3	4	5
0	34	56	43	55	87	76
1	67	98	33	12	89	77
2	54	45	78	99	78	34
3	29	67	23	82	11	12

Gyoretu[3,2]とした場合、
取得できる値は、［　　］である。

２行４列目の値を
取得したい場合
［　　　　］と指定する。

◆解説動画◆
https://youtu.be/4YDGwDt5kxI

問６

次のプログラムを実行した場合の
出力結果を答えてください。

```
x ← 8, gokei ← 0

繰り返し,
    gokei ← gokei + x
    x ← x + 1
    x を表示する。
を,x ≧ 10になるまで実行する
```

出力結果

◆解説動画◆
https://youtu.be/YEkHRmKHWFs

問７

```
x ← 8, gokei ←  0

繰り返し,
  gokei ←  gokei + x
  x ← x + 1
  x を表示する。
を,x ≧ 10になるまで実行する
```

上記のプログラムと
同じ意味になるように
空欄部分を埋めてください
終了条件→継続条件に変更

```
x ← 8, gokei ←  0

繰り返し,
  gokei ←  gokei + x
  x ← x + 1
  x を表示する。
を,x [　] 10の間実行する
```

◆解説動画◆
https://youtu.be/bdNzTqhQ_i4

問８

次のプログラムを実行した場合の
出力結果を答えてください。

```
x ← 8, gokei ←  0

x < 10 の間,
  gokei ←  gokei + x
  x ← x + 1
  x を表示する。
を繰り返す
```

出力結果

◆解説動画◆
https://youtu.be/q2R79XbUmM4

問９

次のプログラムについて変数kazuに
代入される数値を答えてください。
kazuの初期値は7とする

```
kazu ← 1+ kazu / 2
```

答え：

```
kazu ← 1+ kazu ÷ 2
```

答え：

```
kazu ← 1 + kazu%2
```

答え：

```
kazu ← (2 + kazu) / 3
```

答え：

◆解説動画◆
https://youtu.be/hrgiPtyZiGU

問１０

次の判定式について
「真(True)」か「偽(False)」
いずれかで答えてください。

“ABC” ＝ “ABC”

答え：

“ABC” ＝ “abc”

答え：

“ABC” ≠ “ABC”

答え：

“ABC” ≠ “abc”

答え：

```
mozi ← “test”
mozi = “mozi”
```

答え：

◆解説動画◆
https://youtu.be/vICODCCHI_8

問１１

xが12以上27以下なら
真（True）となるように
空欄部分を埋めてください。

```
もし x □ 12 □ x □ 27 ならば
    x ← x + 2
を実行する
```

xが偶数、または負の値ならば
真（True）となるように
空欄部分を埋めてください。

```
もし x□□ = 0 □ x □ 0ならば
    x ← x + 2
を実行する
```

◆解説動画◆
https://youtu.be/JwROy2IXxNY

問１２

配列の要素番号（添字）が、
0から始まるとき
出力結果を答えてください。

```
Tokuten ← {87, 45, 72, 100}

xを1から3まで2ずつ増やしながら
    Tokuten[x-1]  を表示する。
を繰り返す
```

出力結果

◆解説動画◆
https://youtu.be/MIuTvBD5Mfw

問１３

次のプログラムの
出力結果を答えてください。

```
xを1から2まで1ずつ増やしながら
    x を表示する。
    yを0から1まで 1 ずつ増やしながら
        y を表示する。
    を繰り返す

を繰り返す
```

出力結果

◆解説動画◆
https://youtu.be/3n6mMI1qVIY

問１４

配列の要素番号（添字）が、
0から始まるとき
出力結果を答えてください。

```
Tokuten ← {87, 45, 72, 100}

xを3から1まで2ずつ減らしながら
    Tokuten[x-1]  を表示する。
を繰り返す
```

出力結果

◆解説動画◆
https://youtu.be/ybk4UOD6100

問１５

次のプログラムを実行した場合の
出力結果を答えてください。

```
x ← 1

もし x = 3 ならば
    x ← x + 2
を実行し、そうでなくもし x > 1ならば
    x ← x - 1
を実行し、そうでなければ
    x ← x + 3
を実行する

x を表示する
```

出力結果

◆解説動画◆
https://youtu.be/MB5xheY8ASA

問１６

次の空欄部分を埋めてください。

値n が奇数のとき真を返し，
そうでないとき偽を返す
関数「奇数(n)」を用意する。

奇数(3)は「　　」を返却する。
奇数(4)は「　　」を返却する。

指定された値nを2進数で表示する
関数「二進数で表示する（n）」
を用意する。

二進数で表示する（3）　について
関数の処理内で「　　」が表示され
る。

◆解説動画◆
https://youtu.be/ak5BJCDRitU

問１７

次のプログラムを実行した場合の
出力結果を答えてください。

```
x ← 3, y ← 5

面積を表示する(x, y)

x を表示する

y を表示する
```

```
関数 面積を表示する(tate, yoko)を
    menseki = tate x yoko
    menseki を表示する
と定義する
```

出力結果

代入は基本的に「←」ですが、
問題によっては「=」が使われていることもあるので、
今回一部の代入処理で「=」を使っています。（←と同じ意味で扱ってください）

◆解説動画◆
https://youtu.be/ef7z1nZySf0

問１８

次のプログラムを実行した場合の
出力結果を答えてください。

```
x ← 4, y ← 5

kekka = 面積を計算する(x, y)

kekka を表示する
```

```
関数 面積を計算する(tate, yoko)を
    menseki = tate x yoko
    menseki を返却する
と定義する
```

