You will submit your solution to this assignment to the Curator System (as `HW1`). Your solution must be either a plain text file (e.g., NotePad) or a MS Word document; submissions in other formats will not be graded.

Except as noted, credit will only be given if you show relevant work.

---

1. [20 points] Using the rules given in the course notes, perform an exact count complexity analysis, for the worst case, of the body of the following function. (Take `list.length` to be N.)

```
int part(int[] list, int barrierIdx) {

   int barrier, maxIdx, temp;

   barrier = list[barrierIdx];                // 1  2
   maxIdx  = list.length - 1;                 // 2  2 or 3

   temp = list[barrierIdx];                   // 3  2
   list[barrierIdx] = list[maxIdx];           // 4  3
   list[maxIdx] = temp;                       // 5  2

   barrierIdx = 0;                            // 6  1

   for (int i = 0; i < maxIdx; i++) {         // 7  1 before; 2 per pass;
                                              //    1 to exit loop
      if ( list[i] < barrier ) {              // 8  2
         temp = list[barrierIdx];             // 9  2
         list[barrierIdx] = list[i];          //10  3
         list[i] = temp;                      //11  2
         barrierIdx++;                        //12  1
      }
   }
   temp = list[maxIdx];                       //13  2
   list[maxIdx] = list[barrierIdx];           //14  3
   list[barrierIdx] = temp;                   //15  2

   return barrierIdx;                         //16  1
}
```

$$T(N) = 2+3+2+3+2+1+1+\sum_{i=1}^{N-1}\left(1+2+2+3+2+1+1\right)+1+2+3+2+1$$

$$= \sum_{i=1}^{N-1} 12 + 23$$

$$= 12(N-1)+23$$

$$= 12N+11$$

2. [40 points] For each part, determine the simplest possible function g(n) such that the given function is $\Theta(g)$. No justification is necessary.

a) $a(n) = 3n^2 + 14n + 47$ n^2

b) $b(n) = 14n + 3n\log n$ $n\log n$

Hint: the last three take a little analysis.

c) $c(n) = n^{0.9} + \log n$ $n^{0.9}$

d) $d(n) = 3n^2 \log n + n^3$ n^3

e) $e(n) = 3n\log^2 n + 3n^2 \log n$ $n^2 \log n$

---

3. [20 points] Suppose that f and g are non-negative functions such that f is Θ(g). Is it necessarily true that:

$$2^{f(n)} \text{ is } \Theta\left(2^{g(n)}\right)$$

If so, prove it. (You may assume that the limit referred to in Theorem 8 exists.) If no, give a specific counter-example and show that it is a counter-example.

The assumption the limit exists does not result in a proof, since:

$$\lim_{n\to\infty} \frac{2^{f(n)}}{2^{g(n)}} = \lim_{n\to\infty} 2^{f(n)-g(n)}$$

The assumption that f is Θ(g) may seem to tell us anything useful here, since it would at most imply something about the limit of the ratio of f to g, not their difference.

That observation, though, suggests that there may be a simple counter-example. Does the fact that the limit of f/g is finite really imply that the limit of f-g is finite? Consider a simple example, say:

$$f(n) = n \text{ and } g(n) = 2n$$

Now, f(n)/g(n) = 2, so the limit of the ratio is clearly 2 and so f is clearly Θ(g). But, on the other hand, f(n) – g(n) is n, and the limit of n isn't finite. That is:

$$\lim_{n\to\infty} \frac{2^n}{2^{2n}} = \lim_{n\to\infty} \frac{1}{2^n} = 0$$

and so 2^f is clearly not $\Theta(2^g)$.

4. [20 points] Suppose that executing an algorithm on input of size N requires executing T(N) = N log N + 8N instructions. How long would it take to execute this algorithm on hardware capable of carrying out 2^{24} instructions per second if N = 2^{32}? (Give your answer in hours, minutes and seconds, to the nearest second.)

The number of instructions to be executed would be T(2^32):

$$T(2^{32}) = 2^{32} \log 2^{32} + 8 \cdot 2^{32} = 32 \cdot 2^{32} + 8 \cdot 2^{32} = 40 \cdot 2^{32}$$

So the time required (in seconds) would be:

$$\text{Time} = \frac{40 \cdot 2^{32}}{2^{24}} = 40 \cdot 2^{8} = 10240$$

Converting to HH:MM:SS gives us 2:50:40 (2 hours, 50 minutes, 40 seconds).