CSC 505, Homework 1

Due date: Monday, September 9, 9:00 PM

Homework should be submitted via Moodle in printed form. To avoid reduced marks, please do **NOT submit scanned writing or drawings**. All assignments are due on 9 PM of the due date. Late homework will be accepted only in circumstances that are grounds for excused absence under university policy. The university provides mechanisms for documenting such reasons (severe illness, death in the family, etc.). Arrangements for turning in late homework must be made by the day preceding the due date if possible.

All assignments for this course are intended to be individual work. Turning in an assignment which is not your own work is cheating. The Internet is not an allowed resource! Copying of text, code or other content from the Internet (or other sources) is plagiarism. Any tool/resource must be approved in advance by the instructor and identified and acknowledged clearly in any work turned in, anything else is plagiarism.

If an academic integrity violation occurs, the offending student(s) will be assessed a penalty that is at least as severe as getting a 0 for the whole homework for which the violation occurred, and the case will be reported to the Office of Student Conduct.

**Instructions about how to “give/describe” an algorithm** (taken from Erik Demaine): Try to be **concise, correct, and complete.** **To avoid deductions,** you should provide (1) a textual description of the algorithm, and, if helpful, flow charts and pseudocode; (2) at least one worked example or diagram to illustrate how your algorithm works; (3) a proof (or other indication) of the correctness of the algorithm; and (4) an analysis of the time complexity (and, if relevant, the space complexity) of the algorithm. **Remember that, above all else, your goal is to communicate.** If a grader cannot understand your solution, they cannot give you appropriate credit for it.

Here (and elsewhere), the function lg indicates the binary logarithm.

1. (12 points) *Purpose: Learn about bubblesort and practice how loop invariants are used to prove the correctness of an algorithm.* Please re-read Section 2.1 in our textbook and solve problem 2-2 on page 40 of the textbook.

2. (13 points) *Purpose: Practice counting basic operations and analyzing algorithms.* Assume *n* *>* 0 and consider the following algorithm.

1. *l* ← 1
2. *k* ← 0

(3) **for** *i* ← 1 **to** 2*n* **do**

(4) *l*← *2 \* l* \* *i*  
(5) **if** *k* < *n* **do**

(6) *k* ← 3 \* *i \* i + 5*

(7) **return(** *k, l* **)**

a) (5 points) For *n* = 1,2,3,4,5 what values for *k* and *l* are returned in line 7? How many multiplications (“\*”) does the algorithm perform for computing these values? How many additions (“+”) does the algorithm perform for computing these values?

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *N* | 1 | 2 | 3 | 4 | 5 |
| Return value (*k*) |  |  |  |  |  |
| Return value (*l*) |  |  |  |  |  |
| #multiplications(“\*”) |  |  |  |  |  |
| # additions(“+”) |  |  |  |  |  |

b) (2 points) As a function of *n*, what is the value of *k* returned in line 7? Justify your results.

c) (2 points) As a function of *n*, what is the value of *l* returned in line 7? Justify your results.

d) (2 points) As a function of *n*, how many multiplications (“\*”) does the algorithm perform? Justify your results.

d) (2 points) As a function of *n*, how many additions (“+”) does the algorithm perform? Justify your results.

3. (20 points, 8 for the correct order and 12 for the justification) *Purpose: Practice working with asymptotic notation.* Rank the following functions by order of asymptotic growth; that is, find an arrangement *g*1, *g*2, … of the below functions with *g*1∈Ω(g2), *g*2∈Ω(*g*3) …. Mark the functions that are asymptotically equivalent, i.e. *g*k∈Θ(*g*k+1) by a “\*”. **Justify your solution.** In this question you don’t have to refer to the formal c,n0-definitions but you may use limit rules and all other tools and results that our textbook provides to compare the asymptotic behavior of functions.

*a)* 2*n+*2*n*, sqrt(2)*n*+20, 22*n*, 2*n*-20, 2*n*2+20/*n*, 3*n*+30, *n*lg(*n*!)

*b)* (*n*+2)!*,* lg(*n*1.9)*,* 1/*n,* lg(*n*), *n*2lg(*n*2)*, n*1.9lg(*n*4)*,* (*n*+1)!

4. (16 points) Prove or disprove rigorously (ie give values for c and n0 that will make your argument work and justify the values) using the formal definitions of Θ, ω (little-omega), and o (little-oh).

a) (4 points) *n*lg(*n*)+*n* ∈ o(*n*) (little-o).

b) (4 points) 3*n*2-sin(*n*) - *n*∈ Θ(*n*2).

c) (4 points) ∈ O(sqrt(*n*)).

d) (4 points) *n*lg(*n*)+*n*2∈ ω(*n*) (little-omega)

5. (8 points) *Purpose: Practice how to design, analyze, and communicate algorithms.* Describe an **non-recursive Θ(lg *n*) algorithm** which computes (3*a)n*/2, given *a* and *n*. You may assume that *a* is a positive real number, and *n* a positive integer, but do **not** assume that *n* is a power of 2. **Please follow the above instructions for describing your algorithm.** Please give both, a textual description and pseudocode of your algorithm, and make sure that you justify the asymptotic running time of your algorithm.