University of Nevada, Las Vegas Computer Science 477/677 Fall 2021

Assignment 2: Due Wednesday September 8, 2021 11:59 pm

Name:____

You are permitted to work in groups, get help from others, read books, and use the internet. Turn in the completed assignment on canvas, using instructions given to you by the grader, Mr. Heerdt, by 11:59 PM September 6.

1. Each of these code fragments takes if $O(n \log n)$.time, but not necessarily $\Theta(n \log n)$. Give the asymptotic complexity of each in terms of n, using Θ in each case.

(a)
```
for(int i = 1; i < n; i++)
 for(int j = 1; j < i; j = 2*j);
  cout << "Hello" << endl;
```

(b)
```
for(int i = 1; i < n; i++)
 for(int j = i; j < n; j = 2*j);
  cout << "Hello" << endl;
```

(c)
```
for(int i = 1; i < n; i=2*i)
 for(int j = 1; j < i; j++);
  cout << "Hello" << endl;
```

(d)
```
for(int i = 1; i < n; i=2*i)
 for(int j = i; j < n; j++);
   cout << "Hello" << endl;
```

(e)
```
for(int i = n; i > 1; i=i/2)
 for(int j = i; j > 1; j--);
  cout << "Hello" << endl;
```

(f)
```
for(int i = n; i > 1; i=i/2)
 for(int j = n; j > i; j--);
  cout << "Hello" << endl;
```

2. These problems are harder than the ones above. Given the asymptotic complexity of each fragment in terms of n, using Θ; except for the last problem.

(g)
```
for(int i = 1; i < n; i=2*i)
 for(int j = 1; j < i; j=2*j);
  cout << "Hello" << endl;
```

Hint: Use substitution. Let `m = log n, k = log i, l = log j.`

(h)
```
for(int i = 2; i < n; i=i*i)
 cout << "Hello" << endl;
```

Hint: Use substitution. Let `m = log n, k = log i.`

(i)
```
for(int i = 2; i < n; i=i*i)
  for(int j = 1; j < i; j = 2*j)
    cout << "Hello" << endl;
```
Hint: Use substitution. Let `m = log n, k = log i, l = log j.`

(j)
```
for(int i = n; i > 1; i = log i)
  cout << "Hello" << endl;
```
Hint: The answer is a function defined on page 136 of the textbook.

(k)
```
for(int i = 2; i < n; i = i*i)
  for(int j = 0; j < i; j++)
    cout << "Hello" << endl;
```
In my opinion, this is the hardest problem in this assignment. The time complexity of the code is O of one function of n and Ω of a different function of n, but is not Θ of any of the "usual" functions of n. Give both the O and the Ω answers, both of which are "usual" functions. [1]

3. Solve each of the following recurrences, giving the answer as Θ of a function of n.

(l) $F(n) = F(n/2) + n^2$

(m) $F(n) = F(n/3) + 1$

(n) $F(n) = 16F(n/4) + n^2$

(o) $F(n) = F(n-1) + n^5$

(p) $F(n) = F(n - \log n) + \log n$

(q) $F(n) = 16F(n/4) + n$

[1] By *usual functions* I mean all the functions we have discussed so far in class, which include polynomials, logarithms, iterated logarithms, powers of logarithms, roots, and even $\log^*$.