

2016 年度 プログラミング II レポート 14

学生用

学籍番号：_____

氏名：_____

下記の注意事項を守り、次ページ以降の問いに答え、レポートを完成させなさい。

提出期限：2017 年 1 月 31 日 (火) 13:00 まで

提出場所：理学部棟 正面玄関内に設置のレポートボックス

注意事項：

- (1) このページを印刷し、必要事項を記入の上 (学籍番号欄と氏名欄は2箇所あるので忘れずに記入すること)、レポートの表紙として提出すること。
- (2) コンピュータ端末室を利用する場合は、情報システム利用ガイドラインを厳守すること。特に、コンピュータ端末室では飲食禁止である。
- (3) クラスメイトのレポートを参考にしたり、クラスメイトと協力してレポートを作成した場合は、教員控の協力者氏名欄にクラスメイトの氏名を記入すること。これらの場合も、自分の言葉で表現し直すこと。コピー禁止。
- (4) プログラミング II について、あなたの声を聞かせてください (教員控の意見・質問欄に記入のこと)。気軽にどうぞ (成績には一切影響しません)。

出題者：幸山 直人

出題日：2017 年 1 月 25 日 (水)

----- 切り取り線 -----

2016 年度 プログラミング II レポート 14

教員控

学籍番号：_____

氏名：_____

協力者氏名：_____, _____, _____

レポート作成に要した時間：_____. _____ 時間

意見・質問：

問 1 ヒープソートのアルゴリズムを用いて、配列 $a[11] = \{0, 23, 12, 48, -4, 22, 35, 88, 30, 15, 11\}$ を昇順にソートしなさい (空欄を埋め、下表を完成しなさい)。ただし、空欄には、要素 $a[0]$ は除き、テキスト (p.174) のソースプログラム「ヒープソートによるソートプログラム」(heapsort.c) の第 33 行および第 51 行の関数「`printarray()`」によって出力される配列を記述すること (テキストの Step とは関係ない)。
ヒント: ヒープソートの特徴である 2 分木を描きながら (親子関係に注意しながら) 考えましょう。

ヒープ化: 第 33 行の関数「`printarray()`」による出力

{0, 23, 12, 48, -4, 22, 35, 88, 30, 15, 11}

{0, 23, 12, 48, 30, 22, 35, 88, -4, 15, 11}

{0, 23, 12, 88, 30, 22, 35, 48, -4, 15, 11}

{0, 23, 30, 88, 15, 22, 35, 48, -4, 12, 11}

{0, 88, 30, 48, 15, 22, 35, 23, -4, 12, 11}

ダウンヒープ: 第 51 行の関数「`printarray()`」による出力

{0, 48, 30, 35, 15, 22, 11, 23, -4, 12, 88}

{0, 35, 30, 23, 15, 22, 11, 12, -4, 48, 88}

{0, 30, 22, 23, 15, -4, 11, 12, 35, 48, 88}

{0, 23, 22, 12, 15, -4, 11, 30, 35, 48, 88}

{0, 22, 15, 12, 11, -4, 23, 30, 35, 48, 88}

{0, 15, 11, 12, -4, 22, 23, 30, 35, 48, 88}

{0, 12, 11, -4, 15, 22, 23, 30, 35, 48, 88}

{0, 11, -4, 12, 15, 22, 23, 30, 35, 48, 88}

{0, -4, 11, 12, 15, 22, 23, 30, 35, 48, 88}

問 2 テキスト (p.174) のソースプログラム「ヒープソートによるソートプログラム」(heapsort.c) を書き換えて、降順にソートするプログラム「ヒープソート: 降順」(report 14_01.c) を作成しなさい。さらに、ソースプログラムをコンパイルし、実行可能ファイルが正しく動作するか確認しなさい。なお、作成したソースプログラムは印刷してレポートに添付すること。

解答例 別紙を参照のこと。

問 3 テキストを参考に、マージソートによって昇順にソートリングするソースプログラム「マージソート：昇順」(**report14_01.c**)を作成しなさい。ただし、要素 **a[0] = 0** とし、ソートリングの範囲に含めないこと。さらに、ソースプログラムをコンパイルし、実行可能ファイルが正しく動作するか確認しなさい。なお、作成したソースプログラムは印刷してレポートに添付すること。
注意：クイックソートを参考に、マージソートのソースプログラムを作成するとよい。
注意：別途ソートリング用の配列を用意し、ソートリング終了後、元の配列に書き戻す。

