Loop Invariants

1. Consider the following algorithm to compute the factorial of a number n.

```
int FACTORIAL(int $n$)
    int $i$ = 1;
    int $f$ = 1;
    while ($i \leq n$)
        $f \leftarrow f \times i$;
        $i \leftarrow i + 1$;
    end while
    return ($f$);
end FACTORIAL
```

   (a) We will denote the values of i and f on the k^{th} iteration of the loop by i_k and f_k. What are the values of i_k, f_k, i_{k+1}, f_{k+1}.

   (b) Fill in the blanks below to define a loop invariant for FACTORIAL. Note that k represents the iteration number.

$$(LI)\ \ \forall k \in \mathbb{N}, (i_k = \_\_\_\_\_\_\_\_ \wedge f_k = \_\_\_\_\_\_\_\_ \wedge i_k \leq n) \rightarrow (i_{k+1} = \_\_\_\_\_\_\_\_ \wedge f_{k+1} = \_\_\_\_\_\_\_\_)$$

   (c) Write a carefully structured proof that the loop invariant (LI) is true.

2. Write a Java method `addBinary`, which takes two 1D `boolean` arrays of length n, called A and B. A and B represent two n-digit binary numbers, where:

   n = max{minimum number of digits needed to represent the first binary number,
   minimum number of digits needed to represent second the binary number}.

   `addBinary` adds the two numbers together, and returns the result in a `boolean` array (the length of the array should be equal to the minimum number of digits needed to represent the result) .

   If one of the numbers has fewer digits than the other, then you may assume that it is padded with 0s (i.e., there are `falses` in front of the first `true`). For example, you may assume that the arrays used to add 1101 and 10 would be:

```
boolean[] A = {true, true, false true};
boolean[] B = {false, false, true, false};
```

   Comment your code. Do not import any Java packages. You may assume that A and B are not `null`, and that A and B are both positive (non-zero) numbers.

3. Let `A` be a 1D array of n natural numbers, and consider the following pseudocode, with the precondition, postcondition and loop invariants inserted as comments:

```
//pre-condition: $\forall k \in \mathbb{N}, 0 \leq k < n \rightarrow 0 \leq \mathtt{A}[k]$
$j = 0$
// Outer loop invariant: $\forall s \in \mathbb{N}, 0 < j \leq s < n \rightarrow 0 \leq \mathtt{A}[0] \leq \mathtt{A}[1] \leq ... \leq \mathtt{A}[j-1] \leq \mathtt{A}[s]$
// Outer loop description: the first $j$ elements are sorted in ascending order,
// and the $j$th element is less than or equal to any element between the $j$+1st and the $n$th element.
while($j < n$){
  imin $= j$
  $k = j + 1$
  // Inner loop invariant: $\forall r \in \mathbb{N}, j \leq r < k \rightarrow \mathtt{A}[\mathtt{imin}] \leq \mathtt{A}[r]$
```

```
    // Inner loop desciption: finding smallest element between the $j$+1st and the $n$th element.
    while($k < n$){
       if(A[$k$] < A[imin]){
          imin = $k$
       }
       $k++$
    } //end of inner loop
    temp = A[$j$]
    A[$j$] = A[imin]
    A[imin] = temp
    $j++$
 } //end of outer loop
 //post-condition: $\forall j \in \mathbb{N}, \forall k \in \mathbb{N}, 0 \le j < k < n \rightarrow$ A[$j$] $\le$ A[$k$]
```

(a) Prove that the outer loop invariant is true at the start of the first iteration of the outer loop.

(b) Prove that if the outer loop invariant is true at the start of an iteration of the outer loop and the outer loop is executed, then the inner loop invariant is true at the start of the first iteration of the inner loop.

(c) Prove that if the inner loop invariant is true at the start of *any* iteration of the inner loop, then it is true at the end of that iteration.

