プログラミング課題（標準時間 30 分）

以下の問題を解くコードを記述してください。
コードはどんなプログラミング言語（疑似コード含む）で書いてもらっても結構です。
コードの正確性は問いません。
アルゴリズム的に正しいかどうか、非効率なアルゴリズムでないかという点が評価の対象となります。

1. 正の数、負の数を両方含みうる integer 配列 a が与えられたときに、
その中から最もゼロに近い数値を一つ返す関数を書いてください。
例：
a = [ 1, 3, -2 ] => 戻り値 = 1
a = [ -2, 3, 4 ] => 戻り値 -2

2. 正の数のみを含む二つの integer 配列 a1 および a2 が与えられたとき、
どちらか片方のみに含まれる要素の配列を返す関数を書いてください。
戻り値の配列の要素の並びは特に気にしません。

例：
a1 = [ 1, 7, 3, 2 ], a2 = [ 7, 6, 1 ] =>
戻り値 = [ 3, 2, 6 ]

3. 根ノード r を持つ二分木 t が与えられたときに、
木の「深さ」を計算する関数を書いてください。
ただし、木の深さとは、根ノードから最も遠いノードまでの距離として定義されます。
また各ノードは left, right という二つのプロパティを持ち、
子を持たない場合は null が格納されます。

4. 巨大な数値の配列が与えられたとします。
この中から１０番目に大きな要素を見つける関数を書いてください。
ただし、入力の配列は巨大なためソートすることができません。
また、入力の配列を書き換えることはできません。

5. 双方向リンクリスト(doubly linked list) l が与えられたとします。
このとき、ちょうど真ん中の要素を見つける処理を、
一つのループだけを用いて記述してください。

ただし l の要素数は奇数個となります。
ただし、双方向リンクリストとは、ノードの鎖構造であり、
各ノードは next, prev という二つのプロパティをもちます。
先頭のノードのみ prev = null, 末尾のノードのみ next = null となります。

また、片方向リンクリストだとどうなりますか？
型方向リンクリストとは各ノードが next プロパティのみを持ちます。