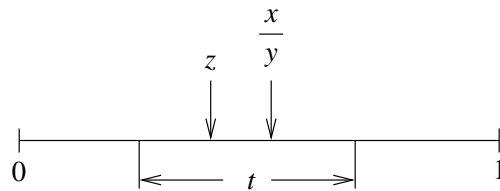


Find the *quotient*  $z$  of real numbers  $0 \leq x < y \leq 1$  to within tolerance  $t$  (without using division, of course).

$$\{0 \leq x < y \leq 1\} \quad \boxed{\mathbf{S}} \quad \{z \leq x/y < z + t\}$$



“Solve for  $\boxed{\mathbf{S}}$ .”

Programming strategy:

```

{0 ≤ x < y ≤ 1}
begin
  make a guess;
  while ¬ good-enough
  { This is a reasonable guess }
  do
    improve the guess
  end
  {z ≤ x/y < z + t}

```

Introduce a variable  $d$  to represent the *known accuracy* of the guess. This is a generalization technique.

$$\text{INV} \equiv z \leq x/y < z + d$$

*Establish* the invariant.

```
{0 ≤ x < y ≤ 1}
begin
z := 0;
d := 1;
while d > t
  {z < x/y < z + d}
  do
    improve z and d
  end
{z ≤ x/y < z + t}
```

Let's try a *binary search* (!?) If  $z + \frac{1}{2}d > x/y$ , then  $z \leq x/y < z + \frac{1}{2}d$ ; otherwise,  $z + \frac{1}{2}z \leq \frac{z}{y} < z + d$ . Either way, we know the quotient to within  $\frac{1}{2}d$ , so

```
{0 ≤ x < y ≤ 1}
begin
z := 0;
d := 1;
while d > t do
  {z < x/y < z + d}
  begin
    if z +  $\frac{1}{2}d > x/y$ 
      then z := z
      else z := z +  $\frac{1}{2}d$ ;
    d :=  $\frac{1}{2}d$ 
  end
end
{z ≤ x/y < z + t}
```

Fix the “cheat” in the `if`-test. Get rid of the division by multiplying through by  $y$ .

```
{0 ≤ x < y ≤ 1}
begin
z := 0;
d := 1;
while d > t do
  {z < x/y < z + d}
  begin
    if zy +  $\frac{1}{2}dy > x$ 
      then skip
      else z := z +  $\frac{1}{2}d$ ;
      d :=  $\frac{1}{2}d$ 
    end
  end
end
{z ≤ x/y < z + t}
```

The test seems costly. Introduce “*trailer variables*”  $u$  and  $v$  to represent of the multiplications, subject to invariants  $u = zy$  and  $v = \frac{1}{2}dy$ .

This technique is called *strength reduction*.

Now we are obligated to maintain these invariants, depending on the test:

Case A:  $u + v > x$ :

$$d' = \frac{1}{2}d$$

$$z' = z$$

$$u' = z' \cdot y = z \cdot y = u$$

$$v' = \frac{1}{2}(d') \cdot y = \frac{1}{2}(\frac{1}{2}d)y = \frac{1}{2}v$$

Case B:  $u + v \leq x$ :

$$d' = \frac{1}{2}d$$

$$z' = z + \frac{1}{2}d = z + d'$$

$$u' = z' \cdot y = (z + \frac{1}{2}d)y = zy + \frac{1}{2}dy = u + v$$

$$v' = \frac{1}{2}(d') \cdot y = \frac{1}{2}(\frac{1}{2}d)y = \frac{1}{2}v$$

```

{0 ≤ x < y ≤ 1}
begin
z := 0;
d := 1;
u := 0;
v := ½y; ] Establish the invariant
while d > t do
  {z < x/y < z + d ∧ u = zy ∧ v = ½dy}
  begin
    d := ½d;

    if u + v > x
      then skip
      else
        begin
          z := z + d;
          u := u + v;
        end;

        v := ½v
      end
    end
  end
  {z ≤ x/y < z + t}

```

PROOF: *by “construction.”* We have synthesized a program from its specification. If the steps are correct, so is the program.

NOTE: This is called *Wensley's algorithm* [Wensley 58] for real number division, specifically computing the fractional parts of floating point representations. Since it reduces integer division to addition and divide-by-two, it is a candidate for use in a typical processor, which will have instructions for these operations.

This version of Wensley's algorithm is not suitable for implementation in hardware because addition takes too long (at least  $\mathcal{O}(\log n)$  for  $n$ -bit operands) this can be optimized (by further strength reduction), but better algorithms exist for hardware division.