CSci 1302 Assignment 9

Due Mon., November 17, 2003

Problem 1 (6 points). Prove the following statements:

1. $n^3 + 1 \in O(n^3 - 1)$
2. $n^3 - 1 \in O(n^3 + 1)$
3. $20n^2 + 100 \in O(n^3 + 1)$

Problem 2 (6 points)

Consider the following functions:

1. $0.001n^2 + 1000n$
2. $5n^2 + 0.5n \log_2 n$
3. $\log_2 n + 70$
4. $n \log_2 n + 5$

For each of the functions above please answer the following questions:

1. Is the function in the class $O(n)$?
2. Is the function in $O(n^2)$?
3. Is the function in $\Theta(n^2)$?

You don't need to prove your answers, but explain your reasoning briefly.

Problem 3 (6 points) Consider the following algorithm for reversing the order of elements in an array (written in a Java-like syntax):

```
// Given: an array a[n] of n integer elements
// Result: the elements of the array are reversed.
// The index of the first element is 1

int i = 1;
int j = n;
while (i < j) {
   // switching the i-th and the j-th elements
   int temp = a[i];
   a[i] = a[j];
   a[j] = temp;
   i = i + 1;
   j = j -1;
}
```

Show how the algorithm works on the array `[1, 2, 3, 4]`. Compute the run-time of the algorithm based on the number of assignment statements (note that there are 3 assignments in each run of the loop). Give the "Big-O" approximation for the run-time (i.e. $O(n), O(n^2)$, etc.). Show your computations.

Problem 4 (6 points) Consider another algorithm for reversing elements of an array. This algorithm is recursive, so we write it as a recursive function. You may assume that the number of elements in the array is a power of 2. We use the notation `a[i..j]` to denote a portion of the array from index i to index j.

The idea of the algorithm is to divide the array in half, reverse each sub-array recursively, and then switch the two halves.

```
// Given: an array a[n] of n integer elements
// Result: the elements of the array are reversed.
// The index of the first element is 1

void reverse (a[1..n]) {
   if (n < = 1) return; // do nothing
   else {
        reverse (a[1..n/2]);
        reverse (a[((n/2) + 1)..n]);
        // switch the two halves:
        int i = 1;
        while (i <= n/2) {
           int temp = a[i];
           a[i] = a[(n/2) + i];
           a[(n/2) + i] = a[i];
        }
   }
}
```

Show how the algorithm works on the array `[1, 2, 3, 4]`. What is the running time of the algorithm? What is the "Big-O" approximation for the run-time? Show your computations.

Problem 5 (2 points) Which of the algorithms in the previous two problems would you recommend for reversing an array? Explain your reasoning.