York University EECS 3101Z Eric Ruppert January 10, 2025

Homework Assignment #1
Due: January 17, 2025 at 5:00 p.m.

- You will submit this assignment online via Crowdmark.
- Before starting the assignment, you must read the course policy on academic honesty on the course web page.
- You may work in pairs. Each pair should submit only one paper.
- If you discuss the assignment with anyone (other than your partner, if you have one), you should mention their names at the beginning of your submission. You should not take written notes away from such discussions, and you should write up your solutions yourselves.

[6] **1.** Put the following four functions in order so that f comes before g if f is $O(g)$. All logs have base 2.

$\sqrt{n}$ $\sqrt{\log n}$ $(\log \sqrt[4]{n})^2$ $\log \sqrt{n+5}$

Briefly explain why your answer is correct using the formal definition of big-O notation.

2. Consider the problem of computing m^n for natural numbers m, n.

[3] **(a)** Fill in the blanks in the following algorithm.

```
1:  Power($m, n$)
2:  |  Precondition: $m, n \in \mathbb{N}$ and $m > 0$
3:  |  $y \leftarrow m$
4:  |  $x \leftarrow n$
5:  |  $z \leftarrow$ ____________
6:  |  loop
7:  |  |  invariant: $y > 0$ and $m^n = z \cdot y^x$
8:  |  |  exit when ____________
9:  |  |  if $x$ is odd then $z \leftarrow$ ____________
10: |  |  $x \leftarrow \lfloor x/2 \rfloor$                    ▷ chop off rightmost bit of $x$
11: |  |  $y \leftarrow y \cdot y$
12: |  return $z$
13: |  Postcondition: returns $m^n$
```

Remark: $y > 0$ is included as part of the invariant so that we know y^x makes sense even if $x = 0$. (0^0 is undefined.)

[4] **(b)** Prove that the statement on line 7 is indeed a loop invariant.

[1] **(c)** Explain why the algorithm terminates.

[1] **(d)** Use the loop invariant to explain why the postcondition is satisfied when the loop terminates.

[1] **(e)** Assume the binary representations of m and n are at most ℓ bits long. Give a good upper bound (using big-O notation) on the number of bits of the largest integer ever stored in any of the variables. *Briefly* explain why your answer is correct.

[2] **(f)** Recall that we considered in class an algorithm that can multiply two k-bit numbers in $O(k^2)$ time. If we use that algorithm to perform the multiplications, give a good upper bound (using big-O notation) on the running time of the algorithm in part (a) in terms of ℓ.

How does this compare with a simple loop that multiplies n copies of m together?