Scratchによる
アルゴリズム入門

第2回　整列（ソート）

2017.03.30

鎌倉シチズンネット（KCN）

整列(1)

- アルゴリズム入門 > 6章　検索・ソート　> 6-6 ソートとは

データを昇順（小さい順）あるいは降順（大きい順）に並べ替えることを、整列あるいはソートと呼ぶ。

元のデータ	84 121 43 93 140 83 14 93 181 58
昇順にソート	14 43 58 83 84 93 93 121 140 181
降順にソート	181 140 121 93 93 84 83 58 43 14

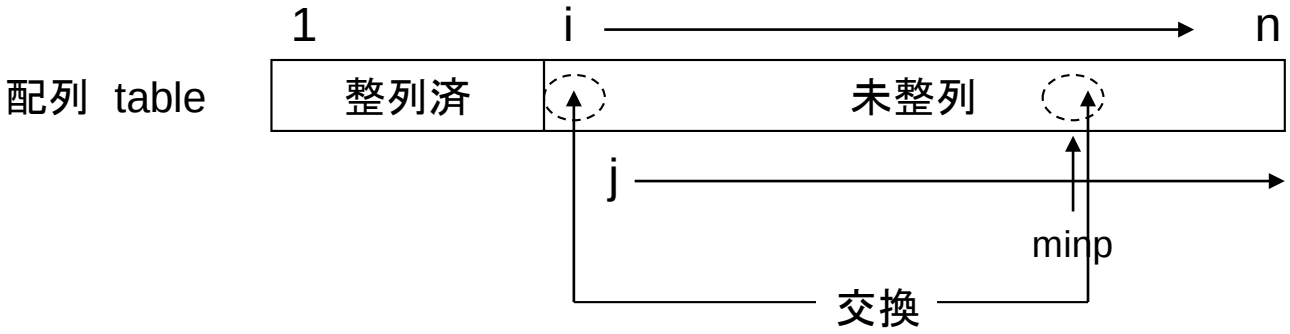
ここでは以下の整列アルゴリズムについて学ぶ。

- 選択法
- バブルソート
- クイックソート

整列(2)

- アルゴリズム入門 > 6章　検索・ソート　> 6-8 選択法

選択法(Selection Sort)と呼ばれるソートアルゴリズムで、以下のように行う。



未整列の要素の中で最小のものを見つけ、未整列の先頭の要素と最小の要素を交換する（上図の ◌ ）。これを未整列の要素が1つになるまで繰り返す。

これをScratchで記述すると次のようになる。

整列(3)

- アルゴリズム入門 > 6章　検索・ソート　> 6-8 選択法

選択法のプロジェクトはこちらです。
https://scratch.mit.edu/projects/153152455/

整列(4)

- アルゴリズム入門 > 6章　検索・ソート　> 6-8 選択法

[変数の意味]

table：整列対象の配列（リスト）
n　　：配列の要素数（リストの長さ）
i　　：未整列の先頭位置
j　　：未整列の要素の現在位置
minp：未整列の中で最小の要素の位置
w　　：配列要素を交換するときの作業用の変数

[練習問題]

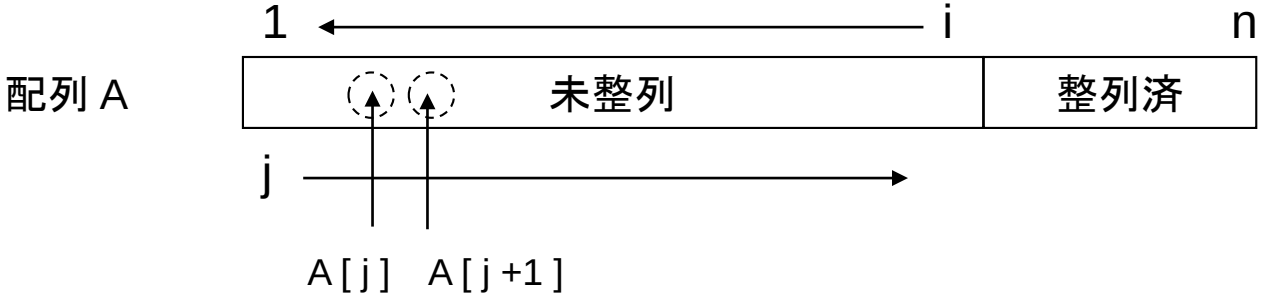
前頁の選択法を次のようにリミックスします。
配列を降順（大きい順）に整列するように変更し、変数名は minp → maxp に変更する。

