CSC236, Summer 2005, Assignment 3
Sample solution
Danny Heap

The methods listed below may be examined and run by downloading A3Examples.java from the web page. If you have any questions about Java, please ask. You may assume (although it’s not true) that the Java type int includes all integers.

1. Use the technique from lecture to prove the following method is correct with respect to its specification (precondition and postcondition).

/**
 * pow(b,p) returns b^p.
 * @param b an integer base to exponentiate.
 * @param p a natural number power to exponentiate to.
 * @return b^p.
 * @postcondition: b is an integer and p is a natural number.
 */
public static int pow(int b,int p) {
    int bToTheP = 1;
    int i = 0;
    while (i != p) {
        bToTheP *= b;
        ++i;
    }
    return bToTheP;
}

**Solution:** I’d like to have bToTheP equal to b^p when the loop exits, so I prove the following invariant:

**Claim 1A:** Let \( P(j) \) be “If there is a \( j \)th iteration of the loop of pow then \( b^{ToTheP_j} = b^i \) and \( i_j = j \.”

I claim \( P(j) \) is true for all \( j \in \mathbb{N} \).

**Proof (Induction on \( j \):** Suppose \( j = 0 \). Then (by the program) \( bToTheP_0 = 1, i_0 = 0, \) so \( bToTheP_j = 1 = p^i \), and the base case holds.

**Induction Step:** Assume that \( P(j) \) holds for some \( j \in \mathbb{N} \). I want to show that this implies \( P(j + 1) \).

If there is no \( (j + 1) \)th iteration, then \( P(j + 1) \) is vacuously true. Otherwise, by the program, \( bToTheP_{j+1} = bToTheP_j \times b = (by \ the \ IH) \ b^j \times b = b^{j+1} \). Also (by the program) \( i_{j+1} = i_j + 1 = (by \ the \ IH) \ j + 1 \). This is what \( P(i + 1) \) claims, so \( P(i) \Rightarrow P(i + 1), \) as wanted.

I conclude that \( P(i) \) is true for all \( i \in \mathbb{N} \). QED.

**Claim 1B:** If the precondition holds when pow starts and pow terminates, when it does terminate the postcondition holds.
**Proof:** Suppose the precondition holds and pow terminates after $j$ iterations. Then (by the program) $i_j = p$, so (by the IH) $j = p$, and (by the IH again) $b \text{ToTheP}_{i_j} = b^p$. Thus $b^p$ is returned, and the postcondition holds.

I want to associate a strictly-decreasing sequence of natural numbers with the loop iterations, so I prove the following claims:

**Claim 1C:** Let $Q(j)$ be “If there is a $j$th loop iteration in pow, then $i_j \leq p$.” I claim that $Q(j)$ is true for all $j \in \mathbb{N}$.

**Proof (induction on $j$):** Suppose $j = 0$. Then (by the program) $i_j = 0$ and (by the precondition) $p$ is a natural number, so $p \geq i_j$, so the base case holds.

**Induction Step:** Suppose $Q(j)$ holds for some arbitrary $j \in \mathbb{N}$. I wish to show that $Q(j+1)$ follows.

If there is no $(j+1)$th iteration of the loop, $Q(j+1)$ holds vacuously. Otherwise, (by the program) $i_j \neq p$ and (by the IH) $i_j \leq p$, so $i_j < p$. The program then sets $i_{j+1} = i_j + 1$, so $i_j < p$ implies $i_{j+1} = i_j + 1 \leq p$. Thus $Q(j+1)$ holds, so $Q(j) \implies Q(j+1)$, as wanted.

I conclude that $P(j)$ is true for all $j \in \mathbb{N}$. QED.

**Claim 1D:** For each $j \in \mathbb{N}$, if the loop in pow executes at least $j + 1$ times, then $p - i_j > p - i_{j+1}$.

**Proof:** If the loop iterates $j + 1$ times, then it certainly iterates $j$ times, so by $P(j)$ and $P(j + 1)$ (above) we have that $i_j = j$ and $i_{j+1} = j + 1$, so $p - i_j = p - j > p - (j - 1) = p - i_{j+1}$, as claimed.

