CS 477/677 Analysis of Algorithms

Spring 2024

Homework 3

Due date: February 20, 2024

1. (U&G-required) [30 points] Consider the following algorithm.

```
ALGORITHM Enigma(A[1..n])
//Input: An array A[1..n] of n integer numbers
                                                  Cost
for i ← 1 to n − 1 do                             .........
     temp = 0                                     .........
     for j ← i + 1 to n do                        .........
          if A[i] == A[j]                         .........
               temp ← temp + 1                    .........
     print temp                                   .........
```

a) [15 points] What does this algorithm do?

b) [25 points] Compute the precise formula for the running time of this algorithm by indicating the costs for each line (i.e., how many times each line is executed), then give the order of growth of the total running time of the algorithm.

2. (U & G-required) [20 points]

a) [10 points] Show the steps taken by `Mergesort` to sort the array `A = [ M, E, R, G, I, N, G]` in alphabetical order.

b) [10 points] Show the steps taken by the `Partition` algorithm on the input `A = [ M, E, R, G, I, N, G]`. Use the following table to illustrate the algorithm's progress:

CALL	**p**	**r**	**q**	**Array**						
				1	2	3	4	5	6	7
PARTITION	1	7	…	M	E	R	G	I	N	G
…										

3. (U & G-required) [20 points]

For the `Insertion Sort` algorithm discussed in class, show how it runs on the input array `A = [4, 5, 3, 9, 2, 7]` and compute the exact number of comparisons and exchanges that it performs. Use the following table format to illustrate the algorithm's progress:

j	**i**	**Key**	**Array**						**Comparisons**	**Exchanges**
			4	5	3	9	2	7		
2	…	…								
…										
									Total =	**Total =**

4. (U & G-required) [30 points]

a) [20 points] Write pseudocode for a **divide-and-conquer algorithm** `Get_Min_Idx` that returns the **index of the smallest element** of the array.

b) [10 points] Write the recurrence for the running time of your algorithm (do not solve the recurrence).

5. (G-Required) [20 points] Use a loop invariant to prove that the following algorithm computes the geometric series $\sum_{k=0}^{n} x^k$, where `x` and `n` are natural numbers:

```
ALGORITHM GeometricSeries(x, n)
//Input: x, n integer numbers
{
    geomSeries ← 0
    pow ← 1
    for i ← 0 to n do
    {
        geomSeries ← geomSeries + pow
        pow ← pow * x
    }
    return geomSeries
}
```

Extra credit:

6. [20 points] Consider the generic algorithm `SolveP` given below, which solves a problem *P* by finding the output (solution) *O* that corresponds to an input *I*.

```
ALGORITHM SolveP(input I, output& O)
 // Input: I (of size n) for problem P
 // Output: O, the solution to problem P
 if size (I) == 1
   compute solution O to basic problem directly
 else
   partition I into 5 inputs I1, I2, …, I5, of size n/3 each
 for j ← 1 to 5 do
   SolveP (Ij, Oj)
 Combine O1, O2, …, O5 to get solution O for P with input I
```

Assume that the algorithm performs $g(n)$ operations for partitioning and combining and no basic operations for an instance of size 1. Write and solve the recurrence equation $T(n)$ for the number of basic operations needed to solve *P* when the input size is *n* and $g(n) = n\lg n$.