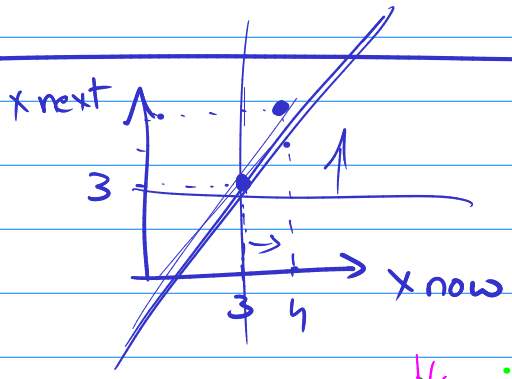
Lecture 20

The Loop Rule

$$\frac{\{\varphi \wedge b\}\ P\ \{\varphi\}}{\{\varphi\}\ \text{while } (b) \text{ do } P\ \{\varphi \wedge \neg b\}}$$

An inductive invariant is a predicate φ such that this holds



Ex: ① z := input();

② x := 4;

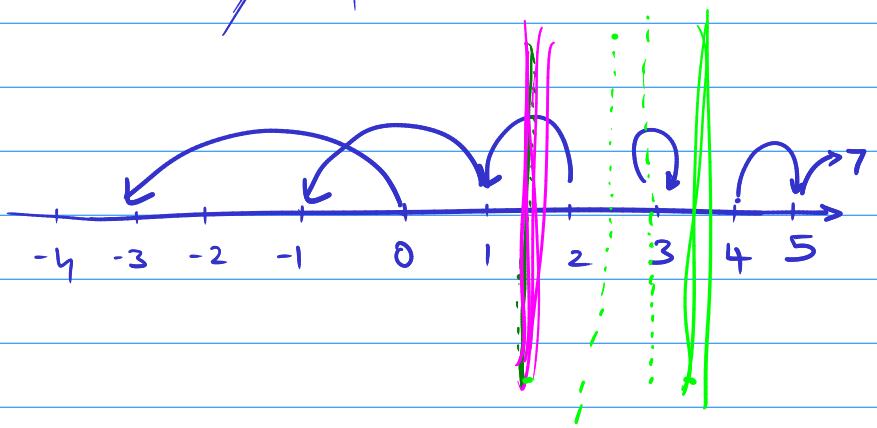
③ while (z > 0) {

④ x := 2x − 3

⑤ z := z − 1

