

2013 石鎚プロジェクト 課題 5-3

NE23-0142F 鈴木海斗

課題 5-3：教材作り

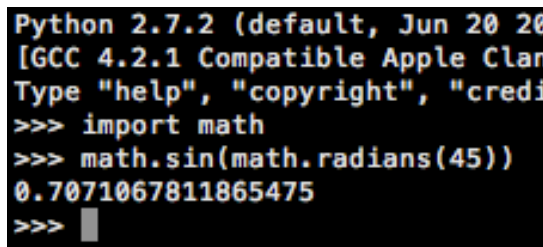
作ったもの：アルゴリズムの説明

概要

ネットワーク情報学部に入学した新入生のうち、プログラミングの経験がない人達向けのアルゴリズムの説明を目標。フローチャートの説明をしたのち、バブルソート、選択ソートなどのソートを説明する。

プログラミング言語とは

プログラミング言語とは、簡単に言えばコンピュータに何らかの処理をさせるための人によって定義された言語の事です（例えば、コンピュータに値を計算させたりする）。例えば下図のような処理ができます。



```
Python 2.7.2 (default, Jun 20 2012)
[GCC 4.2.1 Compatible Apple Clang 4.0
Type "help", "copyright", "credits()"
>>> import math
>>> math.sin(math.radians(45))
0.7071067811865475
>>>
```

図 1-1.値の計算

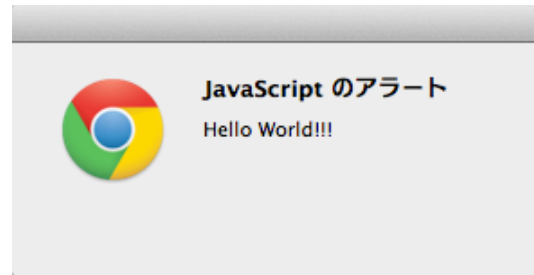


図 1-2.ブラウザに文字を表示

図 1-1 は、コマンドラインを使用して三角関数の値をコンピュータに表示させています。少し見辛いですが「>>>math.sin~~~」という行の部分が命令で、その下の行の数値がその命令に対するコンピュータが返した答えです。

図 1-2 はブラウザに「Hello World!!!」という文字列をアラートで表示させる、という命令です。

この二つの命令、それぞれ違うプログラミング言語を使用して処理していますが、書き方が違うだけでどの言語でも似たような処理は可能です。つまり、「〇〇を計算せよ」、「〇〇を表示せよ」といった命令をその言語使用毎に適した書き方をしているだけで、コマンドラインで「Hello World!!!」という文章を表示させる事も、ブラウザで $\sin 45^\circ$ の計算結果を表示する事も可能です。これらの処理を順にして組み合わせたものをアルゴリズム、その流れを図に示したものをフローチャートと言います。

例えば図 1-1 の場合は簡単に言うと「 $\sin 45^\circ$ を計算」→「計算結果を画面に出力」といったアルゴリズムになっています。このアルゴリズムはどの言語でも一緒なので、アルゴリズムをしっかり理解しておくことが重要になります。

フローチャート

プログラミングを始めるにあたり、どんな処理をしたいのかをまずは明らかにしておく必要があります。ここでは「1~100 までの整数を画面に出力する」という処理をしたい、ということにしておきます。この処理はプログラミングに慣れている人ならパッと思いつくでしょうが、初めての人にはちょっと戸惑うかもしれません。まずは簡単なフローチャートから見ていきましょう。

図2-1は図1-2の処理をフローチャートで表したものです。一番上から始まり矢印の方向に沿って処理をしていきます。

長方形はデータの計算や代入など、平行四辺形は入出力の処理をする場合に使います。ここで str という見慣れない文字が出てきました。これは変数と呼ばれるものですが、詳しい説明は次章で行います。今は str を出力すると「Hello World!!!」という文字列が表示されることだけ覚えておいてください。