**Claim 1E:** If the precondition holds, then the loop in pow terminates.

**Proof:** If the loop condition holds, then by $Q(j)$ we know that the sequence $\langle p - i_j \rangle$, contains only natural numbers, since it is a difference of integers and $p \geq i_j$, according to $Q(j)$. Also, by Claim 1D, the sequence $\langle p - i_j \rangle$ is strictly decreasing. Thus $\langle p - i_j \rangle$ corresponds to a subset of the natural numbers, and hence (PWO) has a least element, which (since the sequence is strictly decreasing) is also its last element. Call the last element $p - i_k$. Then there is no $(k + 1)$th iteration of the loop, so the loop terminates.

Thus I have proven partial correctness (Claim 1A) and termination (Claim 1E) of pow in terms of its precondition/postcondition pair.
2. Use the technique from lecture to prove the following method is correct with respect to its precondition/postcondition pair.

/**
 * recPow(b, p) returns b^p (more efficiently).
 * @param b an integer base to exponentiate.
 * @param p a natural number power to exponentiate to.
 * @return b^p.
 * Precondition: b is an integer and p is a natural number.
 * Postcondition: recPow(b, p) terminates, and when it does b^p is returned.
 */
public static int recPow(int b, int p) {
    int bToTheP = 1;
    if (p % 2 == 1) {
        bToTheP = b;
    }
    if (p < 2) {
        return bToTheP;
    }

    // p/2 is integer division
    return bToTheP * recPow(b, p/2) * recPow(b, p/2);
}

SAMPLE SOLUTION: This is a recursive program, so I prove that if the precondition is true when recPow(b, p) is called, then recPow(b, p) terminates, and when it does it returns b^p. The parameter that expresses the size of the input is p, so I use induction on p.

CLAIM: Let P(p) be "\forall b \in \mathbb{Z}, if the precondition holds when recPow(b, p) is called, then recPow(b, p) terminates, and when it does it returns b^p." Then \forall p \in \mathbb{N}, P(p).

PROOF (COMPLETE INDUCTION ON p): Assume P\({\{0, \ldots, p - 1}\}) is true, for some arbitrary integer \(p\). I want to show that this implies P(p). There are three cases to consider:

CASE 1, p = 0: In this case the program sets bToTheP to 1, does not execute the "(p%2 == 1)" branch (since 0 is even), and then does execute the "(p < 2)" branch (since 0 < 2), terminating and returning bToTheP = 1, which is \(b^0\). Thus this base case holds.

CASE 2, p = 1: In this case the program sets bToTheP to 1, then executes the "(p%2 == 1)" branch (since 1 is odd), setting bToTheP to b, and finally executes the "(b < 2)" branch (since 1 < 2), terminating and returning bToTheP = \(b = b^1\). Thus this base case holds.

CASE 3, p > 1: In this case I have assumed P(p/2) in the IH, since 0 ≤ p/2 < p when p > 1. The program sets bToTheP to 1, and then there are two sub-cases to consider:

SUB-CASE 3A, p IS ODD: In this case bToTheP is set to b (since p is odd), the program does not execute the "(p < 2)" branch (since p ≥ 2), and it returns bToTheP × recPow(b, p/2) × recPow(b, p/2). By the IH, recPow(b, p/2) = \(b^{p/2}\), which equals \(b^{(p-1)/2}\) (since p is odd and p/2 is integer division). Thus the program terminates and returns \(b \times b^{(p-1)/2} \times b^{(p-1)/2} = b^p\). So the claim holds in this case.

SUB-CASE 3B: In this case bToTheP is set to 1, the "(p%2 == 1)" branch is not executed (since p is even), the "(p < 2)" branch is executed (since p ≥ 2), and the program returns
bToTheP \times \text{recPow}(b,p/2) \times \text{recPow}(b,p/2). By the IH \text{recPow}(b,p/2) = b^{p/2}. Since p is even, p/2 + p/2 = p, so the program terminates and returns 1 \times b^{p/2} \times b^{p/2} = b^p. So the claim holds in this case.

In all cases the claim holds, so \( P\{0, \ldots, p - 1\} \Rightarrow P(p). \) I conclude that \( P(p) \) is true for all \( p \in \mathbb{N} \). QED.

