データ構造とアルゴリズム

問題１．C 言語で書かれた以下の関数 f は，二つの自然数の引数 x, y が与えられたときに，printf 文でいくつかの数値をコンマ区切りで出力し，最終的に自然数を返す．この関数 f に関する以下の設問について，指定された答案用紙に解答しなさい．

```
int f(int x, int y) {
        printf("%d,", y);
        if(x%y==0) { /* x%y は x を y で割った余り */
                return y;
        } else {
                return f(y, x%y);
        }
}
```

設問(1) f(42,30)が呼ばれてから出力される文字列を示しなさい．

設問(2) f(16,26)が呼ばれてから出力される文字列を示しなさい．

設問(3) 関数 f がどんな値を返す関数かを簡潔に示しなさい．

設問(4) printf で出力される数が 2 つ以上のとき，その数列が単調減少であることを，数式などを用いて説明しなさい（厳密な証明でなくても良い）．

設問(5) 関数 f(x,y)の最悪時間計算量を，x と y に関するオーダー表記（ビッグオー記法）を用いて示しなさい．計算過程を明記すること．計算過程の記述が十分でない場合や，計算量のオーダーが必要以上に大きい場合は減点または不正解とする．

問題２．頂点 v_1 を根とする根付き木 G=(V,E)を以下の通り定める．

・頂点集合は $V=\{v_1, v_2, \cdots, v_8\}$ で，木の根は v_1 である．各頂点 v_i は値 a[i]を持つ．

・根以外の頂点 v_i の親は $v_{\lfloor i/2 \rfloor}$ である．ただし $\lfloor x \rfloor$ は x の整数部分を意味する．

配列 a の初期値は以下の通りである．

a[1]=8, a[2]=6, a[3]=4, a[4]=3, a[5]=1, a[6]=2, a[7]=7, a[8]=5

頂点 v_i が子を持たないか，全ての子の値が a[i]以上ならば頂点 v_i はヒープ条件を満たすという．以下に C 言語で書かれた関数 heapify(a,i,j)を示す．

```
void heapify(int[] a, int i, int j) {
        int k = i * 2;
        if(k <= j) {
                if((k < j) && (a[k+1] < a[k])) {
                        k++;
                }
                if(a[i] > a[k]) {
                        int tmp = a[k];
                        a[k] = a[i];
                        a[i] = tmp;
                        heapify(a, k, j);
                }
        }
}
```

以上を踏まえた上で以下の設問について，指定された答案用紙に解答しなさい．なお，配列 a の値を答える設問では， a[1],a[2],…,a[8] の順に，値をコンマで区切って示すこと．

設問(1) 木の頂点のうちヒープ条件を満たしている要素の添字を全て示しなさい（例えば v_1, v_2, v_3 がヒープ条件を満たしているならば 1,2,3 のように答える）．

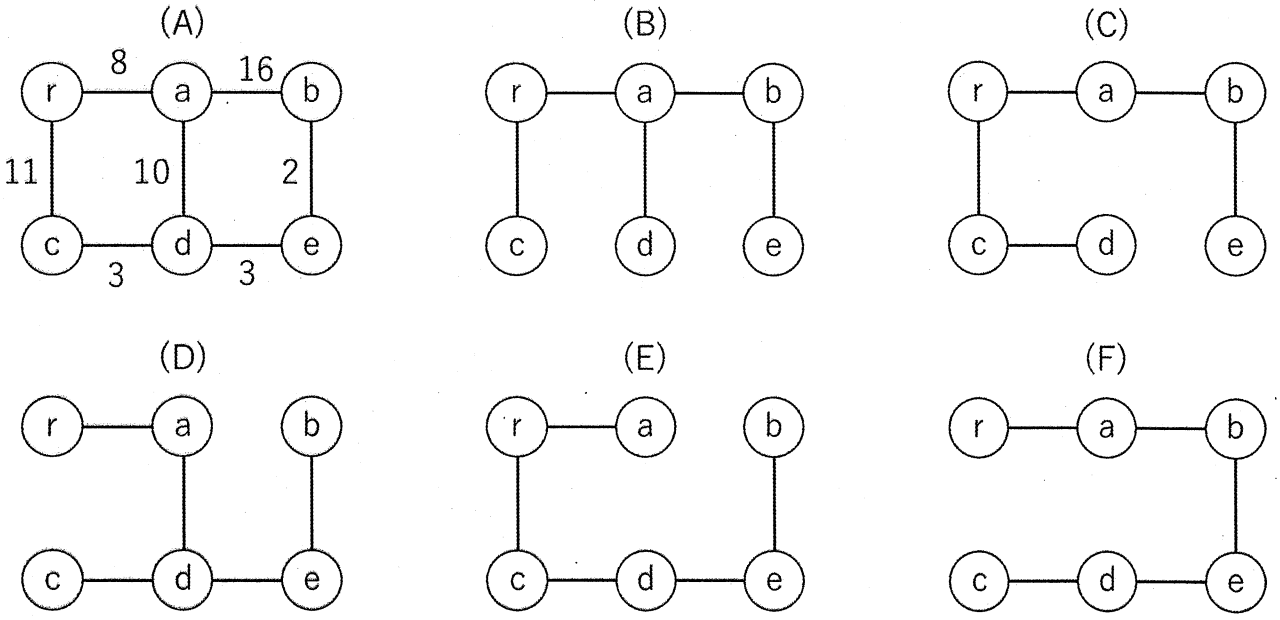
設問(2) 配列 a に heapify(a,3,8); を実行した後の配列 a の中の 8 個の値を示しなさい．

