Exercise Sheet 3
COMS10018 Algorithms

Reminder: $\log n$ denotes the binary logarithm, i.e., $\log n = \log_2 n$.

Example Question: Loop Invariants

Question. Prove that the stated invariant holds throughout the execution of the loop (using the Initialization, Maintenance, Termination approach discussed in the lectures):

Algorithm 1

```
Require: Array $A$ of length $n$ ($n \geq 2$)
1: $S \leftarrow A[0] - A[1]$
2: for $i \leftarrow 1 \dots n-2$ do
3:     $S \leftarrow S + A[i] - A[i+1]$
4: end for
5: return $S$
```

Invariant:

At the beginning of iteration i, the statement $S = A[0] - A[i]$ holds.

Which value is returned by the algorithm (use the Terminiation property for this)?

Solution. Let S_i be the value of S at the beginning of iteration i.

1. *Initialization* ($i = 1$): We need to show that the statement of the loop invariant holds for $i = 1$, i.e., the statement $S_1 = A[0] - A[1]$ holds before iteration $i = 1$. Observe that, in Line 1, S_1 is initialized as $S_1 \leftarrow A[0] - A[1]$. The loop invariant thus holds for $i = 1$.

2. *Maintenance*: Assume that the loop invariant holds for value i, i.e., $S_i = A[0] - A[i]$. We need to show that the loop invariant then also holds for value $i + 1$, i.e., we need to show that $S_{i+1} = A[0] - A[i + 1]$ holds. To this end, observe that in iteration i we execute the operation $S_{i+1} = S_i + A[i] - A[i + 1]$. Since $S_i = A[0] - A[i]$, we obtain $S_{i+1} = A[0] - A[i] + A[i] - A[i + 1] = A[0] - A[i + 1]$.

3. *Termination*: We have that, after the last iteration (or before the $(n-1)$th iteration that is never executed), $S_{n-1} = A[0] - A[n - 1]$ holds. The algorithm thus returns the value $A[0] - A[n - 1]$.

✓

1 Warm up: Proof by Induction

Consider the following sequence: $s_1 = 1, s_2 = 2, s_3 = 3$, and $s_n = s_{n-1} + s_{n-2} + s_{n-3}$, for every $n \geq 4$. Prove that the following holds:

$$s_n \leq 2^n \ .$$

2 Loop Invariant

Prove that the stated loop invariant holds throughout the execution of the loop (using the Initialization, Maintenance, Termination approach discussed in the lectures):

Algorithm 2

```
Require: Array $A$ of $n$ positive integers
 1: $B \leftarrow$ empty array of $n$ integers
 2: $B[0] \leftarrow A[0]$
 3: for $i = 1 \dots n-1$ do
 4:     if $A[i] > B[i-1]$ then
 5:         $B[i] \leftarrow A[i]$
 6:     else
 7:         $B[i] \leftarrow B[i-1]$
 8:     end if
 9: end for
10: return $B[n-1]$
```

Loop Invariant: At the beginning of iteration i, the following statement holds: For every $0 \leq j < i$: $B[j]$ is the maximum of the subarray $A[0, j]$, i.e., $B[j] = \max\{A[0], \dots, A[j]\}$.

Which value is returned by the algorithm (use the Terminiation property for this)?

Hint: The Maintenance part requires a case distinction in order to deal with the if-else statement.

3 Insertionsort

What is the runtime (in Θ-notation) of Insertionsort when executed on the following arrays of lengths n:

1. $1, 2, 3, 4, \dots, n-1, n$
2. $n, n-1, n-2, \dots, 2, 1$
3. The array A such that $A[i] = 1$ if $i \in \{1, 2, 4, 8, 16, \dots\}$ (i.e., when i is a power of two) and $A[i] = i$ otherwise.
4. The array B such that $B[i] = 1$ if $i \in \{10, 20, 30, 40 \dots\}$ (i.e., when i is a multiple of 10) and $B[i] = i$ otherwise.
5. The array C such that $C[i] = 1$ if $i \in \{n^{\frac{1}{10}}, 2 \cdot n^{\frac{1}{10}}, 3 \cdot n^{\frac{1}{10}}, \dots\}$ (i.e., when i is a multiple of $n^{\frac{1}{10}}$) and $C[i] = i$ otherwise. We assume here that $n^{\frac{1}{10}}$ is an integer.

4 Runtime Analysis

```
Algorithm 3
Require: Integer $n \geq 2$
  $x \leftarrow 0$
  $i \leftarrow n$
  while $i \geq 2$ do
    $j \leftarrow \lceil n^{1/4} \rceil \cdot i$
    while $j \geq i$ do
      $x \leftarrow x + 1$
      $j \leftarrow j - 10$
    end while
    $i \leftarrow \lfloor i/\sqrt{n} \rfloor$
  end while
  return $x$
```

Determine the runtime of Algorithm 3 in Θ-notation.

5 Optional and Difficult Questions

Exercises in this section are intentionally more difficult and are there to challenge yourself.

5.1 Proof by Induction

Let n be a positive number that is divisible by 23, i.e., $n = k \cdot 23$, for some interger $k \geq 1$. Let $x = \lfloor n/10 \rfloor$ and let $y = n \ \% \ 10$ (the rest of an integer division). Prove by induction on k that 23 divides $x + 7y$.

Example: Consider $k = 4$. Then $n = 92$, $x = 9$ and $y = 2$. Observe that the quantity $x + 7y = 9 + 7 \cdot 2 = 23$ is divisible by 23.

5.2 Average Case Analysis of Linear Search via Probability Theory

This exercise is included because it is interesting and, hopefully, enjoyable. However, it is not directly relevant to this unit's content, so feel free to skip it. It involves basic probability theory, which is not otherwise required for this unit.

Suppose that you repeatedly flip a fair coin (probability 1/2 each for heads/tails) until you see heads for the first time.

1. What is the expected number of times you need to flip this coin?
2. Argue why the analysis from the previous question can be used to determine the average case runtime of linear search on binary strings.