3. Use the technique from lecture to prove the following method correct with respect to its precondition/postcondition pair.

```java
/**
 * \text{div}(m,n) returns quotient and remainder for m/n
 * @param m a natural number dividend.
 * @param n a positive natural number divisor.
 * @return [q,r] such that m = qn + r and 0 <= r < n.
 * Precondition: m is a natural number, n is a positive natural number.
 * Postcondition: [q,r] is returned and m = qn + r, 0 <= r < n.
 */
public static int[] div(int m, int n) {
    int[] quotRem = {0,0}; // initialized to [0,0]
    while (m != quotRem[0] * n + quotRem[1]) {
        if (quotRem[1] < n-1) {
            ++quotRem[1];
        } else {
            ++quotRem[0];
            quotRem[1] = 0;
        }
    }
    return quotRem;
}
```

**SAMPL SOLUTION:** My informal examination of the program suggests that quotRem[0] \* n + quotRem[1]
never exceeds m, and that it increases by 1 each loop iteration until it exits. Therefore I prove the following claim:

**CLAIM 3A:** Let \( P(i) \) be "\( \forall n, m \in \mathbb{N}, n > 0, \) if the precondition holds when \text{div}(m,n) \) begins and there is an \( i \)th iteration of the loop in \text{div}(m,n), then\( \text{quotRem}[0]_i \ast n + \text{quotRem}[1]_i \leq m \) and \( 0 \leq \text{quotRem}[1]_i < n. \) I claim that \( \forall i \in \mathbb{N}, P(i). \)

**PROOF (INDUCTION ON i):** If \( i = 0 \), then the program sets \text{quotRem}[0]_0 = \text{quotRem}[1]_0 = 0, and certainly \( m \geq 0n + 0 = 0 \), since (by the precondition) \( m \) is a natural number. Also, \( 0 \leq \text{quotRem}[1]_0 = 0 < n, \) since (by the precondition) \( n \) is a positive natural number. Thus \( P(0) \) holds.

**INDUCTION STEP:** Assume that \( P(i) \) is true for some arbitrary natural number \( i. \) I must show that this implies \( P(i + 1). \) If there is no \( (i + 1) \)th loop, then \( P(i + 1) \) is vacuously true. Otherwise, there is an \( (i + 1) \)th loop iteration, so \( m \neq \text{quotRem}[0]_i \ast n + \text{quotRem}[1]_i \) and (from the IH \( P(i) \)) \( m \geq \text{quotRem}[0]_i \ast n + \text{quotRem}[1]_i, \) you must have \( m > \text{quotRem}[0]_i \ast n + \text{quotRem}[1]_i. \) There are two cases to consider:

**CASE 1, \text{quotRem}[1]_i < n - 1:** In this case the "if \( (\text{quotRem} < n - 1)" \) branch is executed, and
\[ \text{quotRem}[1]_{i+1} = \text{quotRem}[1]_i + 1, \text{ and so} \]

\[ m > \text{quotRem}[0]_i \cdot n + \text{quotRem}[1]_i \quad \geq \quad \text{quotRem}[0]_i \cdot n + \text{quotRem}[1]_i + 1 \]

\[ = \text{quotRem}[0]_{i+1} \cdot n + \text{quotRem}[1]_{i+1}. \]

Also, \( \text{quotRem}[1]_i < n - 1 \Rightarrow \text{quotRem}[1]_i + 1 = \text{quotRem}[1]_{i+1} < n. \) Thus \( P(i + 1) \) holds in this case.

\text{CASE 2, } \text{quotRem}[1]_i \geq n - 1: \text{ In this case the "}(\text{quotRem}[1]_i < n - 1) \text{ else" branch is executed, setting } \text{quotRem}[0]_{i+1} = \text{quotRem}[0]_i \text{ and } \text{quotRem}[1]_{i+1} = 0. \text{ By } P(i) \text{ we know that } \text{quotRem}[1]_i \leq n - 1, \text{ so in this case we must have } \text{quotRem}_i = n - 1, \text{ and so } \]

\[ m > \text{quotRem}[0]_i \cdot n + \text{quotRem}[1]_i \quad = \quad \text{quotRem}[0]_i \cdot n + (n - 1) \]

\[ \Rightarrow m \quad \geq \quad \text{quotRem}[0]_i \cdot n + n \]

\[ = \quad (\text{quotRem}[0]_i + 1) \cdot n + 0 \]

\[ = \quad \text{quotRem}[0]_{i+1} \cdot n + \text{quotRem}[1]_{i+1}. \]

Also, \( 0 \leq 0 = \text{quotRem}[1]_i < n \) (since \( n \) is a positive number, according to the precondition).

Thus \( P(i + 1) \) holds in this case as well.

In either case, \( P(i + 1) \) holds, so \( P(i) \Rightarrow P(i + 1) \), as wanted.