出力結果

◆解説動画◆
https://youtu.be/zqNiHjEIDtM

問１９

受け渡された配列の中の要素を表示する関数の空欄部分を埋めてください。

※要素数（n)は受け渡された配列の
　要素数を返す関数である。
※要素番号は１から始まるものとする。

```
Tokuten ← {34, 56, 43}

配列を表示する(Tokuten)
```

```
関数 配列を表示する(Hai)を

　jを1から要素数(Hai) まで1ずつ増やしながら
　|　[　　] を表示する
　を繰り返す

と定義する
```

出力結果

◆解説動画◆

https://youtu.be/uYd9tj3EtCE

問２０

受け渡された配列の中の要素を表示する関数の空欄部分を埋めてください。

※要素数（n)は受け渡された配列の
　要素数を返す関数である。
※要素番号は０から始まるものとする。

```
Tokuten ← {34, 56, 43}

配列を表示する(Tokuten)
```

```
関数 配列を表示する(Hai)を

　jを1から要素数(Hai) まで1ずつ増やしながら
　|　[　　] を表示する
　を繰り返す

と定義する
```

出力結果

◆解説動画◆

https://youtu.be/NXOjk6KqD5o

問２１

問題は随時増やしていく予定です

2022年1月25日版はここまで

◆解説動画◆

参考・引用文献

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(144KB)」(2022/1/4)

https://www.dnc.ac.jp/albums/abm.php?f=abm00040701.pdf&n=%E4%BB%A4%E5%92%8C%EF%BC%93%E5%B9%B4%E5%BA%A6%E8%A9%A6%E9%A8%93%E5%85%B1%E9%80%9A%E3%83%86%E3%82%B9%E3%83%88%E6%89%8B%E9%A0%86%E8%A8%98%E8%BF%B0%E6%A8%99%E6%BA%96%E8%A8%80%E8%AA%9E%28DNCL%29%E3%81%AE%E8%AA%AC%E6%98%8E.pdf

「情報」大学入学共通テスト対策

共通テスト手順記述標準言語

DNCLとは？

～以下動画の説明資料になります～

https://youtu.be/0BlhCZ6T0oM

朝日新聞デジタル ＞ 記事

受験特集 ＞

共通テストでの「情報」追加を正式決定　2025年から

桑原紀彦　2021年7月30日 12時04分

シェア　ツイート　ブックマーク　メール　印刷

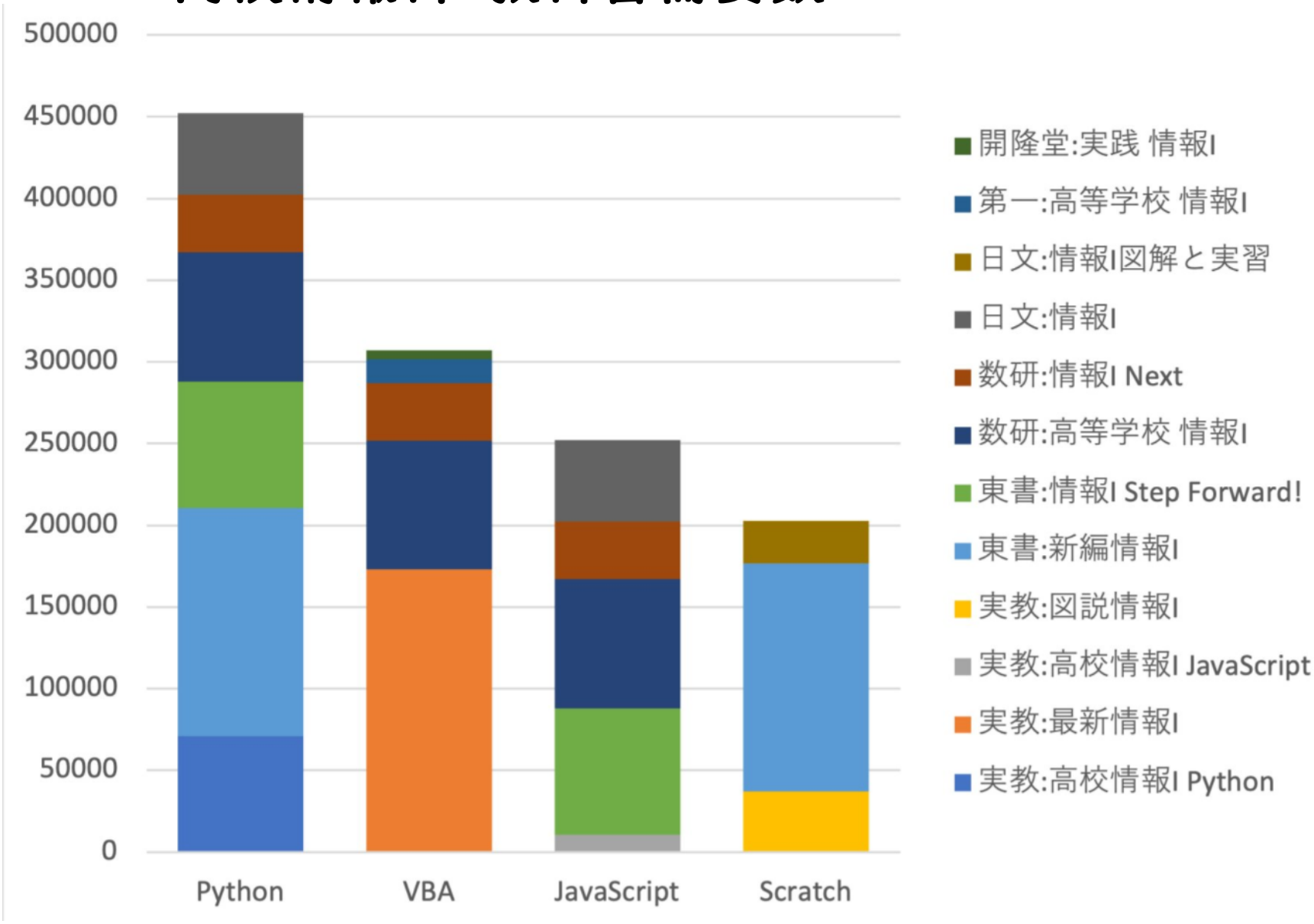
list　41

文部科学省 は30日、2025年以降の 大学入学共通テスト の出題科目を正式に決めた。来年度から導入される高校の新 学習指導要領 を反映し、「情報」を出題教科に追加。一方で、国語・数学での記述式問題導入と、英語民間試験の活用については、いずれも見送りを正式に決めた。