解答例 別紙を参照のこと。

問 2 の解答例 「ヒープソート：降順」(report14_01.c)

```

:      ----- 第 1 行から第 16 行までは同じなので省略 -----

17: void heapsort(int a[])
18: {
19:     int i, j, k, n = N, x;
20:
21:     for (k = n / 2; k >= 1; k--) {
22:         i = k;
23:         x = a[i];
24:         j = 2 * i;
25:         while (j <= n) {
26:             if (j < n && a[j] > a[j + 1]) j++;
27:             if (x <= a[j]) break;
28:             a[i] = a[j];
29:             i = j;
30:             j = 2 * i;
31:         }
32:         a[i] = x;
33:         printarray(a);
34:     }
35:     printf("%d\n", n);
36:
37:     while (n > 1) {
38:         x = a[n];
39:         a[n] = a[1];
40:         n--;
41:         i = 1;
42:         j = 2 * i;
43:         while (j <= n) {
44:             if (j < n && a[j] > a[j + 1]) j++;
45:             if (x <= a[j]) break;
46:             a[i] = a[j];
47:             i = j;
48:             j = 2 * i;
49:         }
50:         a[i] = x;
51:         printarray(a);
52:     }
53: }
54:
:      ----- 関数「printarray()」は同じなので省略 -----
```

問 3 の解答例 「マージソート：昇順」 (report14_02.c)

```
1: #include <stdio.h>
2:
3: #define N 10
4:
5: void mergesort(int a[], int start, int end);
6: void printarray(int a[], int start, int end);
7:
8: int main(void)
9: {
10:     int a[N + 1] = {0, 4, 10, 5, 2, 1, 7, 8, 6, 3, 9};
11:
12:     mergesort(a, 1, N);
13:
14:     return 0;
15: }
16:
17: void mergesort(int a[], int start, int end)
18: {
19:     int i, j, k, mid, tmp[N + 1];
20:
21:     if (start != end) {
22:         mid = (start + end) / 2;
23:         mergesort(a, start, mid);
24:         mergesort(a, mid + 1, end);
25:
26:         // 分割された 2 つの配列をソートしながら一時的な配列 (tmp[]) に結合
27:         // 1 つ目の配列の要素が余った場合、残りを一時的な配列 (tmp[]) へ書き出す
28:         // 2 つ目の配列の要素が余った場合、残りを一時的な配列 (tmp[]) へ書き出す
29:         // ソーティングが完了した一時的な配列 (tmp[]) を元の配列 (a[]) に書き戻す
30:
31:         // ----- 続く -----
32:     }
```

参考：

第 21 行	分割の条件 (要素が 1 つになるまで)
第 22 行	分割位置の決定
第 23 行～第 24 行	分割の実行 (再帰的)
第 26 行	1 つ目の配列の開始位置の設定
第 27 行	2 つ目の配列の開始位置の設定
第 29 行～第 38 行	分割された 2 つの配列をソーティングしながら一時的な配列 (tmp[]) に結合
第 39 行～第 42 行	1 つ目の配列の要素が余った場合、残りを一時的な配列 (tmp[]) へ書き出す
第 43 行～第 46 行	2 つ目の配列の要素が余った場合、残りを一時的な配列 (tmp[]) へ書き出す
第 48 行	ソーティングが完了した一時的な配列 (tmp[]) を元の配列 (a[]) に書き戻す

問 3 の解答例 「マージソート：昇順」(report14_02.c)

```

:           ----- 続き -----

26:         i = start;
27:         j = mid + 1;
28:         for (k = start; k <= end; k++) {
29:             if (i <= mid && j <= end) {
30:                 if (a[i] < a[j]) {
31:                     tmp[k] = a[i];
32:                     i++;
33:                 }
34:                 else {
35:                     tmp[k] = a[j];
36:                     j++;
37:                 }
38:             }
39:             else if (i <= mid) {
40:                 tmp[k]=a[i];
41:                 i++;
42:             }
43:             else if (j <= end) {
44:                 tmp[k]=a[j];
45:                 j++;
46:             }
47:         }
48:         for (k = start; k <= end; k++) a[k] = tmp[k];
49:         printarray(a, start, end);
50:     }
51: }
52:
53: void printarray(int a[], int start, int end)
54: {
55:     int i;
56:
57:     for (i = 1; i <= N; i++) {
58:         if (i == start) printf("{");
59:         printf("%3d ", a[i]);
60:         if (i == end) printf("}");
61:     }
62:     printf("\n");
63: }
```