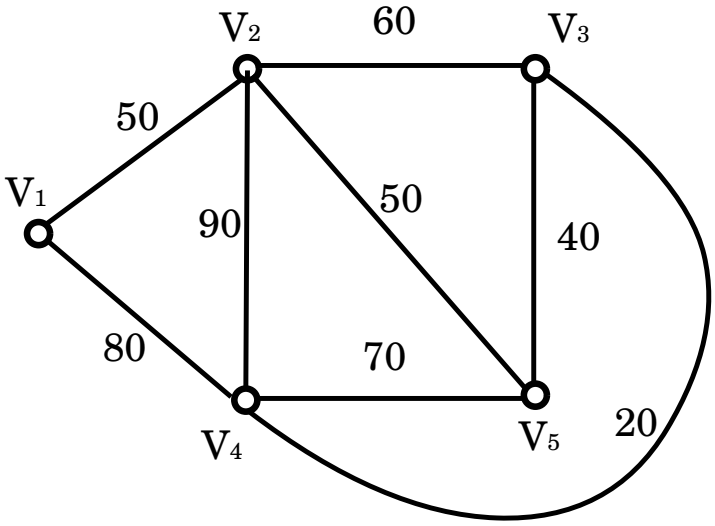
情報知能学科「アルゴリズムとデータ構造」試験問題

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 2-2):**(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 2-3): **(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 2-4): **(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 2-5): 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 3-2): 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。

```
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
     #
     #  ここに適切なコードを数行書く
     #
  end  # def of height
end # class of Node
```