Fall 2023 CIS 3362 Homework #5 Grading Criteria

1) 5 pts – give full credit if correct, subtract 1 pt for each error, cap at 0.

2) 5 pts – give full credit if correct, subtract 1 pt for each error, cap at 0.

3) 5 pts – 1 pt stating Fermat, 2 pts to rewrite exponent, 2 pts to get to correct answer.

4) 5 pts – 2 pts phi calculation, 2 pts to rewrite exponent, 1 pt final answer.

5) 10 pts – 2 pts stating that k = 3 and m = 213 from step 1.
3 pts – correct calculation for $3^{213} = 368 \bmod 1705$ shown
3 pts - correct calculation for $368^2 = 729 \bmod 1705$
2 pts - correct calculation for $729^2 = 1186 \bmod 1705$ and concluding it's composite

6) 10 pts – 2 pts for starting chart at x = 496.
5 pts for the rows on the chart, ½ pt off round down if a row is incorrect
3 pts to use last row of chart to get a factorization (just 569 x 431 is needed)

7) 5 pts – 4 pts for proving that if gcd(k, p – 1) = 1, then α^k is also a primitive root
4 pt breakdown:
1 pt – starting with assume primitive root alpha (or whatever)
1 pts – considering raising alpha to a power k as described above
1 pts – looking at exponents of alpha
1 pts - completing proof by contradiction

1 pts for proving if gcd(k, p – 1) != 1, then α^k is not a primitive root.

Note: other ways to do this, so if you aren't sure let me grade it.

8) 5 pts - 2 pts for making the observation that each base is counted exactly once
In the sum on the left without proof.
3 pts for proving that if gcd(k, p – 1) = d, then the order of α^k is $\frac{p-1}{d}$.

Note: other ways to do this, so if you aren't sure let me grade it.

9) 50 pts – 5 pts for compiling and reading the right tokens from standard input.
5 pts – attempts to look for non-trivial divisors of p-1
10 pts – attempts to find the modular exponent of the generator to several terms
20 pts for small test cases (+1 for correct case -1 for incorrect case, cap at 0)
10 pts for large test cases (+1 for correct case -1 for incorrect case, cap at 0)
TIME LIMIT = 3 sec

Note: for Python use primroot2.in for large cases,
for C use primroot3.in for large cases (C overflows after 10^{18}, so max p ~ 10^9)