各大学は今後、25年以降の共通テストで受験生にどの教科・科目を課すかを検討する。 文科省 のルールでは、出題科目などに大きな変更がある場合、実施の2年前に

引用：朝日新聞デジタル
https://www.asahi.com/articles/ASP7Z3WFKP7ZUTIL016.html

高校情報科 教科書需要数



引用：https://twitter.com/nakano_lab/status/1471322239140532226

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。

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。

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。

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。

```
(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

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。

```
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

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。

［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

「共通テスト手順記述標準言語（DNCL）の説明」（2021年１月現在）

「情報関係基礎」で用いられる

独立行政法人大学入試センター

2021 年 1 月

高等学校におけるアルゴリズムやプログラムに関する教育では，採用されるプログラミング言語は多様で，プログラミングの実習時間も異なります。大学入試センターではこのような事情を考慮し，「情報関係基礎」の出題にあたり，共通テスト用の手順記述言語 (DNCL) を使用します。

以下，参考のために DNCL の基本を説明します。しかしながら，問題文の記述を簡潔にするなどの理由で，この説明文書の記述内容に従わない形式で出題することもあります。したがって，「情報関係基礎」の受験に際しては，当該問題文の中の説明や指示に注意し，それらに沿って解答してください。

目次

問題文の記述を簡潔にするなどの理由で，この説明文書の記述内容に従わない形式で出題されることもあります。

問題文の中の説明や指示に注意し，それらに沿って解答してください。

大学入試センター

令和３年度 共通テスト（1月16日・17日）の問題|大学入試センター (dnc.ac.jp)

https://www.dnc.ac.jp/kyotsu/kako_shiken_jouhou/r3/jisshikekka/r3_dai1_mondai.html

変数と値

変数名は，英字で始まる英数字と『_ 』の並びで表す。

特に指示がない限り、

・小文字で始まる変数は通常の変数表す

例） kosu

※令和３年の情報関係基礎で、この変数が使われているのは宝探しゲームなので、
読みとしては、宝の「個数（コスウ）」が妥当ですが、
動画説明について、今回はローマ字読みそのままの「kosu(コス)」と読ませてもらっています。
音声編集がすべて終わった後で気づきました・・

・アンダースコア『_』を文字途中で入れることも可能

例） kosu_gokei

・大文字で始まる変数は配列を表す

例） Tokuten

・すべて大文字の変数は実行中に変化しない値を表す

※一般的には「定数」と呼んだりする

例） TOKUTEN

変数と値（一次元配列／二次元配列）

・大文字で始まる変数は配列を表す。

要素番号（添字）

０以上の整数
※問題によって１以上から
始まる場合もある

配列名：Tokuten

0	1	2	3	4	5
３４	５６	４３	５５	８７	７６

Tokuten[2]
４３の値が取得できる

1	2	3	4	5	5
３４	５６	４３	５５	８７	７６

添字が１から始まる場合
Tokuten[2]
５６の値が取得できる

二次元配列名：Gyoretu

添字をカンマで区切り、
行と列の添字を指定

	0	1	2	3	4	5
0	３４	５６	４３	５５	８７	７６
1	６７	９８	３３	１２	８９	７７
2	５４	４５	７８	９９	７８	３４
3	２９	６７	２３	８２	１１	１２

Gyoretu[3,2]
２３の値が取得できる

変数と値（数値・文字列の表し方）

・数値は特に断らない限り、10 進法で表す。

例）　100
　　qq.qqq

・文字の並びを『「』と『」』，または，『″』と『″』でくくって表す。

例）　「見つかりました」
　　"It was found."

数値や文字列や変数の値を表示する表示文

・複数の値を表示する場合は『と』で区切って並べ，
最後に『を表示する』と書く。

例）　「整いました」と表示する

実行結果：整いました

例）　kosu = 3
kosuと「個見つかった」を表示する

実行結果：3個見つかった

例）　x = 5
y = -1
"(" とx と", " とy と")" を表示する

実行結果：(5, -1)

代入文①

『←』の左辺に変数または添字付きの配列を，右辺に代入する値を書く。

例）　kosuに3を代入する場合
　　kosu ← 3

例）　Tokuten配列の添字４に100を代入する場合
　　Tokuten[4] ← 100

例）配列のすべての要素に纏めて同じ値を代入する場合
　　Tokuten のすべての要素に0 を代入する

例）Tokuten配列の１番目に87、２番目に45
　　３番目に72、４番目に100を代入する場合
　　Tokuten ← {87, 45, 72, 100}

代入文②

『←』の左辺に変数または添字付きの配列を，右辺に代入する値を書く。

例）複数の代入文を，『，　』で区切りながら，横に並べることができる。
　　代入文は左から順に実行される。

kosu_gokei ← kosu, tokuten ← kosu × (kosu + 1)

1番目　　2番目

例）同じ変数に対する加算や減算を伴う代入（インクリメントやデクリメント）は，
　　『～を～増やす』や『～を～減らす』によって表すこともできる。
　　『kosu をl 増やす』は『kosu ← kosu + 1』と同じ
　　『saihu をsyuppi 減らす』は『saihu ← saihu − syuppi』と同じ