(d) Prove that if the outer loop invariant is true when the outer loop terminates, then the post-condition is true.

4. Consider the following algorithm (the lines starting with "//" are comments that will not be executed).

```
// precondition: $a \in \mathbb{N}, b \in \mathbb{N}$
$m := a$
// loop invariant: $m \ge 0 \land \exists x \in \mathbb{N}, a = bx + m$
while $m \ge b$ do
    $m := m - b$
end while
// postcondition: $m = a \bmod b$, i.e., $0 \le m < b \land \exists x \in \mathbb{N}, a = bx + m$.
```

(a) Prove that the loop invariant is true before the first iteration of the loop.

(b) Let m' denote the value of m at the end of some iteration of the loop and assume that the loop invariant is true for m'. Furthermore, let x' denote the value that makes $a = bx' + m'$ true.

Prove that the loop invariant remains true at the end of the next iteration of the loop, if there is such an iteration (use m'' to denote the value of m at the end of the next iteration).

(c) Prove that the postcondition is true after the last iteration of the loop. (You may assume that the loop invariant is true at the end of every iteration of the loop.)

5. Examine the method `mult(m,n)` below. Prove or disprove that if the loop invariant is true at the beginning of a loop iteration, then it is true at the end of a loop iteration. Your proof should be in the structured proof format from class.

```
/**
 * mult returns the product of integers m and n
 * @param m an integer
 * @param n an integer
 * @return mn, the product of m and n
 * // precondition: m and n are integers
```

```
   * // postcondition: integer product mn is returned
   */
  public static int mult(int m, int n) {
    int x = m, y = n, z = 0;
    // loop condition: z = mn - xy
    while (x != 0) {
      if (x % 2 != 0) {
        if (x * m > 0) {
          z = z - y;
        }
        else {
          z = z + y;
        }
      }
      x = (int)(Math.floor(x / -2.0));
      y = y * 2;
    }
    // postcondition z = mn
    return z;
  }
}
```

Asymptotics

1. Working with logarithms and binary. Recall that $\log_a x = b$ means that $a^b = x$. Prove the following:
   (a) $\log_2 x = c \log_{10} x$ where $c \in \mathbb{R}^+$.
   (b) Suppose that $f(x) = \log_a x$ and $g(x) = \log_b x$. What can we infer from (a) about $f(x), g(x), \mathcal{O}, \Omega$ and Θ?.
   (c) Convert $(123)_{10}$ and $(177)_{10}$ to binary. Convert $(1101011)_2$ and $(1000011)_2$ to decimal.
   (d) Working only in binary, multiply $(1100)_2$ by (101_2). Suppose that you need to compute $9x$ where x is a binary number. Explain in words how you would transform the digits of x to get $9x$. Now let $x = 110110$ and compute $9x$.

2. From the definitions of $\mathcal{O}$ and Ω prove or disprove each of the following.
   (a) $6n^5 + n^2 - n^3 \in \mathcal{O}(n^5)$
   (b) $(1 + 1/n) \in \Omega(1/n)$
   (c) $nlogn \in \mathcal{O}(n^2)$
   (d) **Bonus Question**: Prove that $n^{n \log_2 n} \in \mathcal{O}((\log_2 n)^n)$ or $(\log_2 n)^n \in \mathcal{O}(n^{n \log_2 n})$.

3. Consider the following argument that $\sum_{k=1}^{n} kn \in \mathcal{O}(n^2)$. In other words,

$$\exists c \in \mathbb{R}^+, \exists b \in \mathbb{N}, \forall n \in \mathbb{N}, n \geq b \rightarrow \sum_{k=1}^{n} kn \leq cn^2$$

   Note that:

$$\begin{gathered} n \leq 1n \text{ therefore } n \in \mathcal{O}(n) \\ 2n \leq 2n \text{ therefore } 2n \in \mathcal{O}(n) \\ \vdots \\ kn \leq kn \text{ therefore } kn \in \mathcal{O}(n) \end{gathered}$$

   Therefore $\sum_{k=1}^{n} kn \in \underbrace{\mathcal{O}(n) + \mathcal{O}(n) + \ldots + \mathcal{O}(n)}_{n \text{ times}} = n \cdot \mathcal{O}(n) = \mathcal{O}(n^2)$.

