

プログラミング技法

第4回演習

資料は

http://www-hiraki.is.s.u-tokyo.ac.jp/lectures/prog_giho/

今日の目標

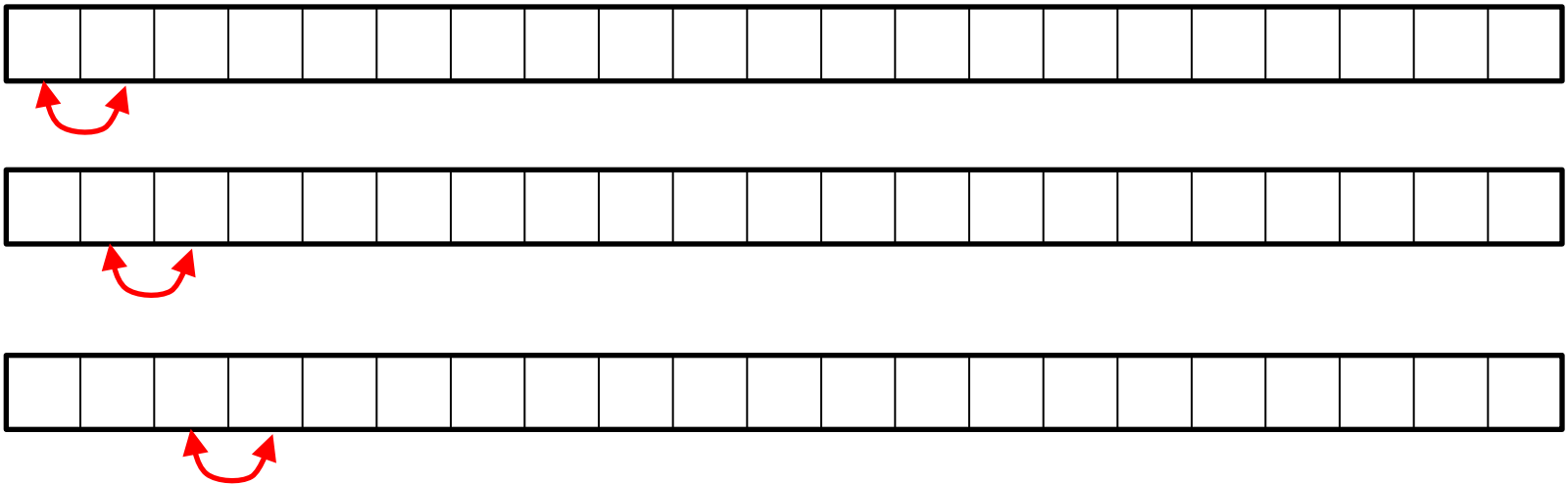
- 新しい課題に挑戦—— Sorting
 - データを1次元の大小順に並べ直す
- アルゴリズムの大事さ/実装の速さの関係
- 3言語全部でSORT問題を書く(Optional)

一番簡単なSortingを演習ページに置いた

何故実行時間が変わるか

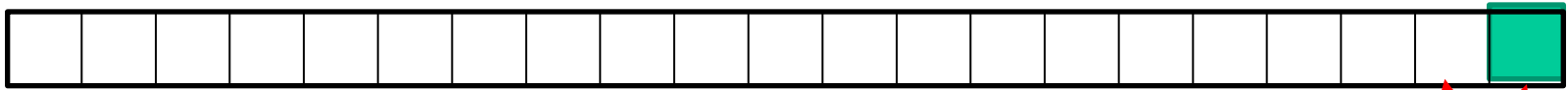
- 比較・データ移動の回数が、データの数とどのような関係か？

– 例のプログラム バブルソート $O(n^2)$



バブルソート 2

最大値



バブルソートの自分での実装

- 自分でバブルソートを書いてみる
- 例プログラムと速度比較
- データサイズ 100から10000位まで5点くらい

アルゴリズムを変える

- バブルソートは計算回数が $O(n^2)$ の遅いアルゴリズム
- 良く知られている速いアルゴリズムは
 - Quick Sort
 - Merge Sort
 - Radix Sort などいろいろある。
- 本やWebページで探して、実装して速度比較

ポイント

- 計算量のオーダー(例えば $O(\log n)$)が小さいアルゴリズムが常に速いわけではない
- JavaのJIT(Just In Time)コンパイラの効き目は大きい
- なぜかMacのRubyは遅い

実行結果例 (on Mac)

Mac環境 = Xeon W3520, OSX 10.10.5, Apple LLVM 7.0.0 (clang-700.0.72) -Ofast, JDK 1.6.0_65 (Server VM), ruby 2.0.0p481

C

500,	0.000184,	0.000184,	0.000183,	0.000184,
2000,	0.002883,	0.002933,	0.002883,	0.002919,
4000,	0.011710,	0.011664,	0.011635,	0.011806,
6000,	0.025798,	0.025814,	0.026381,	0.025532,
8000,	0.046585,	0.045872,	0.044729,	0.045478,
10000,	0.071115,	0.071065,	0.069996,	0.071442,

Java

500,	0.003559,	4.5E-4,	2.03E-4,	1.97E-4,
2000,	0.002943,	0.002998,	0.00296,	0.002939,
4000,	0.011844,	0.011839,	0.011813,	0.011724,
6000,	0.026704,	0.026679,	0.026775,	0.026665,
8000,	0.047838,	0.047576,	0.047828,	0.04762,
10000,	0.075231,	0.074106,	0.074147,	0.074319,

Ruby

500,	0.041676,	0.041382,	0.041463,	0.041565,
2000,	0.653054,	0.651173,	0.651091,	0.652389,
4000,	2.608098,	2.603893,	2.649164,	2.641159,
6000,	5.869905,	5.920909,	5.938819,	5.904991,
8000,	10.525021,	10.507519,	10.531742,	10.473472,
10000,	16.40252,	16.862218,	16.603004,	16.376962,

実行結果例 (on Linux)

Linux環境 = Core i7-920, CentOS 7.1, GCC 4.8.3 -O3, JDK 1.8.0_60, Ruby 2.0.0p598

C

500,	0.000185,	0.000190,	0.000185,	0.000185,
2000,	0.002933,	0.002934,	0.002937,	0.002937,
4000,	0.011639,	0.011679,	0.011641,	0.011637,
6000,	0.026111,	0.026138,	0.026152,	0.026173,
8000,	0.046522,	0.046382,	0.046380,	0.046361,
10000,	0.072390,	0.072400,	0.072404,	0.072378,

Java

500,	0.005704471,	0.001819815,	5.43581E-4,	2.07921E-4,
2000,	0.003087148,	0.003086791,	0.00295379,	0.003080792,
4000,	0.011716112,	0.011834578,	0.011823093,	0.011849949,
6000,	0.026388724,	0.02649999,	0.026641345,	0.026638378,
8000,	0.047312579,	0.047174071,	0.047170846,	0.047277527,
10000,	0.073556909,	0.073694132,	0.074268603,	0.074842909,

Ruby

500,	0.050006323,	0.042032965,	0.037384864,	0.033574135,
2000,	0.495958805,	0.49685297,	0.496542461,	0.496584893,
4000,	1.984901537,	1.984239613,	1.98564696,	1.983702763,
6000,	4.469238888,	4.473405921,	4.510673699,	4.468921489,
8000,	7.941280614,	7.940602592,	7.9413131,	7.946932947,
10000,	12.40263134,	12.37210187,	12.36273734,	12.379538527,