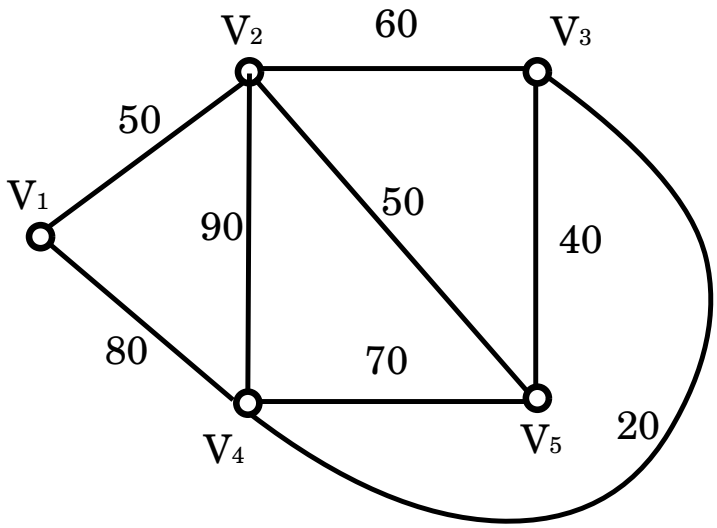
# 情報知能学科「アルゴリズムとデータ構造」試験解答

出題者: 白井英俊　　日時: 2010 年 7 月 27 日 16:00–17:30 (826 教室)

注意: これは教科書のみ持ち込み可の試験である。解答は問題番号を明記して解答用紙に書くこと。解く順番は任意でよい。また解答用紙が足りなければ裏面を使うこと。それでも足りない場合は追加の解答用紙を配るので手をあげること。

問題 1: 下図で示されるグラフにおいて、すべての頂点間の最短距離を求めることを考えよう。なお辺の上の数は、頂点の間の距離 (コスト) を表すものとする。

そのための方法として講義では All-Shortest-Paths アルゴリズムを紹介した。下図に対する All-Shortest-Paths アルゴリズムの振る舞いを示せ。




## 問 1 の解答

| k | $(V_1, V_2)$ | $(V_1, V_3)$ | $(V_1, V_4)$ | $(V_1, V_5)$ | $(V_2, V_3)$ | $(V_2, V_4)$ | $(V_2, V_5)$ | $(V_3, V_4)$ | $(V_3, V_5)$ | $(V_4, V_5)$ |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 50 | $\infty$ | 80 | $\infty$ | 60 | 90 | 50 | 20 | 40 | 70 |
| 1 | 50 | $\infty$ | 80 | $\infty$ | 60 | 90 | 50 | 20 | 40 | 70 |
| 2 | 50 | 110 | 80 | 100 | 60 | 90 | 50 | 20 | 40 | 70 |
| 3 | 50 | 110 | 80 | 100 | 60 | 80 | 50 | 20 | 40 | 60 |
| 4 | 50 | 100 | 80 | 100 | 60 | 80 | 50 | 20 | 40 | 60 |
| 5 | 50 | 100 | 80 | 100 | 60 | 80 | 50 | 20 | 40 | 60 |

問題 2: ヒープについて問う。

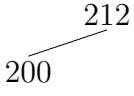
(問 2-1): 次の数を入力としてヒープが構成される過程を示せ。 ただしここでは木として表すとする。

212　200　53　76　179　162　16　131　221　123　100

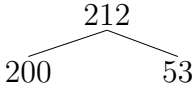
## 問 2-1 の解答

(1)　212

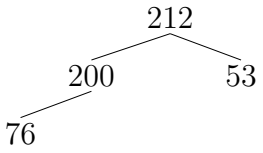
(2)



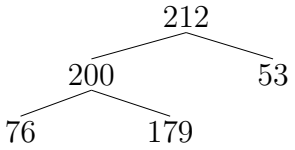

(3)



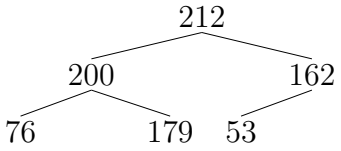
(4)


212
200
53
76

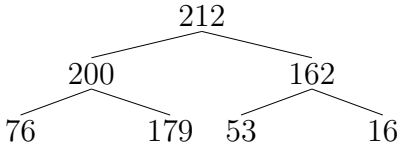
(5)


