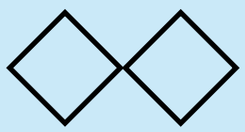
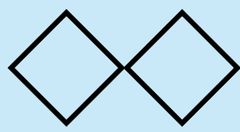
本日の課題

以下に示すソートアルゴリズムはcomb sortと呼ばれるものである.
数のリスト `lst`が与えられたとき，ソートされたリストを返すプログラム`kadai(lst)` を作れ.

1. データの総数 n を1.3 で割って小数点以下を切り捨てた数を間隔hとおく.
2. i = 0〜 n - 1 について，i + h < n ならば，i番目とi + h番目のデータを調べて大小関係が逆であれば入れ換える.
3. h > 1 ならば，h = Math.floor(h / 1.3) によって hを更新して 2. を繰り返す. h == 1 ならば，hは変更せず，2においてデータの変更が起こらなくなるまで繰り返す.



解答例 (1)



combソートのプログラムは以下のようになる. 適当な感覚で入れ替えるだけなので単純である.

```
function kadai(lst){
    var n = lst.length
    var h = Math.floor(n / 1.3)
    while (true){
        var i = 0
        var changed = false
        while (i + h < n){
            if (lst[i] > lst[i + h]){
                var tmp = lst[i]
                lst[i] = lst[i + h]
                lst[i + h] = tmp
                changed = true
            }
            i += 1
        }
        if (h == 1 && ! changed) return lst
        if (h > 1)  h = Math.floor(h / 1.3)
    }
}
```

解答例 (2)

計算時間は以下のようになる．10000個のランダムな要素についてのソートではクイックソートを下回る5msでソートできる．

```
function test(){
    function make_data(n){
        var i = 0
        var res = []
        while (i < n){
        res.push(Math.random())
            i += 1
        }
        return res
    }
    var N = 10000
    var lst = make_data(N)
    var t1 = new Date()
    res = kadai(lst)
    var t2 = new Date()
    puts(res[0] + ":" + res[N - 1])
    puts("Time = " + (t2 - t1) + "ms")
}

test()
```

```
0.00002189157040211054:0.9999855880306
01
Time = 5ms
```

解答例 (3)

combソートには簡単な最適化が知られている．h = 9 または h = 10 になったとき，強制的に h = 11 に上げることによって，h = 1 となるまえに，もともと左端にある大きな数の要素（カメ(turtle)と呼ばれている）をあらかじめ右に動かせる可能性が高くなり，h=1のときの繰返し回数が減って効率が上がる．

```
 9 -> 6 -> 4 -> 3 -> 2 -> 1
10 -> 7 -> 5 -> 3 -> 2 -> 1
```

のように変化する代わりに

```
11 -> 8 -> 6 -> 4-> 3 -> 2 -> 1
```

なる．この工夫を含めたアルゴリズムをcombsort11と呼ぶ．

解答例 (4)

- comsort11のプログラムは以下のようになる.

```
function kadai(lst){
    var n = lst.length
    var h = Math.floor(n / 1.3)
    while (true){
        var i = 0
        var changed = false
        while (i + h < n){
            if (lst[i] > lst[i + h]){
                var tmp = lst[i]
                lst[i] = lst[i + h]
                lst[i + h] = tmp
                changed = true
            }
            i += 1
        }
        if (h == 1 && ! changed) return lst
        if (h > 1)  h = Math.floor(h / 1.3)
        if (h == 9 || h == 10) h = 11
    }
}
```

30万個の数のソート

```
0.00000161313692148556950:0.99999644173950400
0.00000161313692148556950:0.99999644173950400
CombSort   Time = 86ms
CombSort11 Time = 80ms
```

状況にもよるが，若干性能は上がる

←ここを追加する