設問(3) 設問(2)の配列に，さらに heapify(a,2,8); を実行した後の配列 a の中の 8 個の値を示しなさい．

設問(4) n を自然数とする．a[1],a[2],...,a[n] が入った状態で $i=\lfloor n/2 \rfloor, \lfloor n/2 \rfloor-1, \cdots, 1$ に対して heapify(a,i,n); を実行する．そのときの時間計算量を，n を用いたオーダー表記で示しなさい．計算過程を明記すること．計算過程の記述が十分でない場合は減点あるいは不正解とする．

問題３．設問の下には，各辺に重みのある無向グラフ(A)とその部分木(B),(C),(D),(E),(F)が描かれている．各設問では部分木に関する性質が示されている．設問の条件に該当する部分木を**全て**答えなさい．例えば，解答は「 (1) B,C 」のように記入すること．該当する木がない場合は「 (1) なし」と解答する．設問中の用語については図の下にある「(用語の説明)」で定義を確認すること．

設問(1) グラフ(A)における，r を根とする深さ優先探索木
設問(2) グラフ(A)における，r を根とする幅優先探索木
設問(3) グラフ(A)の最小スパニング木
設問(4) グラフ(A)における，r を始点とする最短経路木



(用語の説明)

深さ優先探索は，探索を開始する頂点から，以下の順番で無向グラフ G の全ての頂点を訪問する．訪問済の頂点を順に $v_1, v_2, \cdots, v_k$ としたとき，k+1 番目に訪問する頂点は v_k に隣接する未訪問の頂点のいずれかである．v_k に隣接する未訪問頂点がなければ $v_{(k-1)}, v_{(k-2)}, v_{(k-3)}, \cdots$ の順に未訪問の隣接頂点を探す．新たな頂点を見つけた際の辺による G の部分グラフは**深さ優先探索木**と呼ばれる．

幅優先探索は以下の順番で無向グラフ G の全ての頂点を訪問する．まず，探索の開始点に隣接する全ての頂点を任意の順序で訪問する．その後，その時点で訪問済の頂点に隣接する全ての頂点を任意の順序で訪問する．このステップを全ての頂点を訪問するまで繰り返す．新たな頂点を見つけた際の辺による G の部分グラフは**幅優先探索木**と呼ばれる．

連結な無向グラフ G の部分木 T が G の全ての頂点を含むとき，T は G の**スパニング木**と呼ばれる．G の各辺が重みを持つとき，辺の重みの和が最小のものは**最小スパニング木**と呼ばれる．

無向グラフ G の各辺が実数の重みを持つとき，u, v の間の道のうちで道上の辺の重みの和が最小のものを u, v 間の**最短経路**という．G の頂点 r と r 以外の任意の頂点 v に対し，G の部分木 T の辺のみを通る r, v 間の道が G の最短経路であるとき，T は r を始点とする**最短経路木**と呼ばれる．