Lab01-Proof

CS363-Computability Theory, Xiaofeng Gao, Spring 2016

∗ Please upload your assignment to TA's FTP. Contact nongeek.zv@gmail.com for any questions.
∗ Name: Zhang Yupeng StudentId: 5130309468 Email: 845113336@qq.com

1. Prove that for any integer $n > 2$, there is a prime p satisfying $n < p < n!$. (Hint: consider a prime factor p of $n! - 1$ and use proof by contradiction)
   Proof: Assume that for any integer $n > 2$,there is no prime p satisfying $n < p < n!$.
   The adjacent two natural numbers are co-prime, so $n!$ and $n! - 1$ are co-ptime.
   Because $n! = 1 * 2 ... * n - 1 * n$, so $1, 2, 3, ..., n - 1, n$ are all factors of $n!$.
   So $1, 2, ...n$ all aren't factors of $n! - 1$.
   So the prime factor of $n! - 1$ is greater than n, which contradicts our assumption.
   So we proof it by contradiction.

2. Use minimal counterexample principle to prove that: for every integer $n > 17$, there exist integers $i_n \geq 0$ and $j_n \geq 0$, such that $n = i_n \times 4 + j_n \times 7$.
   Proof: If $n = i_n * 4 + j_n * 7$ is not true for every integer $n > 17$, then there are values of n for which $n \neq i_n * 4 + j_n * 7$, and there must be a smallest such value, say $n = k$.
   Since $18 = 1 * 4 + 2 * 7, 19 = 3 * 4 + 1 * 7, 20 = 5 * 4, 21 = 3 * 7, 22 = 2 * 4 + 2 * 7$, we have $k \geq 23, k - 4 > 18$.
   Sinve k is the smallest value for which $k \neq i_k * 4 + j_k * 7$, so $k - 4 = i_{k-4} * 4 + j_{k-4} * 7$ is true.
   However, we have $k = k - 4 + 4 = i_{k-4} * 4 + j_{k-4} * 7 + 4 = (i_{k-4} + 1) * 4 + j_{k-4} * 7$,which derived a contradiction. So our original assumption is false.

3. Suppose $a_0 = 1$, $a_1 = 2$, $a_2 = 3$, $a_k = a_{k-1} + a_{k-2} + a_{k-3}$ for $k \geq 3$. Use strong principle of mathematical induction to prove that $a_n \leq 2^n$ for all integers $n \geq 0$.
   Proof: Obviously, $a_0 \leq 2^0, a_1 \leq 2^1, a_2 \leq 2^2$, and $a_3 = a_0 + a_1 + a_2 = 1 + 2 + 3 = 6 < 2^3$.
   We assume that $a_n < 2^n$ is true for every n satisfying $n_0 \leq n \leq k, k \geq 0$.
   Then, $a_{k+1} = a_k + a_{k-1} + a_{k-2} \leq 2^k + 2^{k-1} + 2^{k-2} < 2^{k+1}$.
   So, we proof the original assumption.

4. Consider the following loop, written in pseudocode:

```
while B do
  | S;
end
```

   A condition P is called an invariant of the loop if whenever P and B are both true, and S is executed once, P is still true.

   (a) Prove that if P is an invariant of the loop, and P is true before the first iteration of the loop, then if the loop eventually terminates (i.e., after some number of iterations, B is false), P is still true.
   Proof: Because P is an invariant of the loop. And P is true before the first iteration of the loop, so if B is true, then S is excuted once, P is still true.
   If B is false, then S cannot be excuted, so P will maintain the value in the last iteraion. So P is still true.
   So we can proof that when the loop terminates, P is still true.

   (b) Suppose x and y are integer variables, and initally $x \geq 0$ and $y > 0$. Consider the following program fragment:

```
q = 0;
r = x;
while r ≥ y do
    q = q + 1;
    r = r − y;
end
```

By considering the condition $(r \geq 0) \wedge (x = q \times y + r)$, prove that when this loop terminates, the values of q and r will be the integer quotient and remainder, respectively, when x is divided by y; in other words, $x = q \times y + r$ and $0 \leq r < y$.
Proof:We claim that the loop invariant x:

$$x = qy + r$$

Because$x = qy + r, x \geq 0, q = 0, r = x$ before the loop executes. So x is true before the loop.
Then we assume that x is true before the loop is executed. Then, after the loop executes, we have the new values $r_n = r - y$ and $q_n = q + 1$.
Since, by the condition of the loop we know that $r \geq y$, so we have that $r_n = r - y \geq 0$.
Furthermore, $x = qy + r = qy + r - y + y = (qy + y) + (r - y) = (q+1)y + (r - y) = q_n y + r_n$
Thus, x is still true after the loop executes. When the loop terminates, the condition of the loop is false, so that $r < y$. So, $x = q * y + r$ and $0 \leq r < y$. $s_1 = 1$