実行例：181 140 121 93 93 84 83 58 43 14

整列(5)

- アルゴリズム入門 > 6章　検索・ソート　> 6-9 バブルソート（交換法）

バブルソートは配列の隣同士を比較し、左側　＞　右側　なら左右を交換する処理を未整列部分の最後まで行うと、最大の要素が未整列部分の最後にくる。これを未整列部分がなくなるまで繰り返す。降順にソートする場合には、左側　＜　右側　なら左右を交換すればよい。

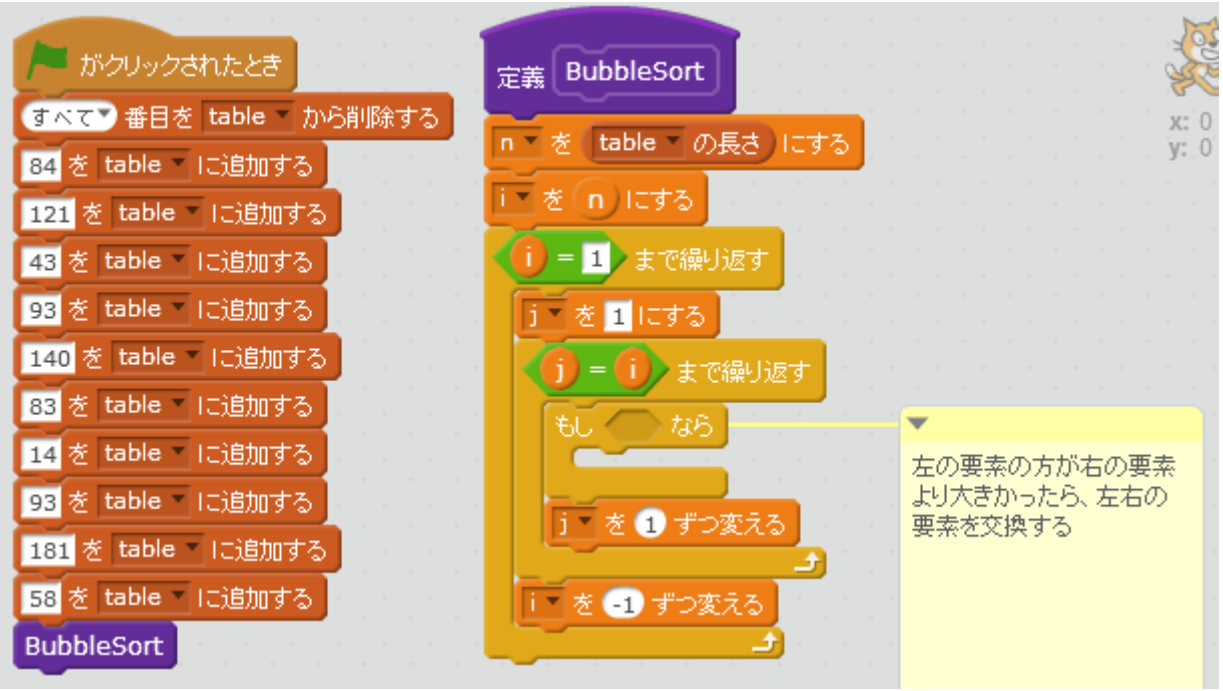


A [j] > A [j + 1] ならA [j] とA [j + 1] を交換する。（昇順にするとき）

A [j] < A [j + 1] ならA [j] とA [j + 1] を交換する。（降順にするとき）

整列(6)

- アルゴリズム入門 > 6章　検索・ソート　> 6-9 バブルソート（交換法）



下記のプロジェクトをリミックスして、バブルソートの処理を完成してください。
https://scratch.mit.edu/projects/153182359/

