§29. プログラミングとフローチャート(いかなる言語にも共通する基本)

近年、『プログラミング』という単語をよく見たり聞いたりするようになりました。子どもを対象としたロボットの工作や科学のイベントで、またコンピュータを仕事で用いる大人の様々な分野で。

本題に入る前に、プログラミングの意味について簡単に触れておきましょう。プログラミングというのはプログラムを作成することという意味合いですが、それではそもそもプログラムとは何でしょう。コンピュータに書き込むことによって、そのコンピュータを思い通りに動かす命令書のようなものです。分かりやすい例で説明します。どの家庭にもある洗濯機ですが、スイッチを入れ洗い物の分量や時間をセットしてスタートさせれば、洗い、濯ぎ、脱水等をすべて自動でこなします。これはマイクロコンピュータ(略してマイコン)という小さなコンピュータが内蔵されていて、そこに一連の作業を実行する命令が書き込まれています。この命令書がプログラムです。PC(パーソナルコンピュータ)は勿論、スマホや携帯も、身近な電気・電子製品(マイコンを搭載した)も、ロボットも、すべて本体だけだったらただの置物に過ぎません。内蔵されているコンピュータにプログラムを書き込むことによって、初めて意味のある存在になります。

それではプログラムはどのように書くのか。我々人間は言葉を交わしたり文を読み書きしたりできますが、それが可能なのは言うまでもなく共通で理解している言語(日本の場合は日本語)のおかげです。実はコンピュータもある言語を理解できるように作られています。したがってコンピュータが理解できる命令で次々と指示を送れば、こちらの意図したとおりのことを実行してくれます。ところがコンピュータが理解できる言語は機械語(マシーン語)と言って、十六進数の 16 個の数字(0～9、A、B、C、D、E、F)で表される命令からなります。したがって人間には非常に分かりにくく、こちらの意図することを効率よくコンピュータに伝えるのは無理があります(PC が世に出回り始めたころ、機械語でゲームを組むつわものが結構いた)。それで、人間とコンピュータとの間の橋渡しをする言語が開発されました。今から紹介する言語はいずれもそうですが、人間がある言語を用いてプログラムを作成すると、その言語が持ち備えた翻訳ソフトで機械語に(即ちコンピュータが理解できる言語に)書き換えてくれるのです

一番機械語に近い言語はアセンブリ言語(アセンブラと言われることが多い)です。簡単な構造のロボットや、趣味でマイコンを活用する人の中には今でもアセンブリ言語を使っている人はいます(私もその一人)。命令語の数が少なく、したがって言語そのものを習得するのは楽ですが、限られた命令語で意図することを書き表すのには少し苦労します。

人間にとって一番分かりやすく作られているのが BASIC 言語です。初心者にもなじみやすく、この言語が出発点として用いられることが多いです。BASIC でプログラミングの基本を学び、その後必要に応じて他の言語に移るという人が多いです。BASIC は言語として分かりやすいという長所がある反面、実際にソフトとして動かすときのスピードが遅いという短所があります(機械語に翻訳するのに時間がかかるため)。

いろいろな分野で一番使われ昔から有名なのは C 言語です。C 言語は人間にも分かりやすく(ただし、非常に奥が深く、高度な内容まで理解するのは簡単ではない)、しかもス

ピードが速いです。ただし進化・発展が目覚ましい世界なので、現在では様々な言語が開発されていて（よく耳にするのは JAVA とか PYTHON など）どれが一番とは言い難いかも知れません。それぞれ科学技術とか NET 関連とか事務処理とか得意とする分野があって、必要に応じて活用されているようです。

さて、前置きが大変長くなってしまいましたが、ここからは本題のプログラミングおいて基本となることについて説明します。取り上げるのは「繰り返し」と「分岐」の二つです。その意味と書き方の要領を心得ていれば、いかなる言語を用いた場合でも同じようにプログラムとして書けるようになるものです。