   (a) What is wrong with this argument?
   (b) Determine $g(n)$ such that $\sum_{k=1}^{n} kn \in \mathcal{O}(g(n))$ and using a carefully structured proof prove that your $g(n)$ is correct.

4. Recall from calculus the definition of a limit $\lim_{n\rightarrow\infty} h(n) = a$. If the limit exists then

$$\textit{For } n \geq n_0 \textit{ there is a } \delta \textit{ such that } |h(n) - a| \leq \delta.$$

   An alternative definition for $f(n) \in \Theta(g(n))$ is the following:

$$f(n) \in \Theta(g(n)) \;\; if \;\; \lim_{n\rightarrow\infty} \frac{f(n)}{g(n)} \;\; = \;\; p \qquad (1)$$

   where p is some *positive* real number.

   (a) Using the definition of limit and the definition of *big-theta* notation, construct a carefully structured proof of (1).
   (b) If $p = 0$ what can we say about $f(n)$ and $g(n)$ and asymptotic bounds?

5. From the definitions of $\mathcal{O}$, Ω and Θ, prove or disprove each of the following using our carefully structured form:
   (a) $n! \in \Omega(n^n)$
   (b) $\frac{n+12}{2n^2+2} \in \mathcal{O}(1)$
   (c) $n^3 + 3n \in \Theta(2n^3)$

6. For any function $f : \mathbb{N} \to \mathbb{R}^{\geq 0}$, define the function $\bar{f} : \mathbb{N} \to \mathbb{R}^{\geq 0}$ as follows:

$$\forall n \in \mathbb{N}, \bar{f}(n) = \lceil f(n) \rceil$$

   For any function $f : \mathbb{N} \to \mathbb{R}^{\geq 0}$, define the function $\hat{f} : \mathbb{N} \to \mathbb{R}^{\geq 0}$ as follows:

$$\forall n \in \mathbb{N}, \hat{f}(n) = \lfloor f(n) \rfloor$$

   Prove or disprove each of the following using our carefully structured form:

   (a) $f \in \mathcal{O}(g) \leftrightarrow f \in \mathcal{O}(\bar{g})$
   (b) $\bar{f} \cdot \hat{f} \in \mathcal{O}(f^2)$
   (c) Consider $\mathcal{O}_2(g) = \{f : \mathbb{N} \to \mathbb{R}^{\geq 0} | \exists c \in R^+, \forall n \in \mathbb{N}, f(n) \leq cg(n)\}$.

      Prove or disprove: $f \in \mathcal{O}(g) \leftrightarrow \bar{f} \in \mathcal{O}_2(\bar{g})$.

7. Prove or disprove each statement below.
   (a) $3n^2 - 4n \in \Omega(n^2)$
   (b) $n \log n + 5n \in \mathcal{O}(n \log n)$
   (c) $200^{10^5 0} + n/4 \in \mathcal{O}(n)$
   (d) $5/n + 1 \in \mathcal{O}(1/n)$
   (e) $\forall a \in \mathbb{R}^+, \forall b \in \mathbb{R}^+, \log_a n \in \Theta(\log_b n)$

8. Prove or disprove the following statement:

      For any functions $f : \mathbb{N} \to \mathbb{R}^+$, $g : \mathbb{N} \to \mathbb{R}^+$, and $h : \mathbb{N} \to \mathbb{R}^+$,
      if $f \in \mathcal{O}(h)$ and $g \in \mathcal{O}(h)$, then $f \in \mathcal{O}(g)$.