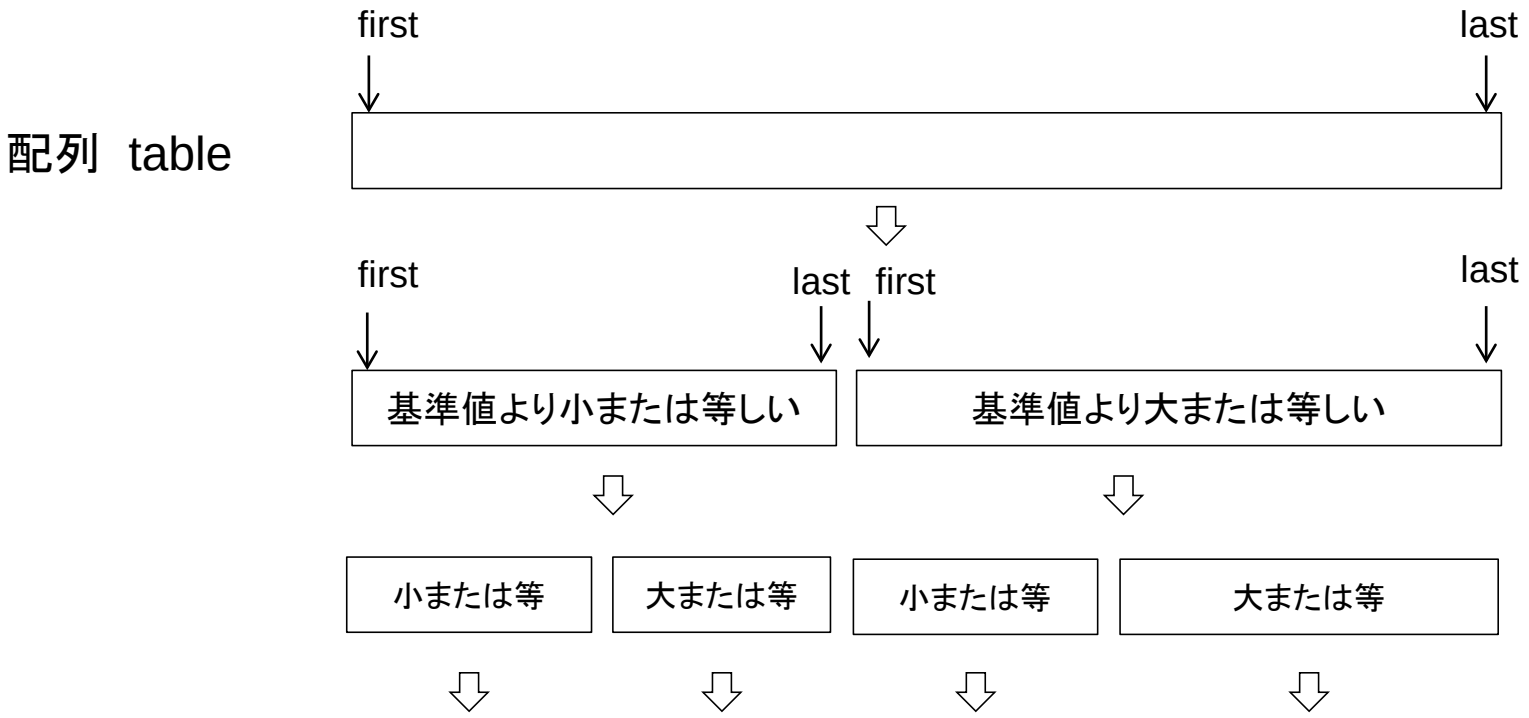
整列(7)

- アルゴリズム入門 > 6章　検索・ソート　> 6-9 バブルソート（交換法）

[変数の意味]
table：整列対象の配列（リスト）
n　　：配列の要素数（リストの長さ）
i　　：未整列の最後（右端）の位置
j　　：未整列の要素の現在位置
w　　：配列要素を交換するときの作業用の変数

整列(8)

- アルゴリズム入門 > 6章　検索・ソート　> 6-11　クイックソート



ここでは配列の先頭要素を基準値（ピボット）とする。

整列(9)

