令和元年８月３日（土）9:00～12:00

大阪大学大学院情報科学研究科

コンピュータサイエンス専攻
情報システム工学専攻
情報ネットワーク学専攻
マルチメディア工学専攻
バイオ情報工学専攻

令和２年度　博士前期課程　入試問題

（A）情報工学

【注意事項】

- 問題数は必須問題２題（問題 1～2），選択問題５題（問題 3～7），合計７題である．
  必須問題は２題すべて解答すること．また，選択問題は２題を選択して解答すること．
- 問題用紙は表紙を含めて 18 ページである．
- 解答用紙は全部で５枚ある．
  １枚目（赤色）の解答用紙には問題１（必須問題）の解答を
  ２枚目（青色）の解答用紙には問題２（必須問題）の(1)の解答を
  ３枚目（緑色）の解答用紙には問題２（必須問題）の(2)の解答を
  ４枚目（白色）の解答用紙には問題 3～7（選択問題）から選択した１題の解答を
  ５枚目（白色）の解答用紙には問題 3～7（選択問題）から選択したもう１題の解答を
  それぞれ記入すること．
  解答用紙は間違えると採点されないことがあるので注意すること．
- 解答用紙は５枚すべてを回収するので，すべての解答用紙に受験番号を記入すること．
- 解答用紙の「試験科目」の欄には解答した問題の科目名（「アルゴリズムとプログラミング」など）を記入すること．
  また，選択問題調査票には，選択した問題の番号（3～7 から二つ）に○をつけること．
- 解答欄が不足した場合は裏面を使用すること．その際，表面末尾に「裏面に続く」と明記しておくこと．解答用紙の追加は認めない．
- 解答用紙には，日本語または英語で解答すること．

令和元年８月３日（土）9:00～12:00

大阪大学大学院情報科学研究科

コンピュータサイエンス専攻
情報システム工学専攻
情報ネットワーク学専攻
マルチメディア工学専攻
バイオ情報工学専攻

令和２年度　博士前期課程　入試問題

（A）情報工学

【注意事項】

- 問題数は必須問題２題（問題 1～2），選択問題５題（問題 3～7），合計７題である．
  必須問題は２題すべて解答すること．また，選択問題は２題を選択して解答すること．
- 問題用紙は表紙を含めて 18 ページである．
- 解答用紙は全部で５枚ある．
  １枚目（赤色）の解答用紙には問題１（必須問題）の解答を
  ２枚目（青色）の解答用紙には問題２（必須問題）の(1)の解答を
  ３枚目（緑色）の解答用紙には問題２（必須問題）の(2)の解答を
  ４枚目（白色）の解答用紙には問題 3～7（選択問題）から選択した１題の解答を
  ５枚目（白色）の解答用紙には問題 3～7（選択問題）から選択したもう１題の解答を
  それぞれ記入すること．
  解答用紙は間違えると採点されないことがあるので注意すること．
- 解答用紙は５枚すべてを回収するので，すべての解答用紙に受験番号を記入すること．
- 解答用紙の「試験科目」の欄には解答した問題の科目名（「アルゴリズムとプログラミング」など）を記入すること．
  また，選択問題調査票には，選択した問題の番号（3～7 から二つ）に○をつけること．
- 解答欄が不足した場合は裏面を使用すること．その際，表面末尾に「裏面に続く」と明記しておくこと．解答用紙の追加は認めない．
- 解答用紙には，日本語または英語で解答すること．

1 【必須問題】アルゴリズムとプログラミング

配点：(1) 12 点，(2) 12 点，(3) 14 点，(4) 25 点，(5-1) 20 点，(5-2) 30 点，(6) 12 点

図 1 に示す ANSI-C 準拠である C 言語のプログラム（program）は，複数の整数（integer）のデータ（data）を，二分木（binary tree）を利用して昇順（ascending order）に整列（sort）して出力（output）するプログラムである．図 1 のプログラムでは，配列（array）の添え字（index）が二分木の節点番号（node number）に対応している．ただし，二分木の根（root）の節点番号を 0 とし，節点番号が i の節点に子（child）がある場合，左の子の節点番号を $2i+1$，右の子の節点番号を $2i+2$ とする．また，配列に格納されたデータは，二分木の対応する節点のデータを示している．

整列するデータは図 2 に示すような形式（format）のファイル data.txt で与えられ，1 行目には整列するデータの個数 n（$\geqq 1$），2 行目以降の n 行には整列するデータの値（value）が書かれている．図 3 は，図 2 の data.txt を与えて図 1 のプログラムを実行した場合の，28 行目が実行される直前の配列 d に対応する二分木であり，丸が節点，丸の左側の数字が節点番号，丸の中の数字がデータの値，線分が枝（edge）を示している．図 1 のプログラムに関する以下の各問に答えよ．

(1) 40 行目で呼び出されている関数（function）sort で実現されている整列アルゴリズム（sorting algorithm）は，一般に何と呼ばれているか名称を答えよ．