例）外部から入力された値を代入する表現方法
　　x ←【外部からの入力】

算術演算

加減乗除の四則演算は，『＋』，『－』，『×』，『/ 』で指定する。
整数の除算では，商を『÷』で，余りを『%』で計算することができる。

例）atai ← 7 / 2　　変数ataiに3.5が代入される。

例）syo ← 7 ÷ 2　　変数syoに3が代入される。

例）amari ← 10%3　　変数amariに1が代入される。

演算子の優先順位

・複数の演算子を使った式の計算では，基本的に左側の演算子が先に計算されるが、『×』，『/ 』，『÷』，『%』は，『＋』，『－』より先に計算される。
・丸括弧『(』と『)』で式をくくって，演算の順序を明示することができる。

例） kosu ← 1＋ kazu ÷ 3　　kosu ← 1＋ (kazu ÷ 3)

1番目　　1番目

kosu ← (1＋ kazu) ÷ 3

1番目

例） heikin ← (hidari ＋ migi)÷ 2　　heikin ← hidari＋ migi ÷ 2

1番目　　1番目

比較演算（数値）

数値の比較演算は，
『＝』，『≠』（あるいは『≒』），『＞』，『≧』，『≦』，『＜』で指定する。
演算結果は，真か偽の値となる。

例：kosu ＞ 3　　kosu が3 より大きければ真となる。

例：ninzu×2 ≦ 8　　ninzu の2 倍が8 以下であれば真となる。

例：kaisu ≠ 0　　kaisu が0 でなければ真となる。

比較演算（文字列）

文字列の比較演算は，『＝』，『≠』（あるいは『≒』）を利用することができる。
『＝』は，左辺と右辺が同じ文字列の場合に真となり，それ以外の場合は偽となる。
『≠』（あるいは『≒』）は，左辺と右辺が異なる文字列の場合に真となり，
それ以外の場合（同じ文字列の場合）は偽となる。

例:「あいうえお」＝「あいうえお」　真となる。

例:「あいうえお」＝「あいう」　偽となる。

例:“ABC”＝“ABC”　真となる。

例:“ABC”＝“abc”　偽となる。

例:「あいうえお」≠「あいうえお」
≠（ノットイコール）はイコールではないという意味で、今回は「偽」となる。

例:「あいうえお」≠「あいう」　真となる。

例:“ABC”≠“ABC”　偽となる。

例:“ABC”≠“abc”　真となる。

論理演算

真か偽を返す式に対する演算で，『かつ』，『または』，『でない』の演算子で指定する。

『〈式1〉かつ〈式2〉』は，〈式1〉と〈式2〉の結果がいずれも真である場合に
真となり，それ以外の場合は偽となる。

例：kosu ≧ 12 かつkosu ≦ 27　kosu が12 以上27 以下なら真となる。

『〈式1〉または〈式2〉』は，〈式1〉と〈式2〉の結果のどちらかが真である場合に
真となり，それ以外の場合は偽となる。

例：kosu%2 ＝ 0 またはkosu ＜ 0
　kosuを２で割った余りが０となる数は偶数なのでkosu が偶数か負の値なら真となる

『〈式〉でない』は，〈式〉の結果が真である場合に偽となり，偽の場合は真となる。

例：kosu ＞ 75 でない　kosu が75 より大きくなければ真となる

論理演算子に優先順位はなく，左側の論理演算が先に実行されるが，
丸括弧『(』と『)』で，演算の順序を指定することができる。

例：kosu ＞ 12 かつ kosu ＜ 27でない　kosu ＞ 12 かつ(kosu ＜ 27 でない)
　　１番目　　１番目

制御文

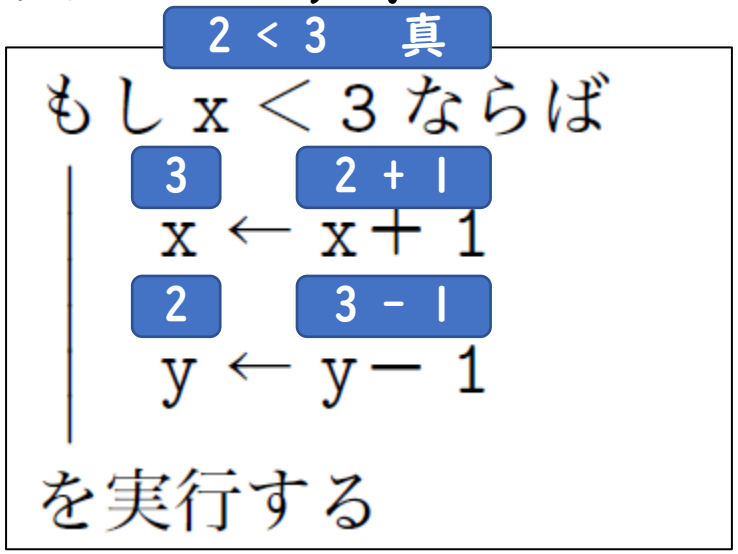
条件分岐文、条件繰返し文、順次繰返し文をまとめて「制御文」と呼ぶ。

条件分岐文

〈条件〉が成り立つかどうかによって，実行する処理を切り替える。
〈条件〉が成り立つときにある処理を実行し，
〈条件〉が成り立たないときに実行する処理がない場合は，
『ならば』で指定する。

```
もし 〈条件〉 ならば
|    〈処理〉
|
を実行する
```

