

AXIOMATIC PROOFS

The exponentiation program

```
{y ≥ 0}
e := 1;
t := y;
while t greater 0 do
  e := e mult x;
  t := t minus 1
end
{e = x**y}
```

To prove the loop partially correct, we need to find the loop invariant. The invariant for this loop is $I = \{e = x^{y-t} \wedge t \geq 0\}$. The first part can be found from a trace of the loop in a simple case:

x	y	t	e
2	3	3	1
		2	2
		1	4
		0	8

This second part comes from the loop termination expression. First we prove the loop itself using the assignment axiom on both the statements in the body:

$\{e = x^{y-(t-1)} \wedge t-1 \geq 0\} \quad t := t \text{ minus } 1 \quad \{e = x^{y-t} \wedge t \geq 0\}$, and

$\{e * x = x^{y-(t-1)} \wedge t-1 \geq 0\} \quad e := e \text{ mult } x \quad \{e = x^{y-(t-1)} \wedge t-1 \geq 0\}$

The precondition for the second one is the same as $\{e = x^{y-t} \wedge t > 0\}$ which is $I \wedge t > 0$, the requirement for applying the loop axiom. Using the axiom for statement sequence, we have: $\{I \wedge t > 0\} \quad e := e \text{ mult } x; t := t \text{ minus } 1 \quad \{I\}$, so applying the loop axiom, we have:

$\{I\} \quad \text{while } \dots \text{ end } \{I \wedge \sim(t > 0)\}$

Taking the invariant back through the first two assignments gives:

$\{e = x^{y-y} \wedge y \geq 0\} \quad t := y \quad \{e = x^{y-t} \wedge t \geq 0\}$, and

$\{1 = x^{y-y} \wedge y \geq 0\} \quad e := 1 \quad \{e = x^{y-y} \wedge y \geq 0\}$

Since $x^{**}0 = 1$, the precondition is just $y \geq 0$, the precondition for the whole program. Using the sequence axiom we thus have:

```
{y ≥ 0}
e := 1;
t := y;
while t greater 0 do
  e := e mult x;
  t := t minus 1
end
```

$$\{e = x^{**}(t-y) \wedge t \geq 0 \sim (t > 0)\}$$

This postcondition implies that $t = 0$, and thus it reduces to $e = x^{**}y$, which completes the proof since this is the required postcondition for the whole program.

To prove termination, we need to find a Floyd expression, which is simply t in this case. We then need to prove well-foundedness first:

$$y \geq 0, I \vdash t \geq 0$$

i.e. from $y \geq 0$ and I , we need to prove that $t \geq 0$. Since it is part of the invariant, this is done. Next we prove termination, by showing:

$y \geq 0, I, t > 0 \vdash t > t[\theta]$, where $t[\theta]$ is the Floyd expression with all substitutions on all paths through the loop body. Clearly the only relevant statement is $t := t \text{ minus } 1$, so we must show that $t > t - 1$. This is true independently of the preconditions, so it is also done. We have thus proved partial correctness using the Hoare axioms, and total correctness using Floyd's method of well-foundedness.

The Division Program

```

{y ≥ 1 ∧ x ≥ 0}
q := 0;
r := x;
while r greater y minus 1 do
  q := q plus 1;
  r := r minus y
end
{x = q * y + r ∧ 0 ≤ r < y}

```

First we find the loop invariant. The computational part is easy, since it is the same as that part of the postcondition:

x	y	q	r
7	2	0	7
		1	5
		2	3
		3	1

However, the termination part is tricky, since r reduces until it is less than or equal to y , and may not be zero. The best we can say is that $r \geq 0$. However, this only occurs if y is positive. If it is zero or negative, the loop will diverge. We must therefore add $y \geq 1$ to the invariant if we want to prove the final result. So,

$$I = \{x = y * q + r \wedge r \geq 0 \wedge y \geq 1\}$$

We proceed as before, proving the loop first. Pusing the invariant back through the loop body, we get:

$$\{x = y * q + (r - y) \wedge r - y \geq 0 \wedge y \geq 1\} r := r \text{ minus } y \{x = y * q + r \wedge r \geq 0 \wedge y \geq 1\}, \text{ and}$$

$$\{x = y * (q + 1) + (r - y) \wedge r - y \geq 0 \wedge y \geq 1\} q := q \text{ plus } 1 \{x = y * q + (r - y) \wedge r - y \geq 0 \wedge y \geq 1\}$$

The precondition reduces to $\{x = y * q + r \wedge r > y - 1 \wedge y \geq 1\}$ and since this is implied by $I \wedge r > y - 1$, we can use the consequence axiom to join the two assignments together, and the strenghtening rule to say:

$$\{I \wedge r > y - 1\} = q := q \text{ plus } 1; r := r \text{ minus } y \{I\}$$

We can then apply the loop axiom to show:

$$\{I\} \text{ while } \dots \text{ end} \{I \wedge \sim(r > y - 1)\}$$

Pushing the invariant back through the first two statements gives us:

$$\{x = y * q + x \wedge x \geq 0 \wedge y \geq 1\} r := x \{x = y * q + r \wedge r \geq 0 \wedge y \geq 1\}, \text{ and}$$

$$\{x = y * 0 + x \wedge x \geq 0 \wedge y \geq 1\} q := 0 \{x = y * q + x \wedge x \geq 0 \wedge y \geq 1\}$$

The precondition reduces to $\{x \geq 0 \wedge y \geq 1\}$, the precondition for the whole program.