(2) 図 2 の data.txt を与えてプログラムを実行した場合の，28 行目が実行された直後の配列 d に対応する二分木を図示せよ．ただし，図 3 にならい，丸で節点，丸の左側の数字で節点番号，丸の中の数字でデータの値，線分で枝を示すこと．

(3) 11 行目および 12 行目が実行されることにより，節点番号が current の節点のデータとその子のデータの間に成立する関係を説明せよ．

(4) 関数 sort で実現されている整列アルゴリズムの最悪時間計算量（worst case time complexity）を，整列するデータの個数 n を用いて理由と共にオーダ表記（order notation）で示せ．

(5) 関数 sort において，28 行目の実行時に関数 swap が呼び出される回数を $T(n)$ とする．n は整列するデータの個数である．28 行目を変更し，28 行目の for ループの繰り返し回数と $T(n)$ の最大値をできる限り削減（28 行目の実行に要する最悪時間計算量を削減）することを考える．以下の各小問に答えよ．

(5-1) 下記の（あ）～（え）を埋めて変更後の 28 行目を完成させよ．

```
for (i = （あ）; 0 <= i; i--) downh( （い） , （う） , （え） );
```

(5-2) 変更後のプログラムにおける $T(n)$ の n に関するオーダ表記を理由と共に示せ．

$\sum_{j=0}^{h} \frac{j}{2^j} = 2 - \frac{2+h}{2^h}$ を用いてよい．

(6) 下線（ア）～（エ）で示す条件式を必要に応じて変更し，データを降順（descending order）に整列して出力することを考える．変更後のプログラムにおける下線（ア）～（エ）の条件式をそれぞれ答えよ．

```
1   #include <stdio.h>
2   #include <stdlib.h>
3   void swap(int d[], int p, int q) {
4       int tmp;
5       tmp = d[p]; d[p] = d[q]; d[q] = tmp;
6   }
7   void downh(int d[], int n, int k) {
8       int child, current = k;
9       while (current < n / 2) {
10          child = current * 2 + 1;
11          if ((child + 1 < n) && (d[child] < d[child + 1])) child++;
                    (ア)                   (イ)
12          if (d[current] < d[child]) swap(d, current, child);
                       (ウ)
13          else break;
14          current = child;
15      }
16  }
17  void uph(int d[], int k) {
18      int parent, current = k;
19      while (0 < current) {
20          parent = (current - 1) / 2;
21          if (d[parent] < d[current]) swap(d, parent, current);
                        (エ)
22          else break;
23          current = parent;
24      }
25  }
26  void sort(int d[], int n) {
27      int i;
28      for (i = 1; i < n; i++) uph(d, i);
29      for (i = n - 1; 0 < i; i--) {swap(d, 0, i); downh(d, i, 0);}
30  }
31  int main() {
32      int i, N, *D;
33      FILE* fp;
34      fp = fopen("data.txt", "r");
35      fscanf(fp, "%d", &N);
36      D = (int*) malloc(sizeof(int) * N);
37      for (i = 0; i < N; i++) fscanf(fp, "%d", &D[i]);
38      fclose(fp);
39
40      sort(D, N);
41
42      for (i = 0; i < N; i++) printf("%d ", D[i]);
43      printf("\n");
44      free(D);
45      return 0;
46  }
```

図 1　プログラム

```
6
40
30
50
10
60
20
```

図 2　data.txt

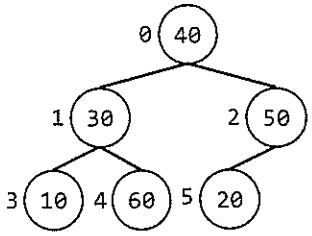
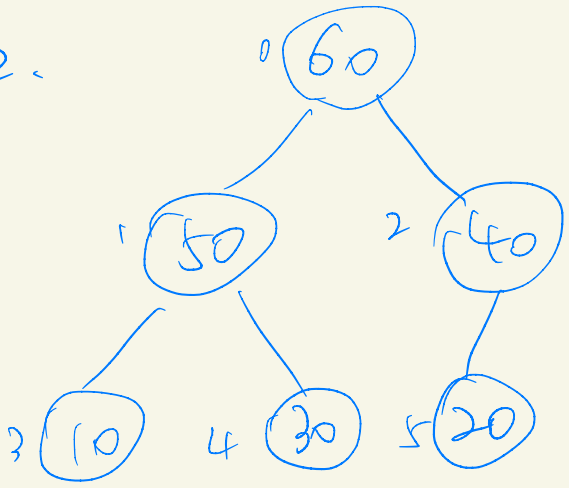


図 3　二分木の例

1. ヒープソート Heapsort

Aiyi Li

2.



3. current のデータ > その子のデータ

4. O(nlogn)

5. (1)(ア) n/2-1 (イ) d (ウ) n (エ) i

(ケ-2) T(n): O(n)

説明：因数

6. (ア) (変更なし)

(イ) d[child] > d[child+1]

(ウ) d[current] > d[child]

(エ) d[parent] > d[current]