3 アルゴリズム（5）〜ソーティングアルゴリズムの比較〜

【準備)】 あらかじめこちらで用意した関数が定義されているファイルをコピーして下さい。

```
$ cp /usr/local/common/kogai/ouyou/util.c util.c
```

【課題 3-1】 前回までに作成した、**バブルソートでソーティングする関数 bubble() を用いて**、70,000 要素の配列にランダムに設定された値をソートするプログラムを作成して下さい。

この課題の main() は、例えば、次のように、配列への値の設定、配列のソートティング、配列の表示を行います。ここで使われている関数 set() は予め準備してあります。

```
/* バブルソートの場合 */
int main()
{
  int array[70000];

  /* 配列にランダムな値を設定する */
  set(array, 70000);
  /* ソート前の配列を表示する */
  printf("-----ソート前-----\n");
  show(array, 70000);
  printf("\n");
  /* バブルソートをする */
  bubble(array, 70000);
  /* ソート後の配列を表示する */
  printf("-----ソート後-----\n");
  show(array, 70000);

  return 0;
}
```

【課題 3-2】 前回までに作成した、**クイックソートでソーティングする関数 quick() を用いて**、70,000 要素の配列にランダムに設定された値をソートするプログラムを作成して下さい。

この課題の main() は、課題 3-1 と同様に、前回の課題で作成したプログラムに追加して作ります。

小テスト

提出する全てのファイルの先頭行に、コメントとして自分の番号と名前を書いてください。

【問 1】 70,000 個の要素の int 型配列に対して、バブルソートとクイックソートをするプログラムをそれぞれ作成し、**これらのプログラムと、実行時間を計測した結果を提出して下さい。**（20 点）

3.1 実行時間の測り方

1. main() 内の画面に出力する処理の部分は、コメントにしておきます。(より正確な実行時間を計るためと膨大なデータが画面を埋め尽くしてしまうため)
2. ソートプログラムの実行時間を計ります。以下のように、「time」コマンドを使うと「./a.out」を実行するのにかかった時間を表示します。

```
$ time ./a.out
・・・(実行結果表示)・・・
real    「プログラムの実行時間（主にソートと表示部分)」
user    「CPU を独占した時間（主にソート部分)」
sys     「システムを利用した時間」
```

3. 手順 5 の実行結果をファイルに書き出します。今回は「script」コマンドを以下のように用います。

```
$ script 「実行結果ファイル名」
$ time ./a.out
・・・(time コマンドの実行結果表示)・・・


$ exit
```

答案の提出

- 保存したファイルは次のように「report」コマンドで提出する
  (ちゃんと提出できた場合は、「Succeed.」と画面に表示される)
  `$ ~kogai/report ouyou3 「プログラムファイル」`
- **今回は以下の 4 つのファイルを提出する**
  提出する全てのファイルの先頭行に、コメントとして自分の番号と名前を書いてください。

```
$ ~kogai/report ouyou3 「70000 要素のバブルソートのプログラム」
$ ~kogai/report ouyou3 「バブルソートの実行結果のファイル」
$ ~kogai/report ouyou3 「70000 要素のクイックソートのプログラム」
$ ~kogai/report ouyou3 「クイックソートの実行結果のファイル」
```

- 同じ問題に対して、複数の提出ファイルが存在した場合は、更新日時が新しい方を提出ファイルとする