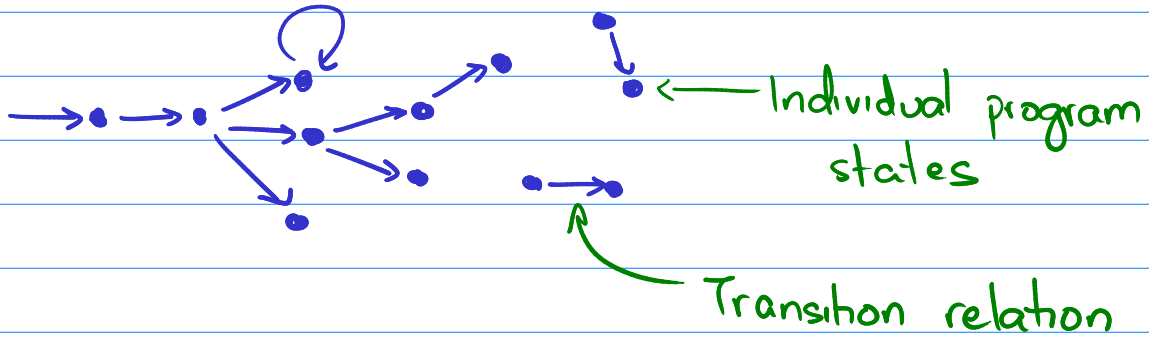
}

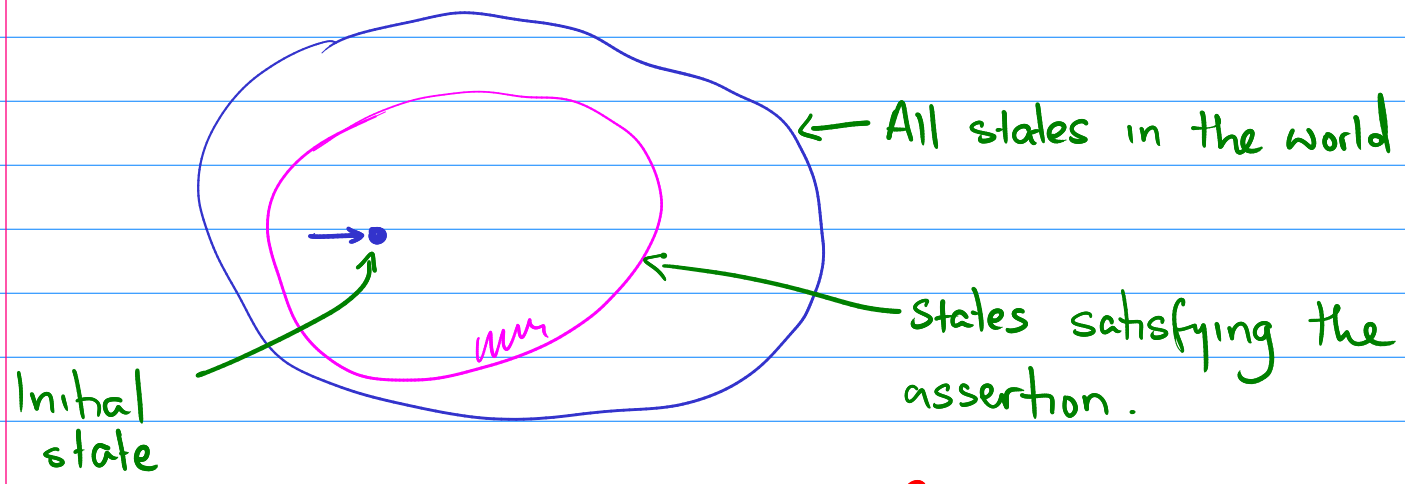
⑥ assert(x > 1);



Claim: If $x \geq 3$, its value monotonically increases in each iteration.

State space of the program





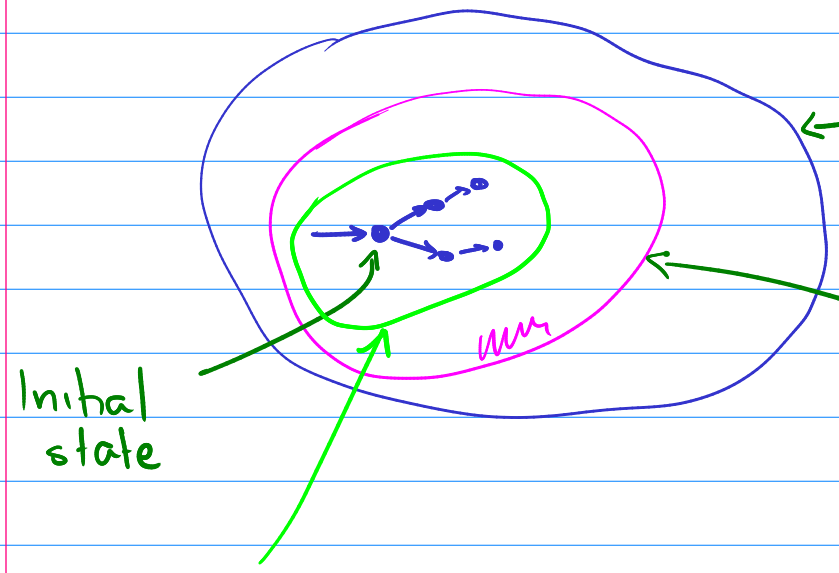
PC ≠ ⑥ or x > 1

PC = ⑥ ⇒ x > 1

The verification problem is to make sure that all reachable states lie inside the pink boundary.