例 x = 2, y = 3



〈処理〉が1 行しかない場合、以下の様に書くことも可能。

```
もし 〈条件〉 ならば 〈処理〉 を実行する
```

例：もしx ＜ 3 ならばx ← x+ 1 を実行する

条件分岐文

〈条件〉が成り立つときにある処理を実行し，
〈条件〉が成り立たないときに別の処理を実行する場合は，
『ならば』と『そうでなければ』を組み合わせて指定する。

もし **〈条件〉** ならば
　　〈処理 1〉
を実行し，そうでなければ
　　〈処理 2〉
を実行する

例

4＜3　偽

もし x ＜ 3 ならば
　　2　　1 + 1
　　x ← x + 1
を実行し，そうでなければ
　　3　　4 − 1
　　x ← x − 1
を実行する

改行位置によって実行結果が変わらないため，各処理が1 行で書ける場合には，次のように書くこともある。

もし **〈条件〉** ならば **〈処理 1〉** を実行し，
そうでなければ **〈処理 2〉** を実行する

例:　もし x ＜ 3 ならば x ← x + 1 を実行し，
　　そうでなければ x ← x − 1 を実行する

条件分岐文

条件分岐の中で複数の条件で実行する処理を切り替えたい場合は，
『ならば』と『そうでなければ』の間に
『そうでなくもし』を使って条件を追加する。

もし 〈条件 1〉 ならば
　　〈処理 1〉
を実行し，そうでなくもし 〈条件 2〉 ならば
　　〈処理 2〉
を実行し，そうでなければ
　　〈処理 3〉
を実行する

例 x = 4, y = 3

もし x ＝ 3 ならば
　　x ← x＋ 1
を実行し，そうでなくもし y ＞ 2 ならば
　　y ← y＋ 1
を実行し，そうでなければ
　　y ← y－ 1
を実行する

3 > 2　真
4
3 + 1

条件分岐文

改行位置によって実行結果が変わらないため，各処理が1 行で書ける場合には，次のように書くこともある。

もし **〈条件 1〉** ならば **〈処理 1〉** を実行し，
そうでなくもし **〈条件 2〉** ならば **〈処理 2〉** を実行し，
そうでなければ **〈処理 3〉** を実行する

例:　もし x ＝ 3 ならば x ← x＋ 1 を実行し，
そうでなくもし y ＞ 2 ならば y ← y＋ 1 を実行し，
そうでなければ y ← y－ 1 を実行する

条件繰返し文（前判定）

〈条件〉が成り立つ間，　〈処理〉を繰り返し実行する。
〈処理〉を実行する前に〈条件〉が成り立つかどうか判定されるため，
〈処理〉が1 回も実行されないことがある。

〈条件〉 の間，
　〈処理〉
を繰り返す

例 x = 8, gokei = 0

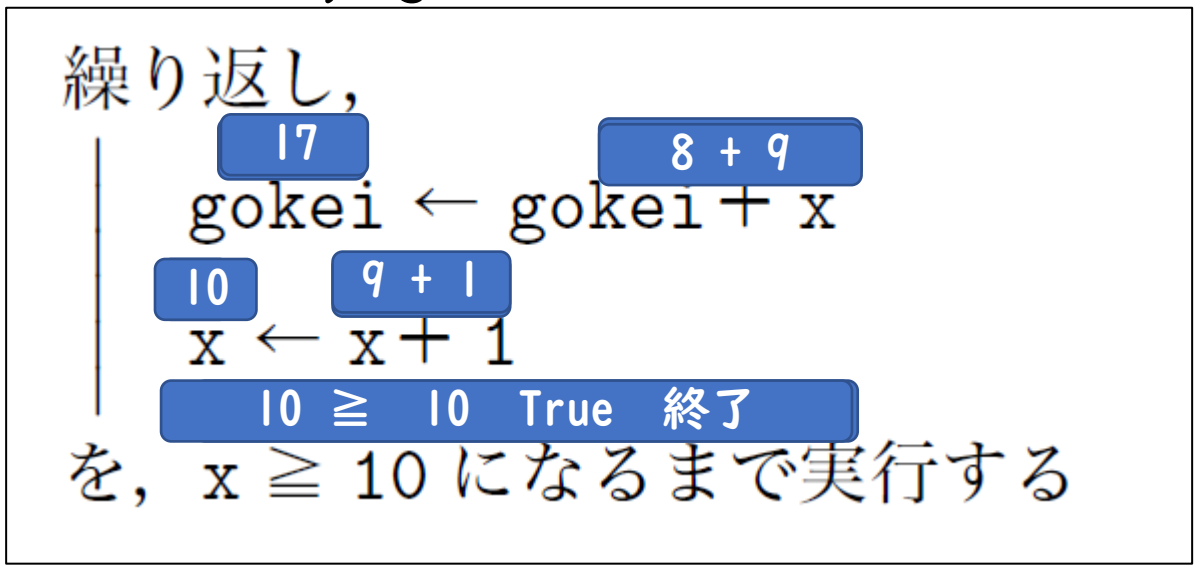
10
x < 10 の間，
　17　　8 + 9
　gokei ← gokei + x
　10　9
　x ← x + 1
を繰り返す

条件繰返し文（後判定）

〈処理〉を実行した後に〈条件〉が成り立つかどうか判定されるため，
に当てはまらない場合も、〈処理〉は少なくとも1 回は実行される。

繰り返し，
　　〈処理〉
を，〈条件〉になるまで実行する

例 x = 8, gokei = 0



条件繰返し文（後判定）

〈処理〉を実行した後に〈条件〉が成り立つかどうか判定されるため，
〈条件〉に当てはまらない場合も、〈処理〉は少なくとも1 回は実行される。

繰り返し，
　〈処理〉
を，〈条件〉になるまで実行する

例 x = 11，gokei = 0

繰り返し，
　gokei ← gokei + x　（11 → 0 + 11）
　x ← x + 1　（12 → 11 + 1）
　12 ≧ 10　True　終了
を，x ≧ 10 になるまで実行する

順次繰返し文

〈変数〉の値を増やしながら，〈処理〉を繰返し実行する。

```
〈変数〉 を 〈初期値〉 から 〈終了値〉 まで 〈差分〉 ずつ増やしながら，
    〈処理〉
を繰り返す
```