9. Prove or disprove the following statement:

      For any functions $f : \mathbb{N} \to \mathbb{R}^+$, $g : \mathbb{N} \to \mathbb{R}^+$, and $h : \mathbb{N} \to \mathbb{R}^+$,
      if $f \in \Theta(h)$ and $g \in \Theta(h)$, then $f \in \Theta(g)$.

10. Prove that for any $k \in \mathbb{N}$ and any $a_k \in \mathbb{R}^+, a_{k-1} \in \mathbb{R}^+, \ldots, a_1 \in \mathbb{R}^+, a_0 \in \mathbb{R}^+$,

$$a_k n^k + a_{k-1} n^{k-1} + \cdots + a_1 n + a_0 \in \Theta(n^k).$$

11. Prove that for any $k \in \mathbb{N}$, $1^k + 2^k + \cdots + n^k \in \Theta(n^{k+1})$.

12. Let $f : \mathbb{N} \to \mathbb{R}^{\geq 0}$, and let

$$O(f) = \{g : \mathbb{N} \to \mathbb{R}^{\geq 0} | \exists c \in \mathbb{R}^{+}, \exists B \in \mathbb{N}, \forall n \in \mathbb{N}, n \geq B \to g(n) \geq cf(n)\}$$

Prove or disprove (in structured proof form) the following claims:

(a) Suppose $f, g : \mathbb{N} \to \mathbb{R}^{\geq 0}$. Then $g \in O(f) \Rightarrow g(n^3) \notin O(f)$

(b) Suppose $f, g, h : \mathbb{N} \to \mathbb{R}^{\geq 0}$. Then $(g \in O(f) \wedge h \in O(f)) \Rightarrow \max(g, h) \in O(f)$.

(c) Suppose $f, f', g, g' : \mathbb{N} \to \mathbb{R}^{\geq 0}$, and $f \circ g(n) = f(g(n))$, $f' \circ g'(n) = f'(g'(n))$. Then $(f \in O(f') \wedge g \in O(g')) \Rightarrow f \circ g \in O(f' \circ g')$.

(d) Suppose $f, g, h : \mathbb{N} \to \mathbb{R}^{\geq 0}$ and $gh(n) = g(n)h(n)$. Then $(g \in O(f) \wedge h \in O(f)) \to gh \in O(f)$.

(e) Suppose $f, g : \mathbb{N} \to \mathbb{R}^{\geq 0}$ and $f'(n) = \lfloor f(n) \rfloor$, $g'(n) = \lfloor g(n) \rfloor$. Then $g \in O(f) \to g' \in O(f')$.

Algorithm Complexity

1. Consider the MatrixMultiplication algorithm that multiplies two matrices. Let A and B be 2D arrays, which store the two matrices. The result of the matrix multiplication is stored in a 2D array C. In the code below, $rows$ and $cols$ are the lengths of the rows and columns of the array.

```
MatrixMultiplication($A, B$)
1.  if ($A$.cols! $= B$.rows)
2.      return null
3.  $i = 0$
4.  while ($i < A$.rows){
5.      $k = 0$
6.      while ($k < B$.cols){
7.          $C[i][k] = 0$
8.          $j = 0$
9.          while ($j < A$.cols){
10.             $temp = C[i][k]$
11.             $result = A[i][j] \cdot B[j][k]$
12.             $C[i][k] = temp + result$
13.             $j = j + 1$
            }
14.         $k = k + 1$
        }
15.     $i = i + 1$
    }
16. return $C$
```

(a) Compute the exact number of steps executed for each line of the algorithm.

(b) Compute $t_{MM}\left(\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 4 & 2 & 9 \end{bmatrix}, \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 1 \\ 2 & 4 & 6 & 8 & 9 \end{bmatrix}\right)$.

(c) For $n \times n$ arrays, A and B, find $T_{MM}(n)$.

(d) For $n \times n$ arrays, A and B, prove $T_{MM}(n) \in \mathcal{O}(n^3)$.

