CSCI 1900 - Homework 16 - B

( 44 points )

# Section 3.2, 3.3: Growth of Functions; Complexity of Functions *(44)*

1. If *f* m( n ) = n mod m , evaluate the following (5)
	1. *f*10 (127)
	2. *f*2 (127)
	3. *f*8 (127)
	4. *f*7 (221)
	5. *f*0 (221)
2. Recall the membership function from Lecture 4. Given the following sets
U = { a, b, c, … , y, z } A = { p, d, q } B = { r, s, v, p} evaluate the following (3)
	1. *f* B ( s )
	2. *f* A ( s )
3. Evaluate each of the following: (8)

a)  b)  c)  d) 

e)  f)  g)  h) 

1. If , evaluate the following (3)
2. If , evaluate the following (4)
	1.
3. If , evaluate the following (4)
4. Suppose that is a function that describes the number of steps to execute an algorithm, where *n* is the number of items to be processed. For each of the following, describe the effect on the number of steps, if the number of items is doubled. (5)
5. Show that is in (1)
6. Show that is in (1)
7. Given the following ordering of functions.
1 < \_\_\_\_\_\_\_ < < \_\_\_\_\_\_\_ < < \_\_\_\_\_\_\_ < < \_\_\_\_\_\_\_
insert the following functions into their proper position to preserve - class ordering. (4)
 , , ,

Analyze the following algorithms. According to the following procedure:

a) Using the operation provide, give a function that describe the number of steps required to execute the algorithm,

b) Give the - class of the function from step a)

1. Algorithm: *Finding the largest value in an array* (Count the number of saves) (2)

function findMax ( theArray[], n )

theMax = theArray[0]

i = 1

while i < n

 if theArray[i] > theMax

 theMax = theArray[i]

 i = i+1

return theMax

1. Algorithm: *Creating an Identity matrix of size n*  (Count the number of compares) (2)

function createIdentity ( theArray[][], n )

j = 0

while j < n

k = 0

while k < n

 if j = k

 theArray[j][k] = 1

 else

 theArray[j][k] = 0

 k = k+1

j = j+1

1. Suppose that you know that an array is in sort-order. How many steps are now required to find the largest value? What is the - class? (2)