212
200
53
76
179

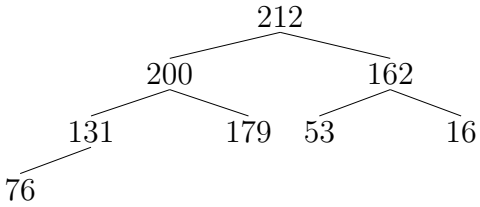
(6)


212
200
162
76
179
53

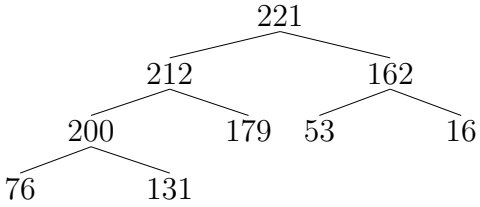
(7)


212
200
162
76
179
53
16

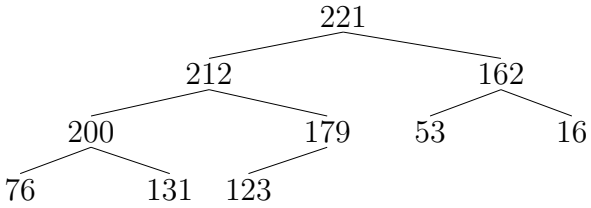
(8)


212
200
162
131
179
53
16
76

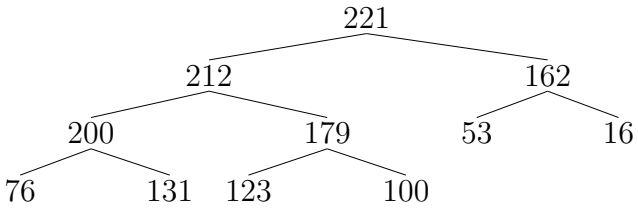
(9)


221
212
162
200
179
53
16
76
131

(10)


221
212
162
200
179
53
16
76
131
123

(11)


221
212
162
200
179
53
16
76
131
123
100

**(問 2-2)**:**(問 2-1)** で得られたヒープ (二進木) において、(a) 根、(b)「179」を値にもつ頂点の親、(c)「179」を値にもつ頂点の兄弟、(d) 「179」を値にもつ頂点の子 (すべて)、(e) 葉 (すべて)、の頂点の値をそれぞれ答えよ。

## 問 2-2 の解答

(a) 221

(b) 212

(c) 200

(d) 123, 100

(e) 76, 131, 123, 100, 53, 16

**(問 2-3)**: **(問 2-1)** で得られたヒープは、配列としても表現できる。その配列を書け 。ただし、配列の要素だけではなく、インデックス (添字、最初のインデックスを 1 とする) も明示すること。

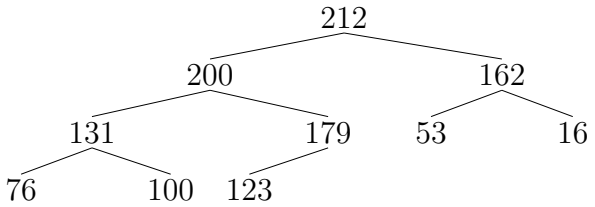
## 問 2-3 の解答

| 添字 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 要素 | 221 | 212 | 162 | 200 | 179 | 53 | 16 | 76 | 131 | 123 | 100 |

**(問 2-4)**: **(問 2-1)** の結果得られたヒープに対し、根とヒープの最後の要素とを取り替え、最後の要素（つまり、元の「根」）を削除したとする。それに対して「ヒープを修復した (remake)」結果、得られるヒープを示せ。

## 問 2-4 の解答

木で表すと




配列で表すと:

| 添字 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 要素 | 212 | 200 | 162 | 131 | 179 | 53 | 16 | 76 | 100 | 123 |

**(問 2-5)**: 頂点の個数が $n$ のヒープに対し、根とヒープの最後の要素とを取り替え、最後の要素（つまり、元の「根」）を削除した後に、「ヒープを修復する (remake)」とする。このときのヒープ修復の時間複雑度 (時間計算量) のオーダは $n$ を用いてどのように表されるか、その根拠もあわせて答えよ。

## 問 2-5 の解答

頂点の個数が $n$ のヒープの高さはほぼ $\log n$ である。ヒープの修復はこの高さに比例した手間で行われるので、ヒープ修復のための時間計算量のオーダは $O(\log n)$ である。

