210CT Week 2 Coursework Tasks
Dr. Diana Hintea

LEARNING OUTCOMES

1. Understand what pseudocode is and the importance of adopting it.
2. Learn the rules of writing pseudocode and remember to be consistent.
3. Reason about the complexity of algorithms and apply the BigO notation in doing so.

BASIC/INTERMEDIATE TASKS

1. Write the pseudocode for a function which returns the highest perfect square which is less or equal to its parameter (a positive integer). Implement this in the programming language of your choice.
2. Look back at last week's tasks. Describe the run-time bounds of these algorithms using the BigO notation.
3. Write the pseudocode corresponding to functions for addition, subtraction and multiplication of two matrices, and then compute A = B*C-2*(B+C), where B and C are two matrices of order N. What is the run-time?

ADVANCED TASK

1. Given two strings of n and m integer elements, write the pseudocode to compute:
   a) The string that contains all the elements belonging to both strings.
   b) The string of all the elements of the two given strings, written once.
   c) The string of the elements from the first string, without the elements that are also in the second string.
   What's the run time for each of the sub-sections above?
2. Write the pseudocode for a recursive program to generate the Cartesian product (product set, direct product, cross product) of N sets.
3. Write a function that takes four parameters representing the constant and multiplier of two linearly growing (as in O(m × n + k)) functions and determines the critical value of n (which should be an integer) at which the relative run-time of the two algorithms switches. That is, at which input size is algorithm A slower than B and at which is B slower than A? Use an iterative approach rather than solving the equations.

READING

Fortnow, L. and Homer, S. (2002). A Short History of Computational Complexity. *The History of Mathematical Logic, North-Holland.*