Tutorial 1 Exercises

COMP526: Efficient Algorithms

7–8 October, 2024

Exercise 1. Suppose A, B, and C are logical propositions. Which of the following expressions are logically equivalent to one another?

1. $A \implies B$
2. $(A \implies C) \wedge (B \implies C)$
3. $B \implies A$
4. $(A \wedge C) \vee (B \wedge C)$
5. $((A \wedge C) \implies B) \wedge ((A \wedge \neg C) \implies B)$
6. $(A \vee C) \wedge (B \vee C)$
7. $\neg A \vee B$
8. $(A \wedge B) \vee C$
9. $(A \vee B) \wedge C$
10. $(A \vee B) \implies C$
11. $\neg(B \wedge \neg A)$

Exercise 2. Consider a society consisting of a set S of people. We say that a person $p \in S$ is a ***dictator*** if for every $q \in S$, q obeys p. We say that S is a ***dictatorship*** if S contains a dictator.

1. Write the condition of S being a dictatorship in logical notation using the quantifiers $\forall$ and $\exists$ and the predicate $P(p,q)$ indicating that p obeys q.
2. Negate your expression from part 1 to obtain an expression for S not being a dictatorship.
3. How can you interpret the expression you devised for part 2 in plain English?

Exercise 3. Consider the following SelectionSort algorithm.

```
procedure MinIndex(A, i, k)    ▷ Find and return the index of the minimum value in the array A between indices i and k, inclusive.
    m ← i
    for j = i, i+1, ..., k do
        if A[j] < A[m] then
            m ← j
        end if
    end for
return m
end procedure
procedure SelectionSort(A, n)    ▷ Sort the array A of size n
    for i = 1, 2, ..., n do
```

```
12:         j ← MinIndex(A, i, n)
13:         Swap(A, i, j)
14:     end for
15: end procedure
```

Assume that the procedure MinIndex(A, i, j) correctly returns the index of the minimum value stored in A between the indices i and j (inclusive). The procedure Swap(A, i, j) swaps the values of A at indices i and j. Prove that
SelectionSort correctly sorts every array A of size n. More specifically:

1. Identify a *loop invariant* that is satisfied at the end of each iteration of the loop in lines 11–14 of SelectionSort.

2. Use mathematical induction to argue that your loop invariant holds.

3. Conclude that after the final iteration, the array is sorted (i.e., $A[1] \leq A[2] \leq \cdots \leq A[n]$).