You will submit your solution to this assignment to the Curator System (as `HW2`). 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) {

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

(by Theorem 13)

b) $b(n) = 14n^2 + 3n\log n$ n^2

(by Theorem 13)

Hint: the last three take a little analysis.

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

Just apply Theorem 8 (guessing one way or the other):

$$\lim_{n\to\infty} \frac{n^{0.9} + \log n}{n^{0.9}} = \lim_{n\to\infty}\left(1 + \frac{\log n}{n^{0.9}}\right) = 1 + \lim_{n\to\infty} \frac{1/n\ln 2}{0.9n^{-0.1}} = 1 + \lim_{n\to\infty} \frac{\ln 2}{0.9n^{0.9}} = 1 + 0 = 1$$

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

$$\lim_{n\to\infty} \frac{3n^2\log n + n^3}{n^3} = \lim_{n\to\infty}\left(\frac{3\log n}{n} + 1\right) = 1 + \lim_{n\to\infty} \frac{3/n\ln 2}{1} = 1 + \lim_{n\to\infty} \frac{3}{n\ln 2} = 1 + 0 = 1$$

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

$$\lim_{n\to\infty} \frac{3n\log^2 n + 3n^2\log n}{n^2\log n} = \lim_{n\to\infty}\left(\frac{3\log n}{n} + 3\right) = 3 + \lim_{n\to\infty} \frac{3/n\ln 2}{1} = 3 + \lim_{n\to\infty} \frac{3}{n\ln 2} = 3 + 0 = 3$$

3. [20 points] An equivalent definition of Θ is:

> Suppose that $f(n)$ and $g(n)$ are non-negative functions of N.
> Then $f(n)$ is $\Theta(g(n))$ if there exist positive constants C_1, C_2 and N
> such that, for all $n \geq N$, $C_1 g(n) \leq f(n) \leq C_2 g(n)$.

Use the alternate definition given above to prove the following statement:

> Suppose that $f(n)$, $g(n)$, r(n) and $s(n)$ are non-negative functions of N,
> such that $f(n)$ is $\Theta(r(n))$ and $g(n)$ is $\Theta(s(n))$.
> Then the function $f(n) + g(n)$ is $\Theta(r(n) + s(n))$.

Suppose that $f(n)$, $g(n)$, r(n) and $s(n)$ are non-negative functions of N, such that $f(n)$ is $\Theta(r(n))$ and $g(n)$ is $\Theta(s(n))$.
Then by the alternate definition, there are constants L_1, U_1 and N_1 such that for all $n \geq N_1$, $L_1 r(n) \leq f(n) \leq U_1 r(n)$.
And, there are constants L_2, U_2 and N_2 such that for all $n \geq N_2$, $L_2 s(n) \leq g(n) \leq U_2 s(n)$.
So, let $L = \min(L_1, L_2)$ and let $U = \max(U_1, U_2)$ and let $N = \max(N_1, N_2)$, then for all $n \geq N$, $L(r(n) + s(n)) \leq f(n) + g(n) \leq U(r(n) + s(n))$.
Therefore, the function $f(n) + g(n)$ is $\Theta(r(n) + s(n))$.

4. [20 points] Suppose that executing an algorithm on input of size N requires executing T(N) = N log N + 16N instructions. How long would it take to execute this algorithm on hardware capable of carrying out 2^{24} instructions per second if N = 2^{30}? (Give your answer in hours, minutes and seconds, to the nearest second.)

The total number of instructions needed is given by T(N), which would be:

$$T\left(2^{30}\right) = 2^{30}\log 2^{30} + 16\cdot 2^{30} = 30\cdot 2^{30} + 16\cdot 2^{30} = 46\cdot 2^{30}$$

So, the total time required (in seconds) would be T(N)/(execution rate):

$$\frac{T\left(2^{30}\right)}{2^{24}} = \frac{46\cdot 2^{30}}{2^{24}} = 46\cdot 2^{6} = 2944$$

Converting, we get a final result of 49 minutes, 4 seconds.