# MET CS 566 - Analysis of Algorithms
# Midterm Assignment - 20 Points

Please note the honor code policy regarding this assignment. You may not discuss particular questions or discuss or transmit answers from the assignment with other people except for the MET CS 566 teaching team.

## Tasks

**1. Ordering by asymptotic growth rates (4 points)**: Rank the following functions by order of growth. This means finding an arrangement $g_1, g_2, \dots, g_9$ of the functions that satisfy $g_1 = \Omega(g_2), g_2 = \Omega(g_3), \dots g_7 = \Omega(g_8)$

1. $g_1 = \log(\log(n))$
2. $g_2 = 2^{\log(n)}$
3. $g_3 = 2^{\sqrt{2 \times \log(n)}}$
4. $g_4 = n^{8.2} + \log(n)$
5. $g_5 = n^{8.2} + \mathrm{n}!$
6. $g_6 = n^{2024} + n^{512} + \mathrm{n} \times \log(n)$
7. $g_7 = e^{\mathrm{n}} + e^{\ln(n)}$
8. $g_8 = \sqrt{\log(n)}$

Tip: Work in stepwise format. Find the fastest growing function first, and then the next fast one

2. **Asymptotic Notations (6 points)** For each pair of the following functions $f(n)$ and $g(n)$, check if

1. $f(n) = \theta(g(n))$?
2. $f(n) = O(g(n))$?
3. $f(n) = \Omega(g(n))$?
4. $f(n) = o(g(n))$? Little o?
5. $f(n) = \omega(g(n))$? Little omega?

Functions $f(n)$ and $g(n)$ are:

1. $f(n) = (128)^{(\frac{n}{4})}$ $vs.$ $g(n) = (512)^{(\frac{n}{8})}$
2. $f(n) = n^{64} \times 2^n$ $vs.$ $g(n) = \log(n) \times 2^n + 4^n + n^{32}$
3. $f(n) = n^{1024}$ $vs.$ $g(n) = 2^{\log(n) \times \log(n)}$

You do not need to provide a formal proof. Describe your justifications in text form.

3. **Big O notation (2 points)**: What is the running time of this code in Big O notation regarding $n$?

**Algorithm 1** What is the running time of this code in Big O notation regarding $n$?

```
$y = 0$
$j = 1$
while $(j * j \leq n)$ do
    $y = y + 1$                                  ▷ Note: line 4 is some constrant cost
    $j = j + 1$
end while
```

4. **Big O notation (2 points)**: What is the running time of this code in Big O notation regarding $n$?

**Algorithm 2** What is the running time of this code in Big O notation regarding $n$?

```
$a = 0$
$k = n \times n$
while $(k > 1)$ do
    for $j \leftarrow 0$ to $n^2$ do
        $a = a + j$
    end for
    $k = k/2$
end while
```

**5. Recursive Function (2 points)**: What is the running time of the following python code in Big O notation regarding $n$?

```python
def isThis(n):
  print(n)
  if (n == 1):
    return True
  if ( n == 0) :
    return False
  if ((n % 3) == 0):
    return (isThis(n / 3))
  else:
    return False
```

Listing 1: What is the running time of this code in Big O notation regarding $n$?

6. **HEAP-EXTRACT-MAX worst case** **(1 point):**

Describe when the worst case of HEAP-EXTRACT-MAX happens after you removed a max value from a heap. Provide an example

7. **Hash Function** **(1 point):** Name two properties a good hash function should have.

Describe your answer (If possible, with an example)

8. **Double Hashing** **(2 points):** When you use double hashing in a hash table, does the number of potential collisions depend on the sequence of key insertions into the table?

In other words, if you change the sequence of insertions, would the number of collisions change? Support your answer using an example.