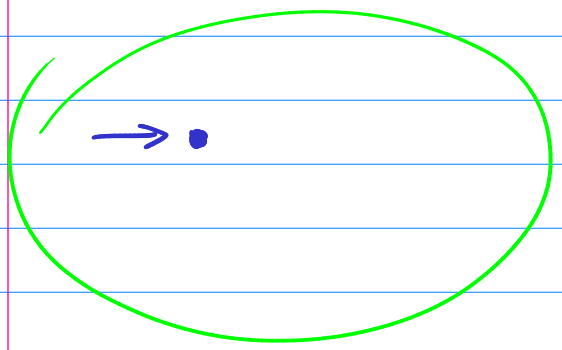
The verification problem seems hard.

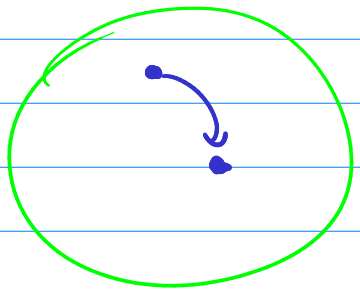
What if we instead show that there is a green boundary such that:



$PC \neq 6$ or $x > 1$

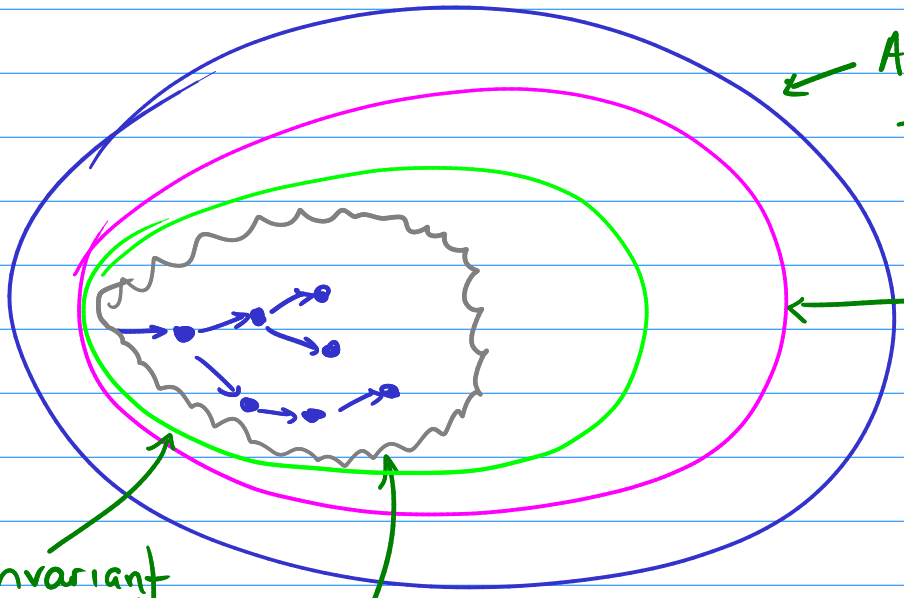
$PC = 6 \Rightarrow x > 1$



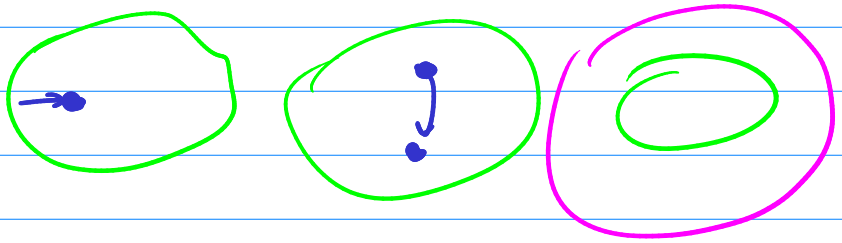


(1) The inital state is in side the green boundary

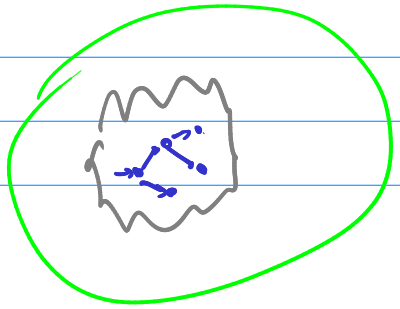
(2) If we are inside the green boundary now, we will remain inside the green boundary even in the next step

