University of Nevada, Las Vegas Computer Science 477/677 Spring 2025

Answrs to Assignment 1: Due Saturday January 31, 2026 23:59:59

Follow instructions given by our Graduate Assistant Rakibul Hassan `hassar2@unlv.nevada.edu` on how to turn in the assignment.

If you find a mistake, you are **obligated** to let me know **immediately.**

Name:____

You are permitted to work in groups, get help from others, read books, and use the internet.

1. The sequence of powers of 2 is generated by the recurrence $2^n = 2 \cdot 2^{n-1}$.

   What is the recurrence which generates the Fibonacci sequence $F_1, F_2, \ldots$?

   $F(n) = F(n-2) + F(n-1)$

2. Write the sequence of all Fibonacci numbers under 100.

   $F_1 = 1$, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89

   You are permitted to start with zero, because $F_0 = 0$.

3. Which one of these statements is true?

   (a) The Fibonacci numbers increase logarithmically.

   (b) The Fibonacci numbers increase linearly.

   (c) The Fibonacci numbers increase quadratically.

   (d) The Fibonacci numbers increase exponentially.

   Exponentially.

4. Find the constant K such that $F_n = \Theta(K^n)$. What is the standard name of this constant?

   Assume that $F_n = K^n$. Solve the equation $K^n = K^{n-2} + K^{n-1}$, which reduces to $K^2 = 1 + K$ By the quadratic formula, and the fact that $K > 0$, we have $K = \frac{-1+\sqrt{5}}{2} \approx 1.6180339\ldots$, the golden ratio.

5. Work Exercise 1 on the handout complexity.pdf.

(xxvi) $\log 8 = 3$

(xxvii) $\log_4 \sqrt{2} = \frac{\log \sqrt{2}}{\log 4} = \frac{1/2}{2} = 1/4$

(xxviii) $\log_3 9 = 2$

(xxix) $\log_4 2 = 1/2$

(xxx) $\log(1/2) = -1$

(xxxi) $\log_3 \sqrt[3]{9} = \frac{\log_3 9}{3} = \frac{2}{3}$

(xxxii) $\log(\sqrt{8}-\sqrt{2}) = \log((2-1)\sqrt{2}) = \log(2-1) + \log(\sqrt{2}) = 0 + 1/2 = 1/2$

(xxxiii) $2^{\log_4 25}$ (Yes, it's an integer!)

$\log_4 25 = \dfrac{\log_2 25}{\log_2 4}$ Note that $25 = 5^2$.
$\log_2 25 = 2\log_2 5$ and $\log_2 4 = 2$, hence $\log_4 25 = \log_2 5$. Finally
$2^{\log_4 25} = 2^{\log_2 5} = 5$.

(xxxiv) $\log\log\log\log 65536$

Note that $2^{16} = 65536$, $2^4 = 16$, $2^2 = 4$, and $2^1 = 2$. Thus:
$\log\log\log\log 65536 = \log\log\log 16 = \log\log 4 = \log 2 = 1$

6. You've seen Landau notation. Originally, there was only "big O," but now there are several others. We will only use three of those this semester.

In each blank, write either Θ, O, or Ω, using the following rules.

- Write Θ it that is correct.
- If Θ is not correct, write either O or Ω, whichever is correct.

Hint: $\log n$ grows more slowly than any polynomially increasing function of n.

(i) $n\log n = \Omega(10n + \log(10n))$

(ii) $\log(n^2) = \Theta(\log(n^3))$

(iii) $10\log n = \Theta(\log(n^2))$

(iv) $n^{1.01} = \Omega(n\log^2 n)$

(v) $n^2/\log n = \Omega(n\log^2 n)$

This next one requires serious thinking. Don't just write down the first thing that occurs to you.

(vi) $n^{0.1} = \Omega(\log^2 n)$

The rest of these are harder, and may require calculation.

(vii) $\log n^{\log n} = O\left(\dfrac{n}{\log n}\right)$

(viii) $\sqrt{n} = \Omega(\log^3 n)$

The next one requires writing things down.

(ix) $n^{1/2} = O(5^{\log_2 n})$

Think before you write an answer to this next one.

(x) $n2^n = O(3^n)$

This next one is (slightly) tricky.

(xi) $2^n = \Theta(2^{n+1})$

This next one is easy, if you think about it correctly.

(xii) $n! = \Omega(2^n)$

The next one is perhaps the hardest one. I had to go over it several times, and could still be wrong!

(xiii) $\log n^{\log_2 n} = O(2^{(\log n)^2})$

If you know your calculus, the next one is easy.

(xiv) $\sum_{i=1}^{n} i^k = \Theta(n^{k+1})$

This next one is quite important for analyzing the time complexity of sorting algorithms, and it **will** appear on exams, and very likely during job inteviews.

(xv) $\log n! = \Theta(n \log n)$

7. The C++ code below implements a function, "mystery." What does it compute?

```
float mystery(float x, int k)
 {
  if (k == 0) return 1.0;
  else if(x == 0.0) return 0.0;
  else if (k < 0) return 1/mystery(x,-k);
  else if (k%2) return x*mystery(x,k-1);
  else return mystery(x*x,k/2);
 }
```

mystery$(x, k) = x^k$, except for the case that $x = 0.0$ and $k = 0$. The program returns 0.0, even though there actually is no correct answer, sort of like dividing by zero.

8. Consider the following C++ program.

```
void process(int n)
 {
  //cout << n << endl;
  if(n > 1) process(n/2);
  cout << n%2;
 }

int main()
 {
  int n;
  cout << "Enter a positive integer: ";
  cin >> n;
```

```
  assert(n > 0);
  process(n);
  cout << endl;
  return 1;
 }
```

The last line of the output of `process(n)` is a string of bits. What does this bitstring represent?

The output is the binary numeral for n.

9. Find the asymptotic time complexity of each code fragment, in terms of n. Use Θ notation.

   (i)
   ```
   for(int i = 0; i < n; i++)
     cout << "Hello";
   ```
   $\Theta(n)$

   (ii)
   ```
   for(int i = 1; i < n; i=2*i)
     cout << "Hello";
   ```
   $\Theta(\log n)$

   (iii)
   ```
   for(int i = 0; i < n; i++)
     for(int j = 1; j < i; j=2*j)
       cout << "Hello";
   ```
   $\Theta(n \log n)$