1 Evaluating Quantifications

Evaluate each of the quantifications below. State whether they are true or false, and explain why this is true.

You should assume that all variables are in the domain $\mathbb{Z}$ (that is, the integers) - although a few quantifications will also limit the domain to only some integers.

(a) $\forall x \forall y, xy \geq 0$

Solution: This quantification is false.
If $x < 0$ but $y > 0$, then the predicate would be false.

(b) $\forall x \exists y, xy \geq 0$

Solution: This quantification is true.
For any value of $x$, it is possible to select $y = -x$; the product of the two is positive (or zero).

(c) $\exists x \forall y, xy \geq 0$

Solution: This quantification is true.
If $x = 0$, then the predicate is true for all possible values of $y$.

(d) $\forall(n \geq 3) \exists(1 < f < n), f \mid n$

Solution: This quantification is false.
If $n$ is a prime number, then there are no factors, more than 1 but less than $n$, which divide it.

(e) $\exists(n \geq 3) \forall(1 < f < n), f \mid n$

Solution: This quantification is false.
There is no number where all of the numbers between 1 and it are its factors. (To double-check, we inspect the extreme case, $n = 3$. In that case, the inner quantification fails because 2 is not a factor of 3. For all larger numbers, any value of $n$ we considered would need to be a multiple of both 2 and 3 (and thus of 6); but multiples of 6 would also need to be multiples of 4 and 5, and so on.
Stated another way, there is no number which has a factor other than itself) which is greater than or equal to its own square root. Thus, the inner quantification cannot be true of any value of $n$.

(f) $\forall x \exists y, x^2 \geq y$
Solution: This quantification is **trivially true**; we simply set \( y = x^2 \).

(g) \( \exists x \forall y, x^2 \geq y \)

Solution: This is **false**, since, for any particular value of \( x \), we can choose an integer \( y > x^2 \).

(h) \( \forall (c > 0) \exists n_0 \forall (n > n_0), (n^2 \geq cn) \)

Solution: This quantification is **true**.

No matter what value we choose for \( c \), we can find an \( n_0 \) (greater than or equal to \( c \), so that, for all \( n > n_0 \), the inequality will be true.

This is because, of course, when \( n_0 = c \) and \( n = n_0 \), the equation becomes

\[
\begin{align*}
\frac{n^2}{c} & \geq cn \\
n_0 \cdot n & \geq n_0 \cdot n
\end{align*}
\]

Clearly, at that point, the two sides are equal; as \( n \) gets larger, the left side grows more rapidly than the right side. Thus, from that point, the predicate will **always** be true (as required by the quantification).

**Reminder:** \( f \mid n \) means “\( f \) divides \( n \)” - that is, \( f \) is a factor of \( n \). Likewise, \( f \nmid n \) means “\( f \) does not divide \( n \)”.
2 Quantifications and Code

Suppose that I have written a method \( p() \) in Java. It takes two integer parameters, and returns an integer. Both integer inputs are allowed to range from 0 to 99 (inclusive).

(a)
Write a method \( p2() \) in Java, which takes a single parameter \( \text{int } x \), and which returns the value of the quantification below.

\[ \exists y, \ p(x, y) == 0 \]

Remember: You only need to loop over the valid range of \( p() \) - not the range of all possible ints!

Solution:

```java
public static boolean p2(int x) {
    for (int y=0; y<100; y++)
    {
        if (p(x,y) == 0)
            return true;
    }
    return false;
}
```

(b)
Write a method \( q() \) in Java, which takes no parameters and returns the value of the following quantification:

\[ \exists x \forall y, \ p(x, y) == 0 \]

I encourage you to use a helper method - although it’s not required.

Solution:

Helper Method Version:

```java
public static boolean q()
{
    for (int x=0; x<100; x++)
    {
        if (q_helper(x))
            return true;
    }
    return false;
}

private static boolean q_helper(int x)
{
```

[Incomplete code]
for (int y=0; y<100; y++)
  if (p(x,y) != 0)
    return false;
return true;
}

Single Method Version:

public static boolean q()
{
  for (int x=0; x<100; x++)
  {
    boolean allOK = true;
    for (int y=0; y<100; y++)
      if (p(x,y) != 0)
      {
        allOK = false;
        break;
      }
    if (allOK)
      return true;
  }
return false;
3 Induction

Prove each of the following conjectures using induction.

(a) Conjecture:
\[ \sum_{i=0}^{n} (5^i) = \frac{5^{n+1} - 1}{4}, \quad n \in \mathbb{N} \]

Solution: Base: \( n = 0 \)
Left side: \( \sum_{i=0}^{0} (5^i) = 5^0 = 1 \)
Right side: \( \frac{5^{0+1} - 1}{4} = \frac{5^1 - 1}{4} = 1 \)
Base case holds.