手順１　〈変数〉に〈初期値〉が代入される

手順２　変数と終了値を比較し〈変数〉の値が
〈終了値〉よりも大きければ，繰り返しを終了する

手順３　〈処理〉を実行し，〈変数〉の値に〈差分〉を加え，手順2 に戻る

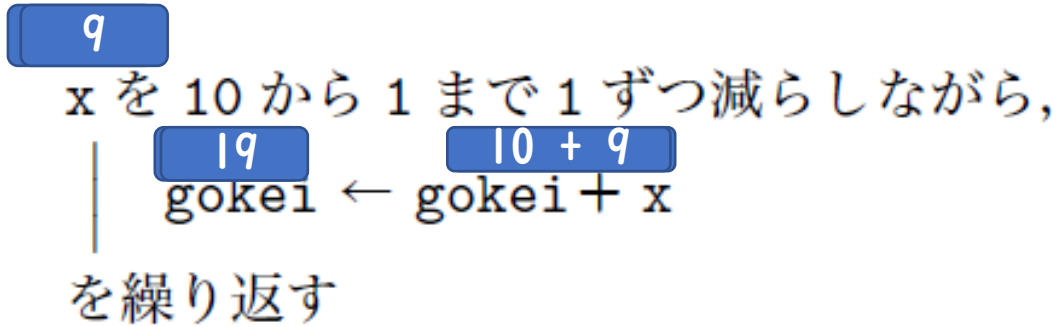
例　goukei = 0

```
2
x を 1 から 10 まで 1 ずつ増やしながら，
    3          1 + 2
    gokei ← gokei + x
を繰り返す
```

順次繰返し文

〈変数〉の値を減らしながら，〈処理〉を繰返し実行する。

例　gokei = 0



関数（値を返却する例）

指定された値の二乗の値を返す関数「二乗」を用意する

y←二乗（x）　xを２乗した値がyに代入される

パラメータ・引数

値m のn 乗の値を返す関数「べき乗(m, n)」を用意する

z←べき乗（x, y）　xをy乗した値がzに代入される

値m以上 値n以下の整数をランダムに一つ返す関数「乱数(m, n)」を用意する

r ← 乱数(1, 6)　r に1 から6 までの整数のうちいずれかが代入される

値n が奇数のとき真を返し，
そうでないとき偽を返す関数「奇数(n)」を用意する

奇数（3）は 真（True）
奇数（4）は 偽（False）

関数（値を返却しない例）

指定された値を2進数で表示する関数「二進数で表示する」を用意する

二進数で表示する（3）　　関数の処理内で11が表示される
※10進数の３を２進数に変換すると１１となる

次のスライ ドからは以下の資料を そのままの形で引用致します。

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(144KB)」(2022/1/4)
https://www.dnc.ac.jp/albums/abm.php?f=abm00040701.pdf&n=%E4%BB%A4%E5%92%8C%EF%BC%93%E5%B9%B4%E5%BA%A6%E8%A9%A6%E9%A8%93%E5%85%B1%E9%80%9A%E3%83%86%E3%82%B9%E3%83%88%E6%89%8B%E9%A0%86%E8%A8%98%E8%BF%B0%E6%A8%99%E6%BA%96%E8%A8%80%E8%AA%9E%28DNCL%29%E3%81%AE%E8%AA%AC%E6%98%8E.pdf

共通テスト手順記述標準言語 (DNCL) の説明

独立行政法人大学入試センター

2021 年 1 月

高等学校におけるアルゴリズムやプログラムに関する教育では，採用されるプログラミング言語は多様で，プログラミングの実習時間も異なります。大学入試センターではこのような事情を考慮し，「情報関係基礎」の出題にあたり，共通テスト用の手順記述言語 (DNCL) を使用します。

以下，参考のために DNCL の基本を説明します。しかしながら，問題文の記述を簡潔にするなどの理由で，この説明文書の記述内容に従わない形式で出題することもあります。したがって，「情報関係基礎」の受験に際しては，当該問題文の中の説明や指示に注意し，それらに沿って解答してください。

目次

1 変数と値

変数名は，英字で始まる英数字と『_』の並びです。

例: `kosu`， `kosu_gokei`， `Tokuten`

特に指示がない限り，小文字で始まる変数は通常の変数を表し，大文字で始まる変数は配列を表します。また，すべて大文字の変数は実行中に変化しない値を表します。

配列の要素は，要素の番号を添字で指定します。2 次元以上の場合は，添字を『, 』で区切ります。たとえば，(1 次元の) 配列 `Tokuten` や 2 次元配列 `Gyoretu` の要素は `Tokuten[2]` や `Gyoretu[3, 2]` のように表します。添字の値は 0 以上の整数ですが，問題によっては 1 以上の添字のみを扱います。

特に断らない限り，数値は 10 進法で表します。文字列は，文字の並びを 『「』と『」』，または，『"』と『"』でくくって表します。

例： `100`
例： `99.999`
例：`「見つかりました」`
例： `"It was found."`

2 表示文

表示文で数値や文字列や変数の値を表示します。表示文では，複数の値を表示する場合は『と』で区切って並べ，最後に『を表示する』と書きます。

例:`「整いました」`を表示する　(`「整いました」`と表示されます。)
例: `kosu` と「個見つかった」を表示する　(`kosu` が 3 のとき，`3 個見つかった`」と表示されます。)
例: `"("` と `x` と `", "` と `y` と `")"` を表示する　(`x` が 5，`y` が −1 のとき，`(5, -1)`」と表示されます。)

3　代入文

代入文は変数に値を設定します。『←』の左辺に変数または添字付きの配列を，右辺に代入する値を書きます。また，配列の各要素に同じ値をまとめて代入することや，他の配列の内容に置き換えることもできます。

例: kosu ← 3
例: Tokuten[4] ← 100
例: Tokuten のすべての要素に 0 を代入する
例: Tokuten ← {87, 45, 72, 100}