- アルゴリズム入門 > 6章　検索・ソート　> 6-11　クイックソート

1. 配列の先頭要素をピボット（基準値）とする
2. 配列要素をピボットよりも小さいもの、ピボットと等しいもの、ピボットより大きいもの、に並び替える
   1) 配列要素の最初から、ピボットよりも大きいか等しいものを探す
   2) 配列要素の最後から、ピボットよりも小さいか等しいものを探す
   3) ピボットよりも小さいものが、ピボットよりも大きいものよりも後にあれば、ソート終了
   4) そうでなければ、その2つの要素を交換する
   5) この処理を、その続きの範囲から繰り返す
3. ピボットよりも左側と、右側の領域で、この処理を再帰的に行う。

整列(10)

- アルゴリズム入門 > 6章　検索・ソート　> 6-11　クイックソート

☆ i , j が変化していく様子（ 1回目）　基準値（ピボット）＝84

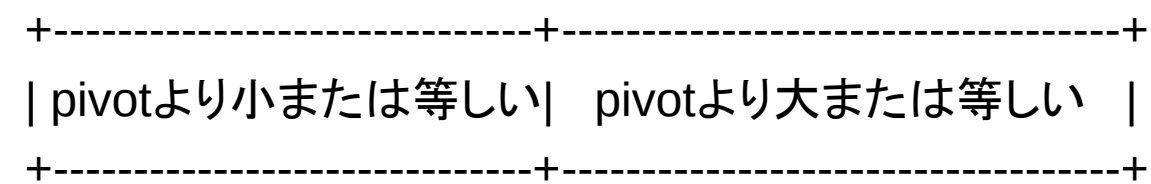
```
84  121   43   93  140   83   14   93  181   51
 i                                            j
51  121   43   93  140   83   14   93  181   84
      i                       j
51   14   43   93  140   83  121   93  181   84
               i          j
51   14   43   83  140   93  121   93  181   84
               j    i
```

整列(11)

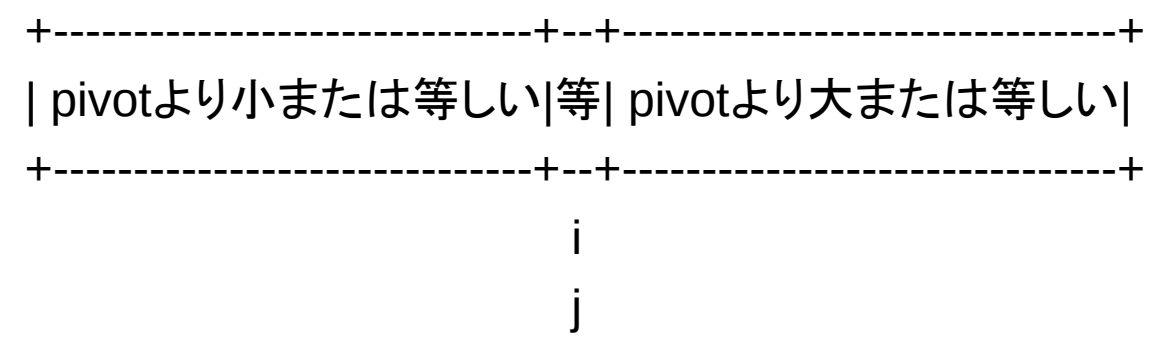
- アルゴリズム入門 > 6章　検索・ソート　> 6-11　クイックソート

☆ 交換終了時の状態

ケース1



ケース2



整列(12)

- アルゴリズム入門 > 6章　検索・ソート　> 6-12　処理速度の比較

データ件数 N を大きくしていったとき、アルゴリズムの計算量がどのように増加するかを「O 記法」（オー記法）で表現する。

選択法、バブルソート、挿入法　→　$O(N^2)$

クイックソート　→　$O(N \times \log_2 N)$

整列(13)

- アルゴリズム入門 > 6章　検索・ソート　> 6-11　クイックソート

右図のスクリプトは QuickSort を再帰的に呼び出す処理が抜けている(P.10の3)。下記のプロジェクトをリミックスして、完成させてください。
https://scratch.mit.edu/projects/153228463/

整列(14)

- 解答例

選択法
https://scratch.mit.edu/projects/153231933/

バブルソート
https://scratch.mit.edu/projects/153174798/

クイックソート
https://scratch.mit.edu/projects/153194866/