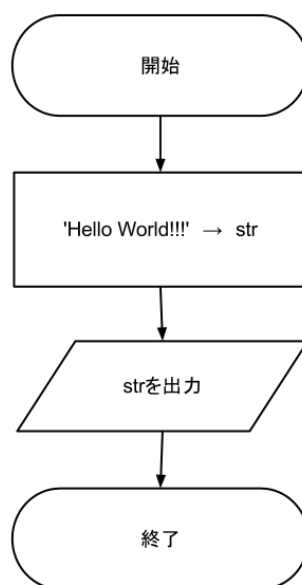


図 2-1.フローチャート

変数の概念について

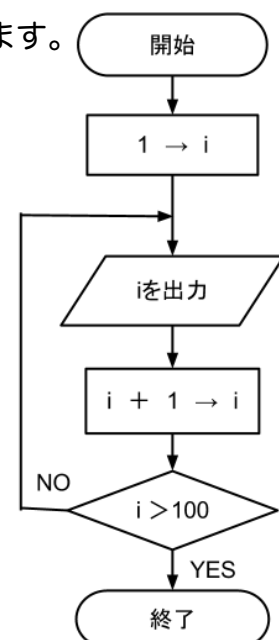
先ほどでも登場しましたが、プログラムでは変数という便利なものを使えます。変数とは一時的にデータを記憶して、後で利用できるようにするために名前を付けたものです。「名前のついた箱に値を格納する」というふうに覚えておくの良いかもしれません。変数には言語によって命名規則がありますが、大抵は英数字を組み合わせた文字列なら OK です。

図 2-1 にあった「'HelloWorld!!!' → str」という命令、この場合は str という変数に「HelloWorld!!!」という文字列を代入しろ、という命令になります。この命令により、プログラム上では str と言ったら「Hello World!!!」という文字列のことになります。

そのため図 2-1 にある次の「str を出力」という命令で str の中身である「Hello…」という文字列が出力されます（変数を使わなくても直接文字列を出力する事も可能です）。

図 2-1 のフローチャートだと変数を使う意味はほとんどありませんが、例えば、「1~100 までの整数を画面に出力する」という処理をコンピュータにやらせたいときは変数を使うとより簡単に実現できます。この処理は「1 を出力」「2 を出力」...「100 を出力」というように変数を使わなくてもできますが、変数を使った方がもっと簡単に、見やすくすることができます。

この処理をフローチャートにすると右の図のようになります。変数 i に 1 を代入して出力した後、 i に 1 を加えてまた出力...といった処理を i が 101 になるまで繰り返す、という内容になっています。



ここで、いくつか新しいものが出てきました。まず「 $i+1 \rightarrow i$ 」という処理。これは i に 1 を足したものを i に代入、という意味の命令です。

たとえば、この処理をする時の i の値が 10 の時は「 $10+1 \rightarrow i$ 」で i は 11 になります。「現在の i の値に 1 を加える」という感じです。

ちょっと考え方が難しいですが、プログラムで処理の回数を数えたりするときに、よく使うので覚えておくべきです。ちなみにこの処理（変数の値を 1 ずつ増やす処理）をインクリメントと呼びます。

次に、フローチャートの菱形の部品。これは条件分岐と呼ばれるもので、条件を満たした場合、満たさなかった場合の処理を分ける事ができます。今回の条件は i が 100 より大きいかどうか、で判断しています。100 まで出力した後、 i に 1 が加えられて 101 になると、その先の数は出力する必要はないので、条件を満たした場合は終了します。

まとめ

- ・ 「 $i+1 \rightarrow i$ 」は変数の値を 1 ずつ増やす処理のこと。ループ回数のカウントなど様々な場面で使われている。
- ・ フローチャートにおける菱形の部品は「条件分岐」を意味する。

配列の概念

配列は添字を使ってデータを区別する事のできる、データの集まりのことです。例えば配列が `array[5]` と定義されてある場合、多くの言語では `array[0]` から `array[4]` までを変数と同じように使用する事ができます。配列を使いこなせば変数をむやみにたくさん宣言する必要もなくなります。

array	80	75	99	55	0
	[0]	[1]	[2]	[3]	[4]

図 4-1.配列の例

※ これより先、未完成

※ 追記

今回目標とする部分まで教材を作成することができなかったため、逐一更新していく予定です。下記の URL で常に最新版が見られます。

<https://github.com/ne230142/project>