Assignment 2. **Running time estimates for simple algorithms and sorting.**

Given January 28, due February 10.

1. What is the asymptotic running time of this program to find all the prime numbers less than n.

```
p = 1
for i = 1, ... , n {
   for j = 2, ... , sqrt(i) {
      if ( ( i / j ) * j == i ) break;  // i is not prime
      p_list(p) = i;
      p = p + 1;
   }
}
```

2. What is the asymptotic time of the following algorithm:

```
for k = 1, ... , n {
   for j = 1, ... , k {
      a(j) = rand();
   }
   mergesort(a,k,temp)0
   sum = 0;
   for j = 1, ... , k-1 {
      sum = sum + ( a(j+1) - a(j) ) *  ( a(j+1) - a(j) );
   }
}
cout << " the sum is " << sum << endl; // Never mind this line.
```

The routine `rand()` produces a different random number in between 0 and 1 each time it is called.

3. We have a sorted list of real n numbers, $a(1) > a(2) > \cdots > a(n)$. For any number, x, $h(x)$ is the number of elements of a larger than x. Suggest an algorithm that will find the largest element of a that is smaller than x in time of order $\log(h(x))$.

4. We have a sorted list of n real numbers, $a(1) > a(2) > \cdots > a(n)$. Give an $O(n)$ algorithm that determines whether $a(j) = -a(i)$ for any pair i, j.

5. It is known that there is no algorithm that can sort n numbers using fewer than $O(n\log(n))$ comparisons. In view of this fact, is it possible to perform the `delete_max` operation on a heap using $O(1)$ comparisons?