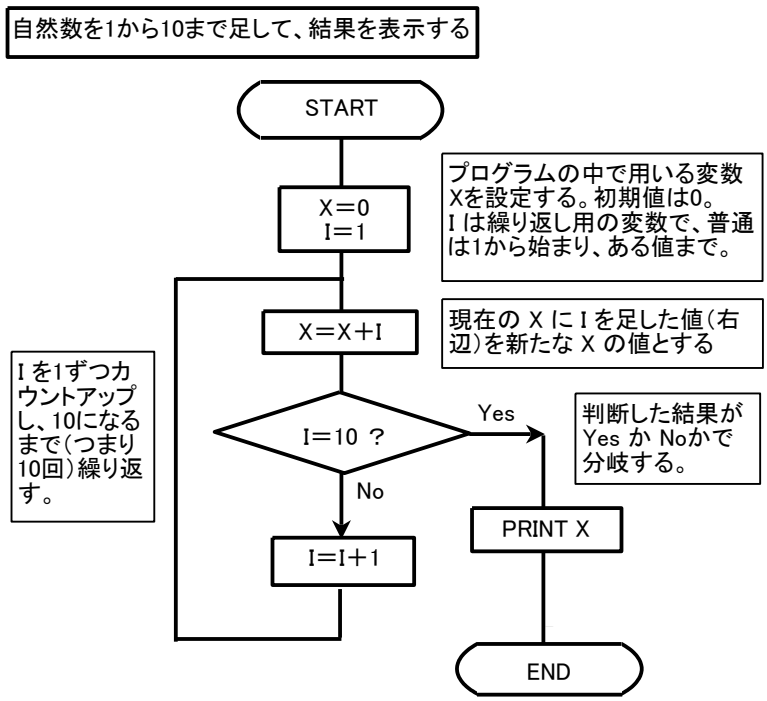
＜繰り返し＞

あること、たとえばモーターを 1 秒間回転させるとか、LED を 0.5 秒間点灯させるというようなことを単に 10 回繰り返せば、それぞれ 10 秒間の回転、5 秒間の点灯ということになります。したがって、1 秒間の回転、0.5 秒間の点灯というような基本となる動作をプログラムとして書くことができれば、任意の時間モーターを回転させたり LED を点灯させたりすることができるわけです。回数をカウントする変数、たとえば N を用意して、一回動作を実行するごとに N を 1 カウントアップし、目的の回数になったら終わるようにします。

＜分岐＞

ある変数、たとえば X があらかじめ定めた値、たとえば A に対して、X≧A ならある動作をさせ、X＜A ならまた別の動作をさせるというのが分岐です。

具体的な例で見ていくことにしますが、言語で書き表す前に右のようなフローチャート（流れ図）で表すのが分かりやすく、非常に効果的です。フローチャートを描くときの大まかな約束は次のとおりです。



プログラムの最初は（START）、終わりは（END）です。一つひとつの実行は長方形の枠で表します。

なおプログラムの中で使われる等式は、通常使われる「＝」と意味合いが異なる場合があります。図の例にある「X＝X＋I」は、右辺の値（現在の X に I を足した値）を左辺に代入する（新たな X の値とする）という意味です。従って、分かりやす

く「X←X＋I」のように書くこともあります。

分岐はひし形を用いて、その中の判断が Yes か No かで矢印の方向へ分岐します。図のフローチャートは「自然数を 1 から 10 まで足し算して結果を表示する」というものです。

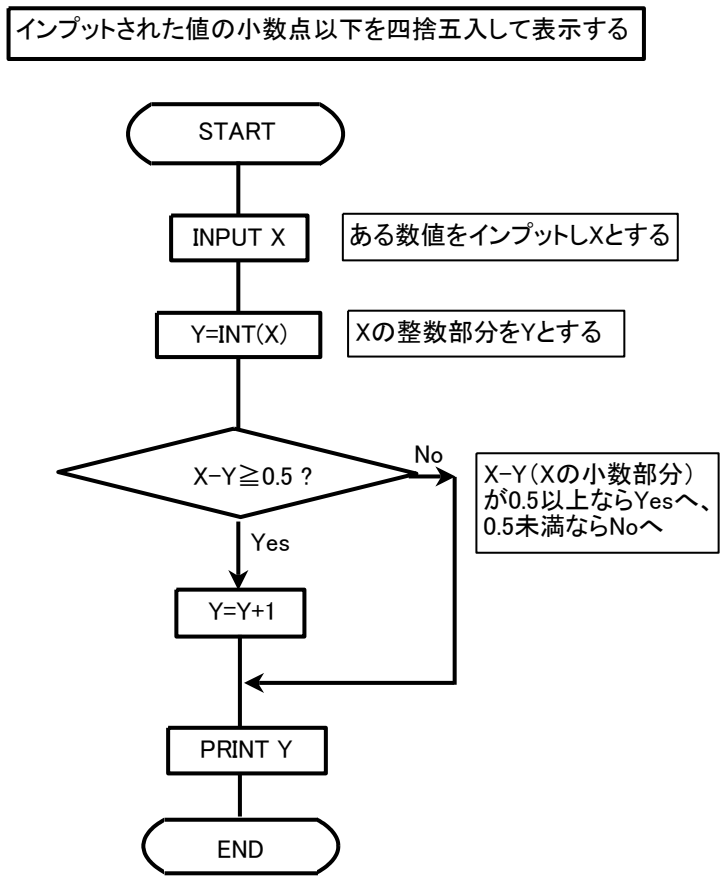
これを BASIC で書くと次のようになります。繰り返しと分岐がどのように行われているかを確認してください。ただし、ここでの分岐は「FOR～NEXT」という BASIC 特有の命令文に組み込まれています。一番左の行番号（10、・・・、50）はその行の番地のようなものです（次の例で明らかになります）。「;」 の後は説明のためのコメントでプログラムそのものとは無関係であり無視されます。

```
10   X=0                    ; 変数として X を用いる。初期値として X＝0 とする
20   FOR I=1 TO 10          ; 変数 I が 1 から初めて 10 になるまで次のことを行う
30        N=N＋I            ; 現在の X の値に I を足した値を新たな X の値とする
40   NEXT I                 ; I の値を 1 プラスし、I≦10 なら 30 行へ戻り、I＞10
                            ; なら繰り返しを終えて次の行へ
50   PRINT "N=";N           ; 結果の『 N＝55 』を表示する
```

