プロ演1::課題3::説明資料３

演習トップページ

→ 問３ → 説明資料３ → START ボタン

http://www.ced.is.utsunomiya-u.ac.jp

/lecture/2004/prog/p1/kadai3/lesson3/

1. 補足：配列と反復

例：`int n[10]; // 10個intの配列変数 n を定義`

```
scanf("%d...%d", &n[0], ..., &n[9]);
```

正しいが柔軟性の足りない直接法

```
int i;
for(i=0; i<10; i++) scanf("%d", &n[i]);
```

同じ効果で柔軟性のある反復法

（10ではなく10000の場合を考えてみよう）

２．行列積算のプログラム

```
#include <stdio.h>
#define M 2
#define N 3
#define P 2
int main(void) {
  int a[M][P], b[P][N], c[M][N], i, j, k;
  /* a と b を入力 */
  for(i=0; i<M; i++)
    for(j=0; j<N; j++) {
      c[i][j] = 0; /* 初期化しておく！ */
      for(k=0; k<P; k++)
        c[i][j] += a[i][k] * b[k][j];
    }
  /* c を出力 */
}
```

３．整列(ソーティング)

データを昇(降)順となるように並べ換える．

// 講義「データ構造とアルゴリズム」で勉強済み!

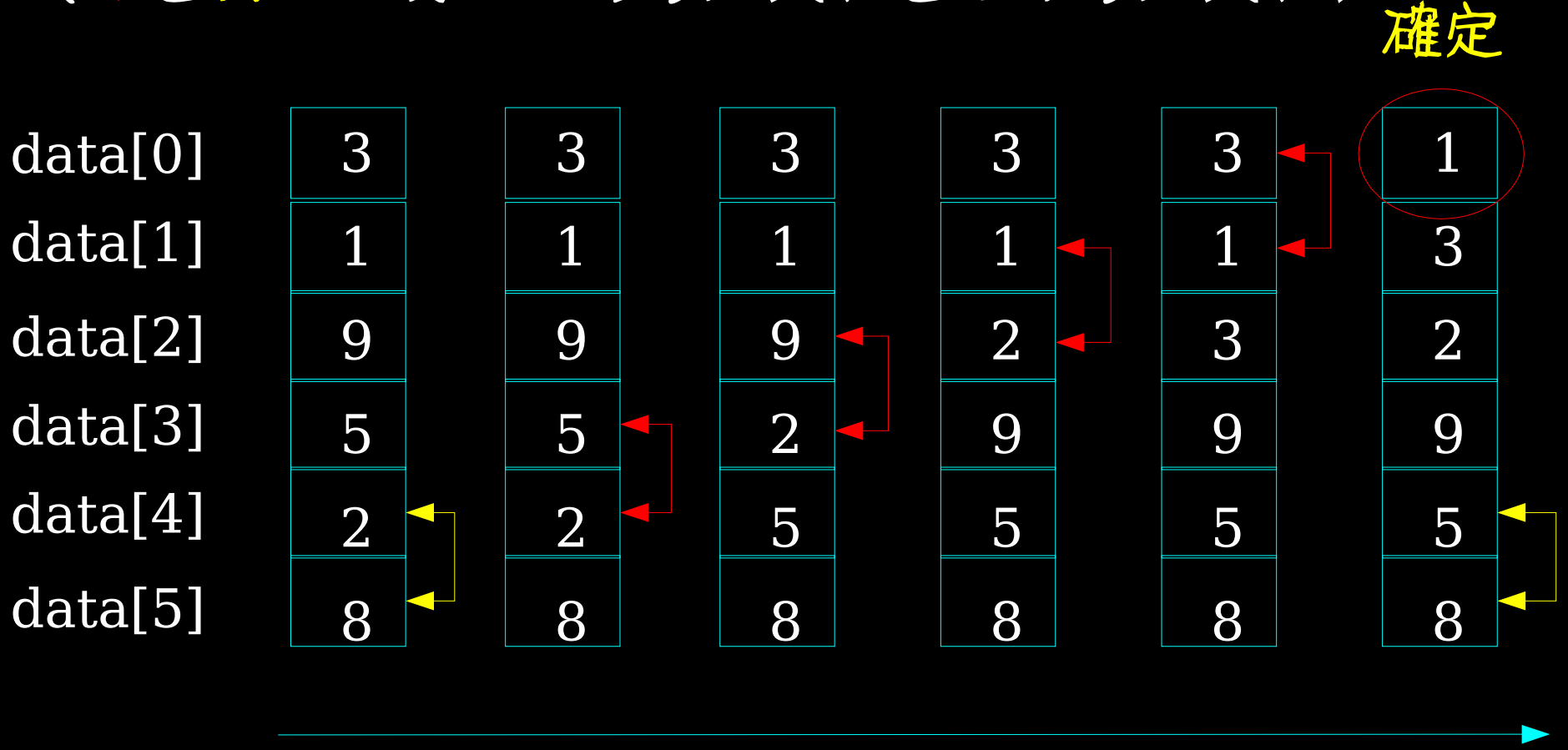
バブルソートの実装を参考にし，

レポート題目：1．単純挿入法を実装する
2．クィックソートを理解する
3．両者のパフォーマンスを考察する

4．参考：バブルソート

特徴：順に隣接データを比較し交換を試みる

（赤と黄色：交換のあった比較となかった比較）



5．バブルソート関数

```
/* a: ソート対象の配列，num: データ数 */
void bubble_sort(int a[], int num) {
  int i, j, tmp;
  for(i=0; i<num-1; i++)
    for (j=num-1; j>i; j++)
      if (a[j-1] > a[j]) { // 交換を試みる
        tmp    = a[j];
        a[j]   = a[j-1];
        a[j-1] = tmp;
      }
}
```

交換

6．単純挿入法

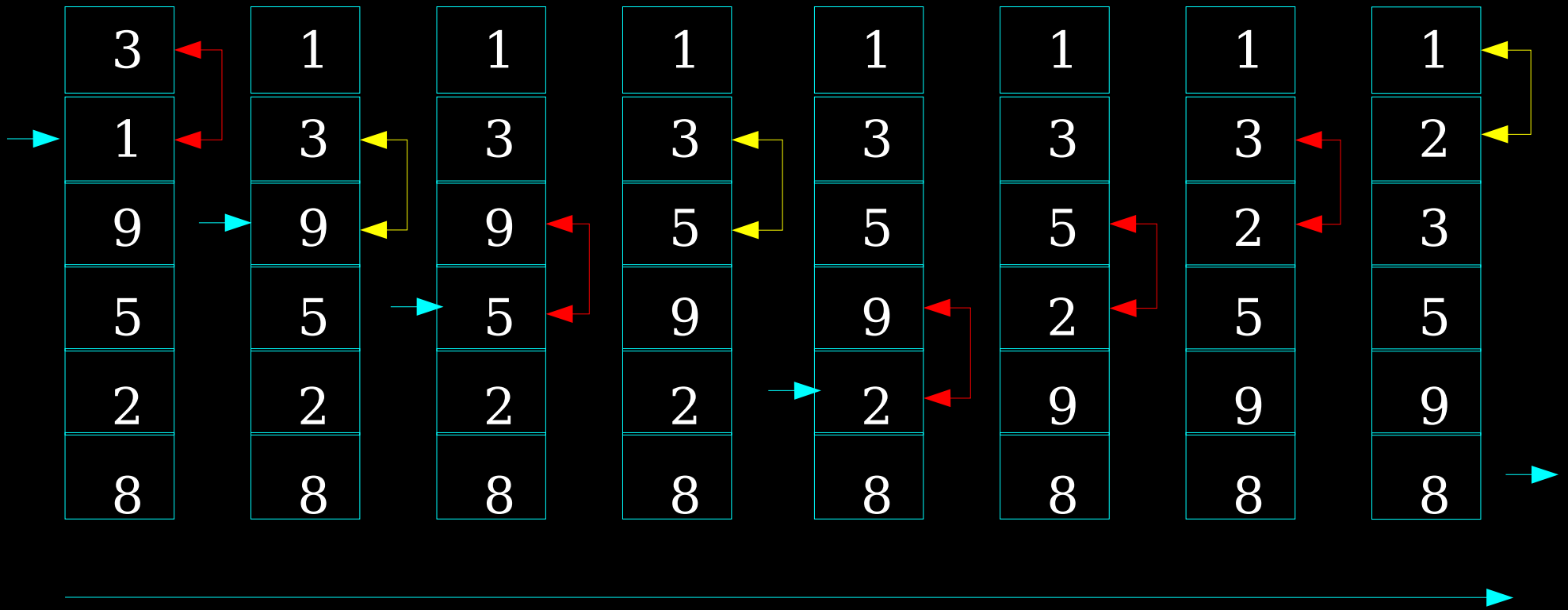
方法：data[i]をdata[1]...data[i]の
〃適切〃な場所に挿入する，i=2, ..., N.