**問題 3**: ソートについて問う。

**(問 3-1)**: ソートとはどのような処理のことか、簡潔に述べよ。

## 問 3-1 の解答

簡単のため、数を要素とする集合に対するソートについて説明する。数を要素とする集合に対するソートとは、入力として数を要素とする集合をとり、その集合の要素を昇順もしくは降順で並べたもの（多くはリストか配列）を返す。そのためのアルゴリズムとしては、マージソートやクイックソートがしられている。

(問 3-2): ソートのアルゴリズムに、マージソートとクイックソートがある。それぞれのアルゴリズムを簡潔に説明し、入力の大きさを $n$ としたときの、それぞれの時間複雑度 (時間計算量) のオーダを答えよ。また、オーダがそのように計算される根拠も述べよ。

## 問 3-2 の解答

クイックソートとは、入力から教科書 26 ページに示された手順により二分木を作った後に、その木の頂点を中間順になぞることで、ソートする方法である。平均的には、木の高さが $\log n$ となり、$n$ 個の要素から二分木を作ることから、二分木を作るために必要な時間計算量は $O(n \log n)$ である。またこれを中間順でなぞる時間計算量も、木の高さ $\log n$ だけの辺をたどって $n$ 個の要素を取出すため、全体の時間計算量は $O(n \log n)$ となる。ただし、最悪の場合は、$O(n^2)$ となる。

マージソートとは、分割統治法によるソートの一つである。集合 S を入力とすると、S を二分割し、それぞれを「マージソート」した後で、二つの要素を昇順（もしくは降順）になるようマージする方法である。$n$ 個の要素をマージソートする時間計算量を $f(n)$ で表すと、$n \geq 1$ の場合、$f(n) = 2 * f(\frac{n}{2}) + n$ が成り立つ。教科書 52 ページの記述より、$f(n) = n \log n$ となる。

別解： S を根とし、その要素を二分割したそれぞれをその子の頂点とし、これを再帰的に適用して分割の木を書くと (葉は要素数 1 の集合)、この木の高さは $\log n$ となる。またマージの計算量は、マージの入力となる要素数に比例した手間である。これから全体の時間計算量は $O(n \log n)$ となる。

**問題 4**: 次の逆ポーランド記法でかかれた数式をスタックを用いて計算する過程を図示せよ (つまり、スタックの状態の変化がわかるように書くこと)。ただし、スタックの底とトップ (top) とを明示すること。

**1　2　3　4　+　*　−　24　6　/　***

## 問 4 の解答

(1) (底) 1 (top)

(2) (底) 1 2 (top)

(3) (底) 1 2 3 (top)

(4) (底) 1 2 3 4 (top)

(5) (底) 1 2 7 (top)

(6) (底) 1 14 (top)

(7) (底) -13 (top)

(8) (底) -13 24 (top)

(9) (底) -13 24 6 (top)

(10) (底) -13 4 (top)

(11) (底) -52 (top)

**問題 5**: Ruby 言語で、木の頂点が以下のような Node クラス (class) によって表されているとする。この頂点のインスタンスに対し、それを根とする (部分) 木の「高さ」を返す メソッド height のアルゴリズム、または動きを説明するコメントつきのプログラムを書け。なお、以下では (参考のため) 頂点のインスタンスに対し、子どもの頂点すべてを要素にもつ配列を返す children が定義されている。また、完成したプログラムには、アルゴリズムよりも高い評価を与える。

## 問 5 の解答

```
class Node
   def initialize(lab, par, fc, nb)
       @label = lab          # ラベル
       @parent = par         # 親頂点
       @firstChild = fc      # 第一子頂点
       @nextBrother = nb     # 兄弟頂点
   end # def of initialize
   attr_accessor :parent, :firstChild, :nextBrother, :label
  #
  def children         # 子頂点の配列を返す
    x = @firstChild        # 第一子の頂点を求める
    ans = []               # 答えの記録用
    while (x != nil)       # 子頂点がある限り
       ans.push(x)             # 答えリストにいれる
       x = x.nextBrother       # x の兄弟を探す
    end # while
    return ans             # 答えを返す
  end  # def of children
  #
  def height  # 木の高さを返す
    x = @firstChild        # 第一子の頂点を求める
    ans = 0                # 答え
    while (x != nil)       # 子頂点がある限り
       tmp = x.height+1          # x を根とする部分木の高さ+1
       ans = tmp  if ans < tmp   # ans よりも tmp が大きければ更新
       x = x.nextBrother         # x の兄弟を探す
    end # while
    return ans             # 答えを返す
  end  # def of height
end # class of Node
```