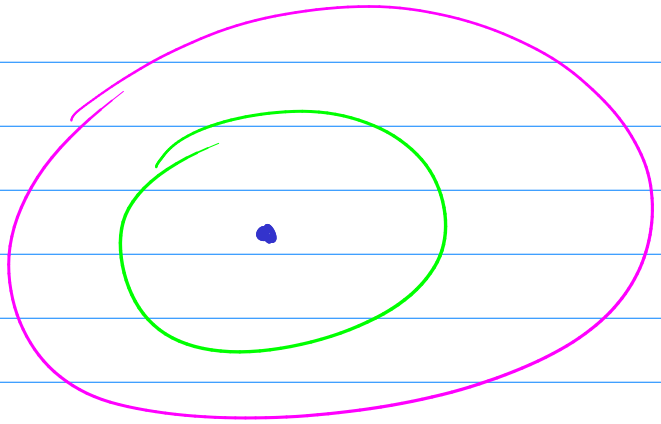


The verification problem asks whether  We have shown.



Corollary: The grey boundary is contained within the green boundary if the proof succeeds.

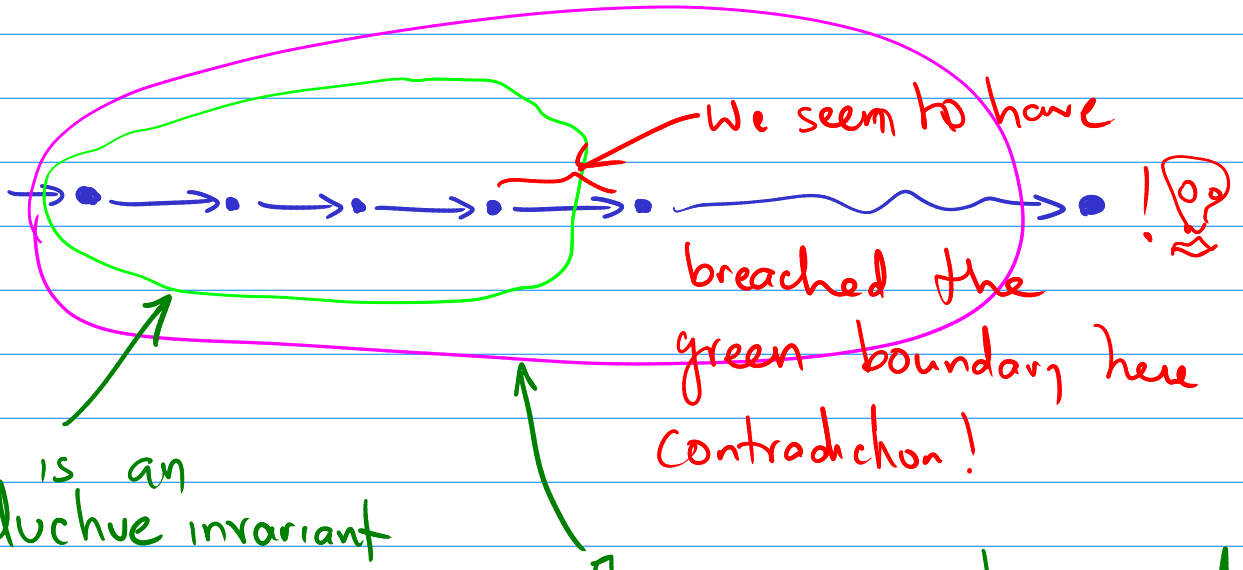


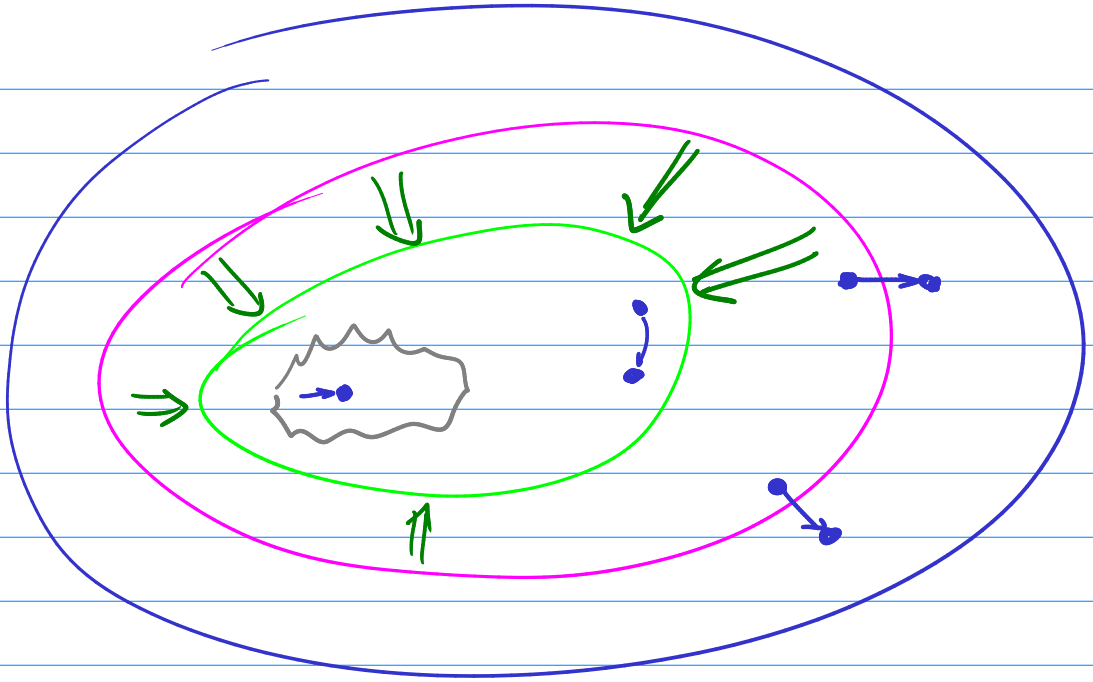


& (3) every state inside the green boundary is also inside the pink boundary.

Claim: If such a green boundary exists, then the system is safe forever.

Proof. Assume not. Let's look at the execution path leading to unsafe behavior.





Program Termination (Total correctness)

Ex:

```
n := input()
x := 0;
while (x < n) {
    x := x + 1
}
```

The program executes the loop $\max(n, 0)$ times.