(e) For $n \times n$ arrays, A and B, prove $T_{MM}(n) \in \Omega(n^3)$.

(f) For $n \times n$ arrays, A and B, is there a natural number $r \in \mathbb{N}$ such that $T_{MM} \in \Theta(n^r)$? Justify your answer.

2. Consider the Selection Sort algorithm below, that sorts an input array A into non-decreasing order.

```
SS($A$):
1.      $i := 0$
2.      while $i < A$.length:
3.          $s := i$
4.          $j := i + 1$
5.          while $j < A$.length:
6.              if $A[j] < A[s]$:
7.                  $s := j$
8.              $j := j + 1$
9.          $t := A[i]$
10.         $A[i] := A[s]$
```

```
11.                 $A[s] := t$
12.                 $i := i + 1$
```

For each question below, find the *exact* answer without using $\mathcal{O}$, Ω, or Θ.

(a) Compute the number of steps executed for each line of the algorithm.

(b) Compute $t_{SS}([4, 2, 1, 3, 5])$.

(c) Compute $T_{SS}(3)$.

(d) Compute $T_{SS}(n)$.

3. Prove that $T_{\text{BFT}}(n) \in \Theta(n^2)$, where BFT is the algorithm below.

```
    BFT($E, n$):
1.      $i := n - 1$
2.      while $i > 0$:
3.          $P[i] := -1$
4.          $Q[i] := -1$
5.          $i := i - 1$
6.      $P[0] := n$
7.      $Q[0] := 0$
8.      $t := 0$
9.      $h := 0$
10.     while $h \leq t$:
11.         $i := 0$
12.         while $i < n$:
13.             if $E[Q[h]][i] \neq 0$ and $P[i] < 0$:
14.                 $P[i] := Q[h]$
15.                 $t := t + 1$
16.                 $Q[t] := i$
17.             $i := i + 1$
18.         $h := h + 1$
```

(Although this is not directly relevant to the question, this algorithm carries our a breadth-first traversal of the graph on n vertices whose adjacency matrix is stored in E.)

4. Your rich uncle gives you some money, so you decide to drop CSCA65 and travel around Europe for the summer. There's a list of n cities $c_1, \ldots, c_n$ that you want to visit, but you only have enough money for $n - 2$ train rides, so you're going to have to walk for one leg of the trip. Seeing as it's summer and it's hot outside, you want to walk for the shortest distance possible, so you want to find the two cities which are closest together. Assume you have a 2 dimensional $n \times n$ array $dist$ such that $dist(c_i, c_j)$ returns the distance between any two cities.

Here is an iterative algorithm for finding the closest pair:

```
Find_closest_pair (c_1, c_2, ..., c_n)          Steps:

1.  if (n == 1) return NIL;                      ____________
2.  min = infinity;                              ____________
3.  min_i = 0;                                   ____________
4.  min_j = 0;                                   ____________
5.  i = 1;                                       ____________
6.  while (i <= n)                               ____________
```

```
7.       j = i+1;                                  ____________
8.       while (j <= n)                            ____________
9.          if (dist(c_i, c_j) < min)              ____________
10.             min = dist(c_i, c_j);              ____________
11.             min_i = i;                         ____________
12.             min_j = j;                         ____________
13.         j = j+1;                               ____________
14       i = i+1;                                  ____________
15. return (min_i, min_j);                          ____________

end
```

(a) Fill in the number of steps for each line above, you may assume that accessing `dist` takes one step as in 1 dimensional arrays.

(b) Let $T_{CP}(n)$ represent the worst case complexity of `Find\_Closest\_Pair(c_1,c_2,c_3,c_4)`. Compute $T_{CP}(4)$ exactly.

(c) Determine a function g such that $T_{CP}(c_1, c_2, \ldots, c_n) \in \Theta(g(n))$ and prove that your g is correct.