複数の代入文を，『, 』で区切りながら，横に並べることができます。この場合は，代入文は左から順に実行されます。

例: kosu_gokei ← kosu, tokuten ← kosu× (kosu＋ 1)

同じ変数に対する加算や減算を伴う代入（インクリメントやデクリメント）は，『〜を〜増やす』や『〜を〜減らす』によって表すこともできます。

例:『kosu を 1 増やす』　は『kosu ← kosu＋ 1』　と同じです。
例:『saihu を syuppi 減らす』は『saihu ← saihu− syuppi』と同じです。

外部から入力された値を代入するために，次のように記述することもあります。

例: x ←【外部からの入力】

4　演算

この節では，算術演算と比較演算，そして論理演算について説明します。比較演算やそれを組み合わせる論理演算は，条件分岐文（5.1 節）や条件繰返し文（5.2 節）の **《条件》** で使うことができます。

4.1　算術演算

加減乗除の四則演算は，『＋』，『−』，『×』，『 / 』で指定します。
整数の除算では，商を『÷』で，余りを『%』で計算することができます。

例: atai ← 7 / 2　（atai には 3.5 が代入されます。）
　　syo ← 7÷ 2　（syo には 3 が代入されます。）
　　amari ← 10% 3　（amari には 1 が代入されます。）

複数の演算子を使った式の計算では，基本的に左側の演算子が先に計算されますが，『×』，『 / 』，『÷』，『%』は，『＋』，『−』より先に計算されます。また，丸括弧『(』と『)』で式をくくって，演算の順序を明示することができます。

例: sogaku ← ne1− ne2− ne3　は，
　　sogaku ← (ne1− ne2)− ne3　と同じです。
例: kosu ← 1＋ kazu÷ 3　は，
　　kosu ← 1＋ (kazu÷ 3)　と同じです。
例: heikin ← (hidari＋ migi)÷ 2 は，
　　heikin ← hidari＋ migi÷ 2　と異なります。

4.2　比較演算

数値の比較演算は，『=』，『≠』（あるいは『≒』），『>』，『≧』，『≦』，『<』で指定します。演算結果は，真か偽の値となります。

例:　kosu ＞ 3　（kosu が 3 より大きければ真となります。）
例:　ninzu× 2 ≦ 8　（ninzu の 2 倍が 8 以下であれば真となります。）
例:　kaisu ≠ 0　（kaisu が 0 でなければ真となります。）

文字列の比較演算は，『=』，『≠』（あるいは『≒』）を利用することができます。『=』は，左辺と右辺が同じ文字列の場合に真となり，それ以外の場合は偽となります。『≠』（あるいは『≒』）は，左辺と右辺が異なる文字列の場合に真となり，それ以外の場合（同じ文字列の場合）は偽となります。

例:「あいうえお」＝「あいうえお」　（真となります。）
例:「あいうえお」＝「あいう」　（偽となります。）
例:　"ABC"＝"ABC"　（真となります。）
例:　"ABC"＝"abc"　（偽となります。）
例:「あいうえお」≠「あいうえお」　（偽となります。）

例: 「あいうえお」≠「あいう」　　(真となります。)
例: "ABC"≠"ABC"　　(偽となります。)
例: "ABC"≠"abc"　　(真となります。)

4.3 論理演算

論理演算は，真か偽を返す式に対する演算で，『かつ』，『または』，『でない』の演算子で指定します。論理演算子に優先順位はなく，左側の論理演算が先に実行されますが，丸括弧『(』と『)』で，演算の順序を指定することができます。

『**〈式 1〉** かつ **〈式 2〉**』は，**〈式 1〉**と**〈式 2〉**の結果がいずれも真である場合に真となり，それ以外の場合は偽となります。『**〈式 1〉** または **〈式 2〉**』は，**〈式 1〉**と**〈式 2〉**の結果のどちらかが真である場合に真となり，それ以外の場合は偽となります。『**〈式〉** でない』は，**〈式〉**の結果が真である場合に偽となり，偽の場合は真となります。

例: kosu ≧ 12 かつ kosu ≦ 27　　(kosu が 12 以上 27 以下なら真となります。)
例: kosu％ 2 ＝ 0 または kosu ＜ 0　　(kosu が偶数か負の値なら真となります。)
例: kosu ＞ 75 でない　　(kosu が 75 より大きくなければ真となります。)
例: kosu ＞ 12 かつ kosu ＜ 27 でない　　は，
(kosu ＞ 12 かつ kosu ＜ 27) でない　　と同じです。(左側の論理演算子が先に実行されるため。)
例: kosu ＞ 12 かつ kosu ＜ 27 でない　　は，
kosu ＞ 12 かつ (kosu ＜ 27 でない)　　と異なります。

5 制御文

条件分岐文 (5.1 節) や条件繰返し文 (5.2 節)，順次繰返し文 (5.3 節) をまとめて制御文と呼びます。制御文の中の **〈処理〉** として，表示文 (2 節)，代入文 (3 節)，値を返さない関数 (6.2 節)，条件分岐文，順次繰返し文，条件繰返し文を，一つ以上並べて使うことができます。また，条件分岐文や条件繰返し文の中の **〈条件〉** として，比較演算 (4.2 節) と論理演算 (4.3 節) を使用することができます。

5.1 条件分岐文

条件分岐文は，**〈条件〉** が成り立つかどうかによって，実行する処理を切り替えます。

〈条件〉 が成り立つときにある処理を実行し，**〈条件〉** が成り立たないときに実行する処理がない場合は，次のように『ならば』で指定します。

《一般形》

もし **〈条件〉** ならば
　〈処理〉
を実行する

例:　もし x ＜ 3 ならば
　　x ← x＋ 1
　　y ← y− 1
　を実行する

〈処理〉 が 1 行しかない場合は，次のように全体を 1 行で書くこともできます。

《一般形》

もし **〈条件〉** ならば **〈処理〉** を実行する

例:　もし x ＜ 3 ならば x ← x＋ 1 を実行する

〈条件〉 が成り立つときにある処理を実行し，**〈条件〉** が成り立たないときに別の処理を実行する場合は，次のように『ならば』と『そうでなければ』を組み合わせて指定します。