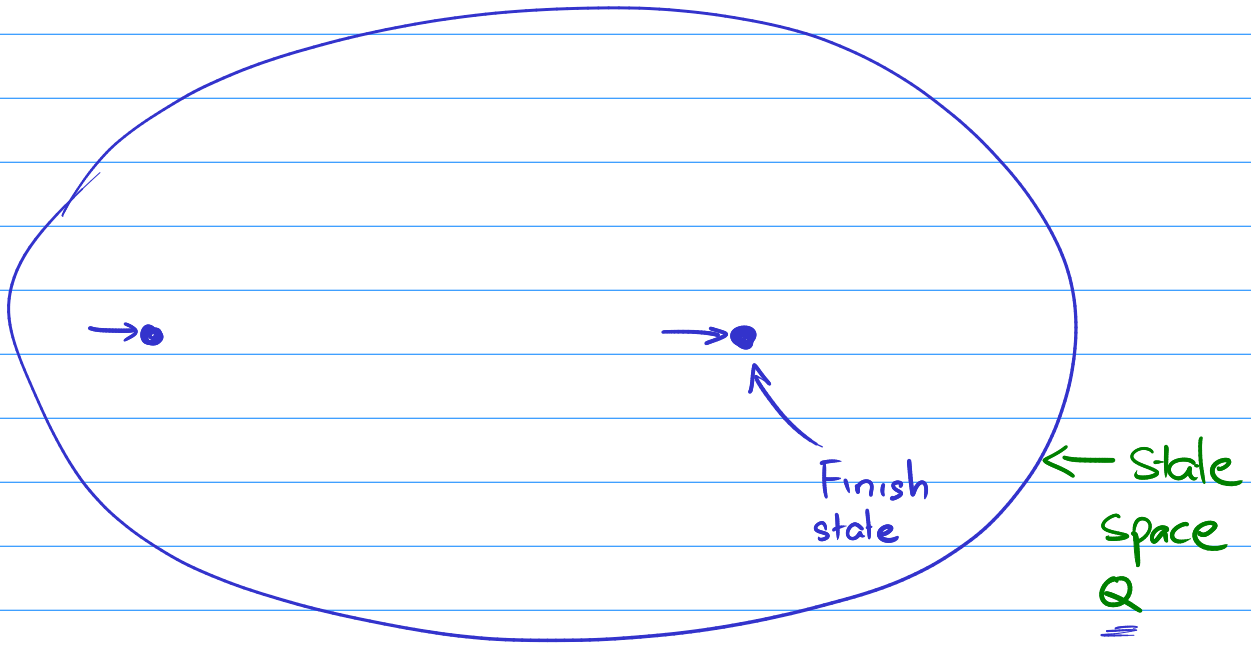
$$f(q) = \max(n - x, 0)$$

Ex:

```
for (i = 0; i < |arr|; i++)
    for (j = 0; j < i - 1; j++)
        if (arr_j > arr_i)
            swap(arr_i, arr_j)
```

← This block is executed $|arr| \cdot \frac{|arr| - 1}{2}$ times

$f(q) =$ ~~~



One approach to termination

Ranking function: $f : Q \to \mathbb{N}$

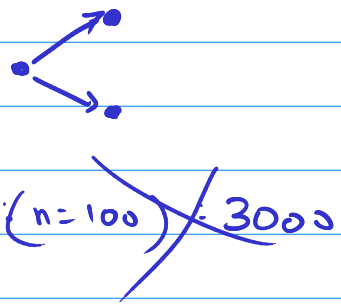
- Q: Tell me the current state
- $\mathbb{N}$: # of steps needed to terminate.

- $f(q) \geq 0$,
- If $q \to q'$, then $f(q) > f(q')$
- $f(q) = 0 \Rightarrow q$ is the finish state

Examples where $f: Q \to \mathbb{N}$ is insufficient for ranking arguments

Ex: `n := input()` ⟸ How much fuel will I need if $n = 100$?

```
for (x=0; x<n; x++) {
    m := input()
    for (y=0; y<m; y++) {
        z := z+1
    }
}
```

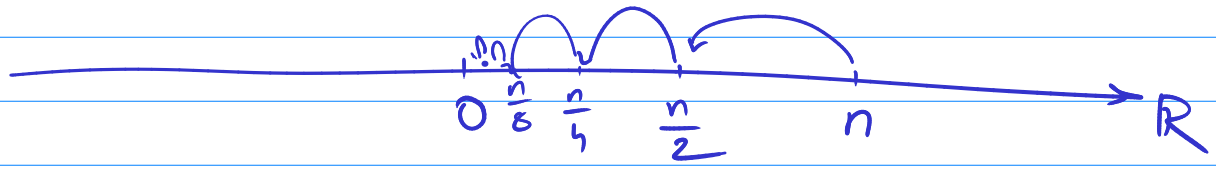


Possible alternative approach to ranking fns

$f: Q \to \mathbb{R}$ ← real number

$-f(q) \geq 0$; If $q \to q'$ then $f(q) \not\geq f(q')$, $f(q) = 0 \Rightarrow$ program has terminated.

Problem with Zeno behavior



$f(x) = \frac{x}{2}$

```
while (x > 0) {
    x := x/2
}
```

Well-orders State space Q

- The program eventually terminates
- I will show you a total order $\leq$ over Q such that there are no infinite ascending chains. $q_0 \lneq q_1 \lneq q_2 \lneq q_3 \lneq \cdots \infty$