I conclude \( P(i) \) is true for all \( i \in \mathbb{N} \). QED.

**CLAIM 3B (PARTIAL CORRECTNESS):** If the precondition is true when \( \text{div}(m,n) \) begins, and if \( \text{div}(m,n) \) terminates, then the postcondition holds.

**PROOF:** Suppose \( \text{div}(m,n) \) satisfies the precondition when it begins, and then terminates after some arbitrary iteration \( k \) of the loop. Since the loop terminates, we must have \( \text{quotRem}[0]_k \cdot n + \text{quotRem}[1]_k = m \). Also, by \( P(k) \), we have \( 0 \leq \text{quotRem}[1]_k < n \). Thus \( \text{quotRem} \) with values that satisfy the postcondition is returned. QED.

**CLAIM 3C, DECREASING SEQUENCE IN \( \mathbb{N} \):** Let sequence \( \langle d_i \rangle \), associated with the \( i \)th iteration of the loop, be defined by \( \langle d_i \rangle = \langle m - (\text{quotRem}[0]_i \cdot n + \text{quotRem}[1]_i) \rangle \). Then I claim that \( \langle d_i \rangle \) is a strictly decreasing sequence of natural numbers.

**PROOF:** Since \( d_i \) is formed from differences, sums, and products of integers, it is clearly an integer.

By \( P(i) \) (proved above) \( d_i \) is non-negative, hence \( d_i \) is a natural number. Suppose there is an \((i + 1)\)th iteration of the loop. Then, examining the two cases in the proof of \( P(i) \), we see that \( d_{i+1} = d_i - 1 \), so the sequence is strictly decreasing. Thus \( \langle d_i \rangle \) is a strictly decreasing sequence of natural numbers. QED.

**CLAIM 3D (TERMINATION):** If the precondition is true when \( \text{div}(m,n) \) is started, then it terminates.

**PROOF:** If the precondition is true when \( \text{div}(m,n) \) starts, then each iteration \( i \) of the loop is associated with element \( \langle d_i \rangle \) of the sequence from Claim 3C. This sequence is non-empty (since it contains, at the very least, \( d_0 \)), so its elements correspond to a non-empty subset of \( \mathbb{N} \). This means (by the PWO) there is a least element, call this \( d_k \). Since the sequence is strictly decreasing, there is no element \( d_{k+1} \), and hence the loop terminates before iteration \( k + 1 \). QED.

4. Define \( T(n) \) by:

\[ T(n) = \begin{cases} 
  c, & 1 \leq n < 5 \\
  a_1 T(\lfloor n/5 \rfloor) + a_2 T(\lceil n/5 \rceil) + dn, & n \geq 5,
\end{cases} \]

\[ \ldots \text{ where } a_1, a_2 \in \mathbb{N}, a_1 + a_2 = 5, \text{ and } c, d \in \mathbb{R}^+ \text{ (the positive reals). Prove there is some positive constant } \kappa \in \mathbb{R}, \text{ such that } T(n) \leq \kappa n \log_5 n, \text{ for all } n \geq 5. \text{ You may NOT use the Master Theorem.} \]
SAMPLE SOLUTION: I need to prove a special case of the result when \( n \) is a natural power of 5, and also that \( T(n) \) is monotonic.

CLAIM 4A: Let \( P(k) \) be \( "T(5^k) = 5^k(c + kd)." \) The I claim \( P(k) \) is true for all \( k \in \mathbb{N} \).

PROOF (INDUCTION ON \( k \)): Suppose \( k = 0 \). Then \( P(0) \) claims that \( T(1) = c \), which is true, by inspecting the definition of \( T(1) \). So the base case holds.

INDUCTION STEP: Assume that \( P(k) \) holds for some arbitrary \( k \in \mathbb{N} \). I want to show that \( P(k + 1) \) holds. Since \( [5^{k+1}/5] = [5^k + 1/5] = 5^k \), we have

\[
T(5^{k+1}) = 5T(5^k) + d5^{k+1} \quad \text{(by definition)}
\]

\[
= 5(5^k(c + kd5^k)) + d5^{k+1} \quad \text{(by IH \( P(k) \))}
\]

\[
= 5^{k+1}c(k + 1)d.
\]

This is exactly what \( P(k + 1) \) claims, so \( P(k) \Rightarrow P(k + 1) \).

I conclude that \( P(k) \) is true for all \( k \in \mathbb{N} \). QED.