《一般形》

もし **〈条件〉** ならば
　〈処理 1〉
を実行し，そうでなければ
　〈処理 2〉
を実行する

例:　もし x＜3 ならば
　　x ← x＋1
　を実行し，そうでなければ
　　x ← x－1
　を実行する

改行位置によって実行結果が変わらないため，各処理が 1 行で書ける場合には，次のように書くこともあります。

《一般形》

もし **〈条件〉** ならば **〈処理 1〉** を実行し，
そうでなければ **〈処理 2〉** を実行する

例:　もし x＜3 ならば x ← x＋1 を実行し，
　　そうでなければ x ← x－1 を実行する

条件分岐の中で複数の条件で実行する処理を切り替えたい場合は，次のように『ならば』と『そうでなければ』の間に『そうでなくもし』を使って条件を追加します。

《一般形》

もし **〈条件 1〉** ならば
　〈処理 1〉
を実行し，そうでなくもし **〈条件 2〉** ならば
　〈処理 2〉
を実行し，そうでなければ
　〈処理 3〉
を実行する

例:　もし x＝3 ならば
　　x ← x＋1
　を実行し，そうでなくもし y＞2 ならば
　　y ← y＋1
　を実行し，そうでなければ
　　y ← y－1
　を実行する

改行位置によって実行結果が変わらないため，各処理が 1 行で書ける場合には，次のように書くこともあります。

《一般形》

もし **〈条件 1〉** ならば **〈処理 1〉** を実行し，
そうでなくもし **〈条件 2〉** ならば **〈処理 2〉** を実行し，
そうでなければ **〈処理 3〉** を実行する

例:　もし x＝3 ならば x ← x＋1 を実行し，
　　そうでなくもし y＞2 ならば y ← y＋1 を実行し，
　　そうでなければ y ← y－1 を実行する

5.2　条件繰返し文

条件繰返し文には，「前判定」と「後判定」の 2 種類があります。

5.2.1　前判定

〈条件〉 が成り立つ間，**〈処理〉** を繰り返し実行します。
〈処理〉 を実行する前に **〈条件〉** が成り立つかどうか判定されるため，**〈処理〉** が 1 回も実行されないことがあります。

《一般形》

〈条件〉 の間，
　〈処理〉
を繰り返す

例:　x＜10 の間，
　　gokei ← gokei＋x
　　x ← x＋1
　を繰り返す

5.2.2　後判定

〈条件〉 が成り立つまで，**〈処理〉** を繰り返し実行します。
〈処理〉 を実行した後に **〈条件〉** が成り立つかどうか判定されるため，**〈処理〉** は少なくとも 1 回は実行されます。

《一般形》

繰り返し，
　〈処理〉
を，**〈条件〉** になるまで実行する

例:　繰り返し，
　　gokei ← gokei＋ x
　　x ← x＋ 1
　を，x ≧ 10 になるまで実行する

5.3　順次繰返し文

順次繰返し文は，**〈変数〉** の値を増やしながら，**〈処理〉** を繰返し実行します。

《一般形》

〈変数〉 を **〈初期値〉** から **〈終了値〉** まで **〈差分〉** ずつ増やしながら，
　〈処理〉
を繰り返す

順次繰り返し文は，以下の手順で実行されます。

1. **〈変数〉** に **〈初期値〉** が代入されます。
2. **〈変数〉** の値が **〈終了値〉** よりも大きければ，繰り返しを終了します。
3. **〈処理〉** を実行し，**〈変数〉** の値に **〈差分〉** を加え，手順 2 に戻ります。

例:　x を 1 から 10 まで 1 ずつ増やしながら，
　　gokei ← gokei＋ x
　を繰り返す

『増やしながら』を『減らしながら』にすると，**〈変数〉** の値を **〈初期値〉** から **〈差分〉** ずつ減らしながら，その値が **〈終了値〉** よりも小さくなるまで，**〈処理〉** を繰り返し実行します。

例:　x を 10 から 1 まで 1 ずつ減らしながら，
　　gokei ← gokei＋ x
　を繰り返す

6　関数

関数には値を返すものと値を返さないものがあります。関数の動作は，問題文の中で定義されます。

6.1　値を返す関数

問題文の中で

- 指定された値の二乗の値を返す関数「二乗」を用意する
- 値 m の n 乗の値を返す関数「べき乗 (m, n)」を用意する
- 値 m 以上値 n 以下の整数をランダムに一つ返す関数「乱数 (m, n)」を用意する
- 値 n が奇数のとき真を返し，そうでないとき偽を返す関数「奇数 (n)」を用意する

のように定義された関数を，表示文 (2 節)，代入文 (3 節)，算術演算（4.1 節），比較演算（4.2 節），あるいは論理演算（4.3 節）の中で使うことができます。関数を呼び出すときは，関数名に続き，『(』と『)』の間にパラメータ（引数）を書きます。複数のパラメータを指定する場合は，『, 』で区切ります。

例:　y ← 二乗 (x)　　(y に x の二乗が代入されます。)
例:　z ← 二乗 (x)＋べき乗 (x, y)　　(z に x の二乗と x の y 乗の和が代入されます。)
例:　r ← 乱数 (1, 6)　　(r に 1 から 6 までの整数のうちいずれかが代入されます。)

6.2　値を返さない関数

問題文の中で

- 指定された値を 2 進表現で表示する関数「二進で表示する」を用意する

のように値を返さない関数が定義されることがあります。

例:　二進で表示する (x)

ご視聴ありがとうございました。