適切とは，挿入後のdata[1]...data[i]が
昇順となるような場所のこと．

注：挿入は一連の交換からなる．次の例を参照

7．単純挿入法の動作例

i: → 交換のあった比較とない比較：赤と黄色



８．単純挿入法の練習(9分)

以下のデータを，紙とペンで，単純挿入法でソートする場合の様子を終りまで示し，TAにチェックしてもらう．

85, 33, 26, 15, 16, 54, 63

９．クィックソート

データを大まかに〝小〟と〝大〟二つのグループに分けておき，それぞれに対し再帰呼び出しを行う．

具体的に，軸を選び，軸より小さいのは〝小〟，大きいのは〝大〟にする．軸の選び方と分け方がキーポイント．

10. クィックソートの関数

```
 1 int q_sort(int a[], int head, int tail) {
 2   int pivot = (a[head] + a[tail]) / 2;
 3   int i = head, j = tail, tmp;
 4   while(i <= j) {
 5     while (a[i] < pivot) i++;
 6     while (a[j] > pivot) j--;
 7     if (i <= j) {
 8       tmp    = a[j];
 9       a[j--] = a[i];
10       a[i++] = tmp;
11     }
12   }
13   if (head < j) q_sort(a, head, j);
14   if (tail > i) q_sort(a, i, tail);
15 }
```

１１．クィックソート練習(9分)

以下のデータを，紙とペンでクィックソートでソートする場合の様子を終りまで示し，TAにチェックしてもらう．

85, 33, 26, 15, 16, 54, 63

１２．ソート関数をテストする

```
#include <stdio.h>
#define MAX 0x10000
/* ここでソート関数を定義する */
int main(void) {
  int data[MAX], num, i;
  scanf("%d", &num); /* データ数の入力 */
  for(i=0; i<num; i++) /* データの入力 */
    scanf("%d", &data[i]);
  /* data[0]...data[num]をソートする */
  for(i=0; i<num; i++) /* 結果を出力 */
    printf("%d ", data[i]);
}
```

１３．ソートの時間を計る

赤：追加分

```
#include <stdio.h>
#include <time.h>
...
int main(void) {
  clock_t start, stop;
  ...
  start = clock(); /* スタートの時間を取得*/
  /* data[0]...data[num]をソートする */
  stop  = clock(); /* ストップの時間を取得*/
  /* ソートにかかった時間(単位 ms)は，
   (stop-start)*1000/CLOCKS_PER_SEC となる */
  ...
}
```

14. レポート課題

1. 単純挿入法を実装(名前が**insert_sort**とする).

2. クィックソートについて，以下の質問に答える.
   Q: 4と7行目の **i<=j** は， **i<j** に変えてもいいか.
   それぞれについて理由を述べよう.

3. 単純挿入法とクィックソートのパフォーマンスを考察せよ. 詳細は，Webページを参照する.

１５．オプション課題

成績がアップすることがある！

// ヒント：講義「データ構造とアルゴリズム」の資料

1. 単純挿入法における番兵の使い方を述べ，
   それを実装してパフォーマンスを考察する．

2. クィックソートにおいては，軸の選び方が大切．
   2行目のやり方の問題点を述べよう(ヒント：オーバフロー)．別の選び方を使った実装を考察する．