Inductive Step:
We will assume that the conjecture holds for some \( k \). We will attempt to prove that it also holds for \( k + 1 \).

The left side is:
\[ \sum_{i=0}^{k+1} (5^i) \]
\[ 5^{k+1} + \sum_{i=0}^{k} (5^i) \]

By the I.H.:
\[ 5^{k+1} + \frac{5^{k+1} - 1}{4} \]
\[ \frac{5 \cdot 5^{k+1} - 1}{4} \]
\[ \frac{5^{k+2} - 1}{4} \]
Inductive step holds.

(b) Instructor’s Note: After reviewing this, I realized that this conjecture holds for all non-negative values of \( n \), not just the even ones. My solution originally used \( k + 1 \) in the Inductive Step, whereas it should have used \( k + 2 \) - but I didn’t notice, because the algebra worked for \( k + 1 \) as well as \( k + 2 \). I’ve updated the proof below to use \( k + 2 \).

Conjecture:
\((7^n - 4)\) is divisible by 3, for all even, non-negative values of \( n \).

Solution: Base: \( n = 0 \)
\( 7^0 - 4 = 1 - 4 = -3 \). This is a multiple of 3. Base case holds.

Inductive Step:
We will assume that the conjecture holds for some \( k \). We will attempt to prove that it also holds for \( k + 2 \).
\[ 7^{k+2} - 4 = 49 \cdot 7^k - 4 = 49 \cdot 7^k - 196 + 192 \]
\[ 49(7^k - 4) + 192 \]
By the I.H., we know that \((7^k - 4)\) is a multiple of 3, and so is 192. Thus, \((49(7^k - 4) + 192)\) is a multiple of 3.

Inductive step holds.

(c) **Conjecture:**
\[
\forall n \in \mathbb{Z}^+, (1 \cdot 1! + 2 \cdot 2! + ... + n \cdot n!) = (n + 1)! - 1
\]

**Solution:**

**Base:** \(n = 1\)
- Left side: \((1 \cdot 1!) = 1\)
- Right side: \((1 + 1)! - 1 = 2! - 1 = 2 - 1 = 1\)

Base case holds.

**Inductive Step:**
We will assume that the conjecture holds for some \(k\). We will attempt to prove that it also holds for \(k + 1\).

The left side is:
\[
1 \cdot 1! + 2 \cdot 2! + ... + n \cdot n!
\]
\[
1 \cdot 1! + 2 \cdot 2! + ... + k \cdot k! + (k + 1) \cdot (k + 1)!
\]

By the I.H.:
\[
(k + 1)! - 1 + (k + 1) \cdot (k + 1)!
\]
\[
(k + 2) \cdot (k + 1)! - 1
\]
\[
\cdot (k + 2)! - 1
\]
Inductive step holds.

(d) **Conjecture:**
Use induction over \(n\) to prove that, if \(x > -1\) and \(n \in \mathbb{Z}^+\), then \((1 + x)^n \geq 1 + nx\).

**Solution:**

**Base:** \(n = 1\)
\[
(1 + x)^1 \geq 1 + 1x
\]
\[
1 + x \geq 1 + x
\]
Thus, the base case holds (even without assuming anything about \(x\)).

**Inductive:**
Suppose that the conjecture holds for \(n = k\). We will try to prove that it also holds for \(n = k + 1\).

In this proof, we will **start** with the I.H.:
\[
(1 + x)^n \geq 1 + nx
\]
We multiply each side by \((1 + x)\). Since \(x > -1\), we know that this is a positive value; if \(x\) was less than -1, then this would be negative, and it would reverse the direction of the inequality.
\[
(1 + x)^{n+1} \geq (1 + nx)(1 + x)
\]
\[
(1 + x)^{n+1} \geq 1 + nx + nx^2 \\
(1 + x)^{n+1} \geq 1 + x(n + 1) + nx^2
\]

Clearly, \(x^2 > 0\), and we also know that \(n > 0\). So we can conclude:

\[
(1 + x)^{n+1} \geq 1 + x(n + 1) + nx^2 > 1 + (n + 1)x
\]

Thus:

\[
(1 + x)^{n+1} > 1 + (n + 1)x
\]

Thus, the inductive step holds.

**Summary:**

Thus, the conjecture holds for all \(n \in \mathbb{Z}^+\).

(e) **Conjecture:**

In any three sequential numbers from the Fibonacci sequence, exactly two of the numbers are odd.