Turning to the postcondition, the assertion after the loop is $\{I \wedge \sim(r > y - 1)\}$. Since this implies $x = y * q + r \wedge 0 \leq r < y$, then the program is proved partially correct. The justification for the final step is as follows:

Loop postcondition: $x = y * q + r \wedge r \geq 0 \wedge y \geq 1 \wedge \sim(r > y - 1)$, which is $x = y * q + r \wedge r \geq 0 \wedge y \geq 1 \wedge r \leq y - 1$, or

$x = y * q + r \wedge r \geq 0 \wedge y \geq 1 \wedge r < y$. Since we can drop any conjunct ($A \wedge B \Rightarrow A$) this implies the final postcondition. Thus we have proved partial correctness.

To prove total correctness, we first find the Floyd expression. Like the termination part of the loop invariant, this is just r . To prove well-foundedness, we need to show:

$$y \geq 1 \wedge x \geq 0, I \vdash r \geq 0$$

Since the invariant contains $r \geq 0$, this is proved.

Termination needs us to show:

$$y \geq 1 \wedge x \geq 0, I, r > y - 1 \vdash r > r[\theta]$$

The substitution caused by the loop body on r is $r - y$, so we need to show that $r > r - y$, or $y > 0$. Since this is the same as $y \geq 1$, we are done. So, total correctness is also proved.

The gcd algorithm

The only invariant that makes sense is simply $I \equiv \text{gcd}(N, M) = \text{gcd}(n, m) \wedge N > 0 \wedge M > 0$. This includes as much of the loop termination condition as we can add; the actual loop condition is implicitly contained in the definition of gcd, i.e. $\text{gcd}(n, n) = n$. To prove the loop correct we first have to prove that the conditional statement which is its body is correct. To do this we need to find an A , such that

$$\{A \wedge N > M\} N := N - M \{I\}, \text{ and}$$

$$\{A \wedge N \leq M\} M := M - N \{I\}$$

Using the assignment axiom for each branch, we have:

$$\{I[N - M \setminus N]\} N := N - M \{I\}, \text{ and}$$

$$\{I[M - N \setminus M]\} M := M - N \{I\}$$

So we have to show that

$$A \wedge N > M \Rightarrow I[N - M \setminus N], \text{ and}$$

$$A \wedge N \leq M \Rightarrow I[M - N \setminus M]$$

To see that the A we need is just the invariant plus the loop condition, consider that

$$N > M \Rightarrow \text{gcd}(N, M) = \text{gcd}(N - M, M), \text{ and}$$

$$M > N \Rightarrow \text{gcd}(M, N) = \text{gcd}(M - N, N)$$

from the definition of gcd. Since $I[N - M \setminus N]$ is $\text{gcd}(N - M, M) = \text{gcd}(n, m) \wedge M > 0 \wedge N > M$, we can say that $I \wedge N \neq M \wedge N > M$ is true whenever $I[N - M \setminus N]$ is true, and $I \wedge N \neq M \wedge N > M$ cannot be true when $I[N - M \setminus N]$ is false (since $N > M$ cannot be true and $\text{gcd}(N, M) = \text{gcd}(N - M, M)$ false at the same time). Therefore $I \wedge N \neq M \wedge N > M \Rightarrow I[N - M \setminus N]$. Similarly we can show that $I \wedge N \neq M \wedge N \leq M \Rightarrow I[M - N \setminus M]$. The left hand side of this implication is the same as $I \wedge N \neq M \wedge N < M$, without the equality part of the condition; we thus have the same implication as the other branch with N and M swapped.

Thus the conditional is proved with a pre-condition of $I \wedge M \neq N$, i.e. $\{I \wedge M \neq N\}$ if ... $\{I\}$. Since the pre-condition of this is precisely the right pre-condition for the loop to be correct, using the while axiom we can assert:

$\{I\}$ while... $\{I \wedge \sim (M \neq N)\}$

Pushing the invariant back through the initial assignments gives us simply $\text{gcd}(n, m) = \text{gcd}(n, m) \wedge n > 0 \wedge m > 0$ so the program's pre-condition is proved.

The while loop post-condition implies that there is an x such that $\text{gcd}(n, m) = \text{gcd}(N, M) = x$, where x is the value of N and M after the loop terminates. Thus the program is proved partially correct relative to the given specification.