CSCI 2300 — Algo
Homework 2
Hayden Fuller

- **Q1**
  2^{2^n} in linear time, assuming multiplication of arbitrary size integers takes unit time. What is the bit-complexity if multiplications do not take unit time, but are a function of the bit-length.

  result = $(1 << (1 << n))$

  base:$(1 << (1 << 1)) = (1 << 2) = 4 = 2^{2^1} = 2^2 = 4$
  Bit shifting left is the same as multiplying by 2, so if you start with a 1, bit shifting left x times is the same as 2^x. This is simply done twice to get our result.
  We multiply by bit shifting, which takes unit time for any modern processor, making this equation run in unit time. If this were an older processer where bit shifting is O(n), this would be O(n^2).

- **Q2**
  $N!$

  (a) If N is an n-bit number, how many bits long is N! in O() notation
  $\lfloor \log_2((2^x - 1)!) \rfloor + 1$

  (b) Give an algorithm to compute N! and analyze its running time

  ```
  def fact(n):
  result = 1
  i in range(1,n+1): result = result * i
  return result
  ```

  O(n) loops
  each multiplication is O(n)
  totals to O(n^2)

- **Q3**
  Find the GCD of 1492 and 1776, using

  (a) the prime factorization method and using Euclid's method, and
  $1776/1492 = 1r284$
  $1492/284 = 5r72$
  $284/72 = 3r68$
  $72/68 = 1r4$
  $68/4 = 17r0$
  $\mathrm{GCD}(1492, 1776) = 17$

  (b) express the GCD as an integer linear combination of the two inputs.
  $ab = qr$
  $r = a - q * b$
  $4 = 72 - 1 * 68$

$68 = 284 - 3 * 72$
$72 = 1492 - 5 * 284$
$284 = 1776 - 1 * 1492$

$4 = 1 * 72 - 1 * 68$
$4 = 1 * 72 - 1 * (284 - 3 * 72)$
$4 = -1 * 284 + 4 * 72$
$4 = -1 * 284 + 4 * (1492 - 5 * 284)$
$4 = 4 * 1492 - 21 * 284$
$4 = 4 * 1492 - 21 * (1776 - 1 * 1492)$
$4 = -21 * 1776 + 25 * 1492$

$4 = -21 * 1776 + 25 * 1492$