**NOTE 1:** For this problem, assume that the first two numbers in the Fibonacci sequence are 0, 1. (However, this conjecture would still be true if you assumed that the first two were 1, 1.)

**NOTE 2:** Remember to consider overlapping sequences! For instance, if you assume something is true about \(F_k, F_{k+1}, F_{k+2}\), then you need to prove something about \(F_{k+1}, F_{k+2}, F_{k+3}\), which has **two values in common** with the previous set.

**Solution:** **Base:** First three values: 0, 1, 1

Base case holds.

**Inductive Step:**

We will assume that the conjecture holds for some three sequential values \(F_k, F_{k+1}, F_{k+2}\).

We will attempt to prove that the conjecture holds for the values \(F_{k+1}, F_{k+2}, F_{k+3}\).

There are 3 possible arrangements of even/odd amongst the values \(F_k, F_{k+1}, F_{k+2}\):

- even, odd, odd - \(F_{k+3}\) would be even
- odd, even, odd - \(F_{k+3}\) would be odd
- odd, odd, even - \(F_{k+3}\) would be odd

Thus, in all three cases, the new range would have exactly 1 even number.

Inductive step holds.
4 Structural Induction (with example)

Consider the following theorem (and its proof). After you have read this proof, prove the same conjecture again - but this time, use the “root plus subtrees” strategy for structural induction.

**Conjecture:**
A non-empty k-ary tree with \( n \) nodes has \( n - 1 \) edges.

**Base Case: 1 node**
A tree with only a single node has no edges between the nodes, and thus the conjecture holds trivially.

**Inductive:**
Assume that the conjecture holds for any tree which has exactly \( n \) nodes. We will prove that it also holds for any tree which has exactly \( n + 1 \) nodes.

Any tree with \( n + 1 \) nodes is a tree with \( n \) nodes, plus a single new leaf, added as a child of a certain node. The new, larger tree has one more edge than the previous one - but it also has one more node.

By the I.H., any tree with \( n \) nodes has \( n - 1 \) edges; thus, the tree with \( n + 1 \) nodes certainly had \( n \) edges.

Thus, the inductive step holds.

**Summary**
Thus, the conjecture holds for all non-empty trees.

*aNote that you can’t assume that it’s a binary tree!*

---

**Solution: Base Case: A single node**
Clearly, this is true (same base case as above).

**Inductive Step:**
Assume that the conjecture holds for any tree which has exactly \( n \) nodes. We will prove that it also holds for any tree which has exactly \( n + 1 \) nodes.

A k-ary tree with \( n + 1 \) nodes \((n + 1 > 1)\) is a root node, plus one or more subtrees. The sub-trees have \( n_1, ..., n_i \) nodes, and thus have \( n_1 - 1, ..., n_i - 1 \) edges respectively.

The total number of nodes is

\[
\sum n_i + 1
\]

Since there is an edge from the root to each of the subtrees, the total number of edges is

\[
(n_1 - 1) + \ldots + (n_i - 1) + i = n_1 + \ldots + n_i
\]

which is one less than the number of nodes.

Thus, the inductive step holds.

---

5 Structural Induction (from scratch)

Prove the following conjecture using structural induction:

**Conjecture:**
A non-empty binary tree with \( n \) internal nodes has no more than \( 2n + 1 \) nodes.
Solution: Base: single node
A tree made up of a single node has 0 internal nodes, and 1 total node.
Base case holds, because $1 \leq 2 \cdot 0 + 1$.

Inductive Step:
We will assume that the conjecture holds for some $k$. We will attempt to prove that it also holds for $k + 1$.

A non-empty binary tree with $n + 1$ internal nodes (with $n + 1 > 1$) is either a root node with one subtree, or a root node with two subtrees. In both cases, the root node must be an internal node - meaning that the total number of internal nodes, between the one or two subtrees, is exactly $n$.

In the first case, there is one subtree, and it has $n$ nodes; by the I.H., the subtree can have at most $2n + 1$ nodes. Thus, the entire tree has at most $2n + 2$ nodes, which is more restrictive than the conjecture requires (the conjecture allows up to $2n + 3$ nodes).

In the second case, there are two subtrees. They have $n_1, n_2$ internal nodes, respectively with $n_1 + n_2 = n$. By the I.H., they have $2n_1 + 1, 2n_2 + 1$ total nodes respectively, and so the total tree has at most this many nodes:

$$(2n_1 + 1) + (2n_2 + 1) + 1 = 2(n_1 + n_2) + 3 = 2n + 3$$

Inductive step holds.