CLAIM 4B: \( \forall n \in \mathbb{N}, n \geq 5 \Rightarrow 1 \leq [n/5] \leq [n/5] < n \).

PROOF: By definition of floor and ceiling we have \([n/5] \leq [n/5]\). Write \( n = 5k + j \), for some \( k \in \mathbb{Z} \) and \( 0 \leq j < 5 \) (this is possible, by the division algorithm). Then we know that (depending on whether \( j = 0 \) or not):

\[
\left\lfloor \frac{n}{5} \right\rfloor = \left\lfloor \frac{5k + j}{5} \right\rfloor = \frac{5k + i}{5}, \quad \text{where} \quad \begin{cases} i = 5 - j, & j > 0 \\ i = j, & j = 0 \end{cases}
\]

So \([n/5] \leq (n + 4)/5\). Also, \( n > 4 \), so \( n + n > n + 4 \), and \( 5n > n + 4 \), so \( n > (n + 1)/5 \geq [n/5] \). At the other end, \( n \geq 5 \) means \( n/5 \geq 1 \), so \([n/5] \geq 1 \). Putting these all together, \( 1 \leq [n/5] \leq [n/5] < n \).

QED.

CLAIM 4C (MONOTONICITY): Let \( P(n) \) be \( "If \( m \) is a positive natural number less than \( n \), then \( T(m) \leq T(n) \)." \) Then I claim that \( \forall n \in \mathbb{N} - \{0\}, P(n) \) is true.

PROOF (COMPLETE INDUCTION ON \( n \)): Assume that \( P(\{1, \ldots, n - 1\}) \) is true for some arbitrary positive natural number \( n \). I want to show that this implies \( P(n) \). There are three cases to consider:

CASE 1, 1 \( \leq n < 5 \): In this case, the only positive natural numbers less than \( n \) are \( 1 \leq m < n \), so \( T(m) = c = T(n) \), so the claim holds in these cases.

CASE 2, \( n = 5 \): In this case, the only positive natural numbers less than \( n \) are \( 1 \leq m < 5 \), with \( T(m) = c \) in each case, so \( T(m) = c < 5T(1) + 5d = 5c + d \), since \( c \) and \( d \) are positive constants. So the claim holds in this case.

CASE 3, \( n > 5 \): In this case we have \( 1 \leq [n/5], [n/5], n - 1 < n \), so the IH assume \( P(n - 1), P([n/5]), \) and \( P([n/5]) \). By \( P(n - 1) \) we know that for any \( 1 \leq m < n - 1 \) we have \( T(m) \leq T(n - 1) \), so it is enough to show that \( T(n - 1) \leq T(n) \), and

\[
T(n - 1) = a_1T\left([n - 1]/5\right) + a_2T\left([(n - 1)/5]\right) + d(n - 1) \quad \text{(since \( n > 5 \Rightarrow n - 1 \geq 5 \))}
\]

\[
\leq a_1T\left([n/5]\right) + a_2T\left([n/5]\right) + dn \quad \text{(by \( P([n/5]) \), \( P([n/5]) \), and \( d > 0 \))}
\]

Thus \( P\{1, \ldots, n - 1\} \Rightarrow P(n) \), as wanted, in each case.

I conclude that \( P(n) \) is true for all \( n \in \mathbb{N} - \{0\} \).

CLAIM 4D: If \( n \geq 5 \), then there is some \( \kappa \in \mathbb{R} \) such that \( T(n) \leq \kappa n \log_5 n \).
Proof: Let \( n \geq 5 \). Then \( n = 5^{\log_5 n} \leq 5^{[\log_5 n]} \). Call \( 5^{[\log_5 n]} \tilde{n} \). Then we have

\[
T(n) \leq T(\tilde{n}) \quad \text{(by CLAIM 4c)}
\]

\[
= \tilde{n}(c + d \log_5 \tilde{n}) \quad \text{(by CLAIM 4A)}
\]

\[
\leq 5n(c + d \log_5 5n) \quad \text{(\( \log_5 \) is monotonic, and \( [\log_5 n] < (\log_5 n) + 1 \))}
\]

\[
= 5n(c + d(\log_5 n + 1)) \leq 5n(c \log_5 n + d(2\log_5 n)) \quad \text{(\( \log_5 n \geq 1 \), since \( n \geq 5 \))}
\]

\[
= 5(c + 2d)n \log_5 n
\]

Thus the claim holds, with \( \kappa = 5(c + 2d) \). QED.