以下、いくつかの例をフローチャートとプログラムで示します。

右の例は、あるインプットされた数値の小数点以下を四捨五入して表示するものです。BASIC には整数部分だけを取り出す命令「INT（X）」（整数という意味の Integer より）があるので、これを用いると簡単です。

プログラムは次のようになります。



```
10  INPUT  X                ; X＝34.56 のとき
20  Y=INT                   ; Y＝34
30  IF X－Y≧0.5             ; X－Y＝0.56≧0.5 で Yes なので、THEN 以下
        THEN Y=Y+1          ; を実行して、Y=35（No ならこの行を無視、Y=34）
40  PRINT  Y                ; Y を表示
```

次の例は 100 個のデータ（正の整数値）を大きい順に並べ替える（ソーティングと言う）プログラムです。フローチャートは次のページの図の通りです。100 個のデータは A(I)（I：0～99）に入っているものとします。A(0)＝123、A(1)＝98、A(2)＝4、・・・、A(99)＝56 といった具合です。同じ数値のものがあってもかまいません。このように BASIC では、関数記号 f(x) 似た表示法による変数付き文字で数多くのデータを表すことが出来ます。これを「配列」と言いますが、プログラミングにおいて非常に助かる命令です。

```
10  FOR I=0 TO 98
20      FOR K=I+1 TO 99
30          IF  A(K)＞A(I)  THEN  SWAP  A(K)、A(I)
40      NEXT K
50      PRINT A(I);
60  NEXT I
70  PRINT A(99)
```

「FOR～NEXT」が二重の構造になっています。10 と 60 の間で I に関して 99 回の繰り返しがあり、そのそれぞれにおいて、20 から 50 までの 99－I 回の繰り返しがあります。最初の方を順を追って見ていくと、まず I＝0 で K＝1 ですから、A(1)と A(0)の比較で、もしも A(1)＞A(0) なら SWAP します（値を入れ替える）。つまり A(0) に大きい方の値が入ります。以下その A(0) と A(2)、A(3)、・・・、A(99)を比較して大きい方を A(0) に入れるを繰り返します。結果として一番大きな値が A(0) に入ります（40 まで）。50 でその A(0)の値を表示します。10 に戻って（I＝1）、A(1) と A(2) ～A(99) まで今と同じことを繰返します。結果として A(1) に二番目に大きな入り表示されます。これを I＝98 まで行うと、一番目から 99 番目（A(98) ）まで大きい順に表示されたことになり、最後に 100 番目の A(99)を表示して終了です。

いずれもフローチャートがプログラムの流れを見る上で非常に有効であることが分かると思います。プログラムが複雑になってくると他人が書いたものを解読するのは非常に困難なことになりますが、フローチャートならだれが書いたものを見ても分かり易いです。目的とするプログラムを作成する際、きちんとしたフローチャートを完成させることが出来れば、（そして用いている言語を思うように使いこなす能力を持っていれば）全工程の半分は終わったと考えてよいでしょう。

100個の整数のデータを大きい順に並べ替える（ソーティング）