You will submit your solution to this assignment to the Curator System (as `HW02`). Your solution must be either a plain text file (e.g., NotePad) or a typed 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. [15 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.

```
public double eval(double[] c, double x) {

   double polyx = c[0];
   double xToK  = x;
   for (int k = 1; k < c.length; k++) {
      polyx = polyx + c[k] * xToK;
      xToK = x * xToK;
   }
   return polyx;
}
```

State both a precise complexity function T(N) and the Θ-complexity of T(N).

---

2. [15 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.

```
public double eval(double[] c, double x) {

   double polyx = c[0];
   for (int k = 1; k < c.length; k++) {
      double xToK = x;
      for (int i = 1; i < k; i++) {
         xToK = x * xToK;
      }
      polyx = polyx + c[k] * xToK;
   }
   return polyx;
}
```

State both a precise complexity function T(N) and the Θ-complexity of T(N).

---

3. [20 points] For each part, determine the simplest possible function g(n) such that the given function is $\Theta(g)$. No justification is necessary, but you might have to do some analysis using the theorems from the notes.

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

b) $b(n) = 3n \log n + 5n$

c) $c(n) = 3n \log\left(n^2\right) + 3n^2 \log n$

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

e) $e(n) = \dfrac{n^2 + 2n + 3}{n^2}$

4. [15 points] Suppose that executing an algorithm on input of size N requires executing T(N) = 8N + log N instructions. How long would it take to execute this algorithm on hardware capable of carrying out 2^{28} instructions per second if N = 2^{40}? (Give your answer in hours, minutes and seconds, to the nearest second.)

---

5. [25 points] Design an efficient algorithm for solving the following problem:

> Given an array `A` holding `N` elements, such that `A[0] < A[1] < A[2] < . . . < A[N-1]`, determine whether there is an index `k` such that `0 <= k <= N-1` and `A[k] = k`.

Write your algorithm as a Java function and state its Θ-complexity.

---

6. [10 points] Prove the following:

$$\text{if } x \text{ is a real number then } \lfloor x \rfloor + 1 = \lfloor x + 1 \rfloor$$