Lecture 19

Partial correctness / Safety / Have finite counterexamples, if any.

assert (φ) ← Does φ always hold?

Axioms for partial correctness of loop-free code

Axiom 1:

$$\frac{}{\{\varphi[e/x]\}\ x := e\ \{\varphi\}}$$

Axiom 2:

$$\frac{\varphi' \Rightarrow \varphi \qquad \{\varphi\}\ P\ \{\psi\}}{\{\varphi'\}\ P\ \{\psi\}}$$

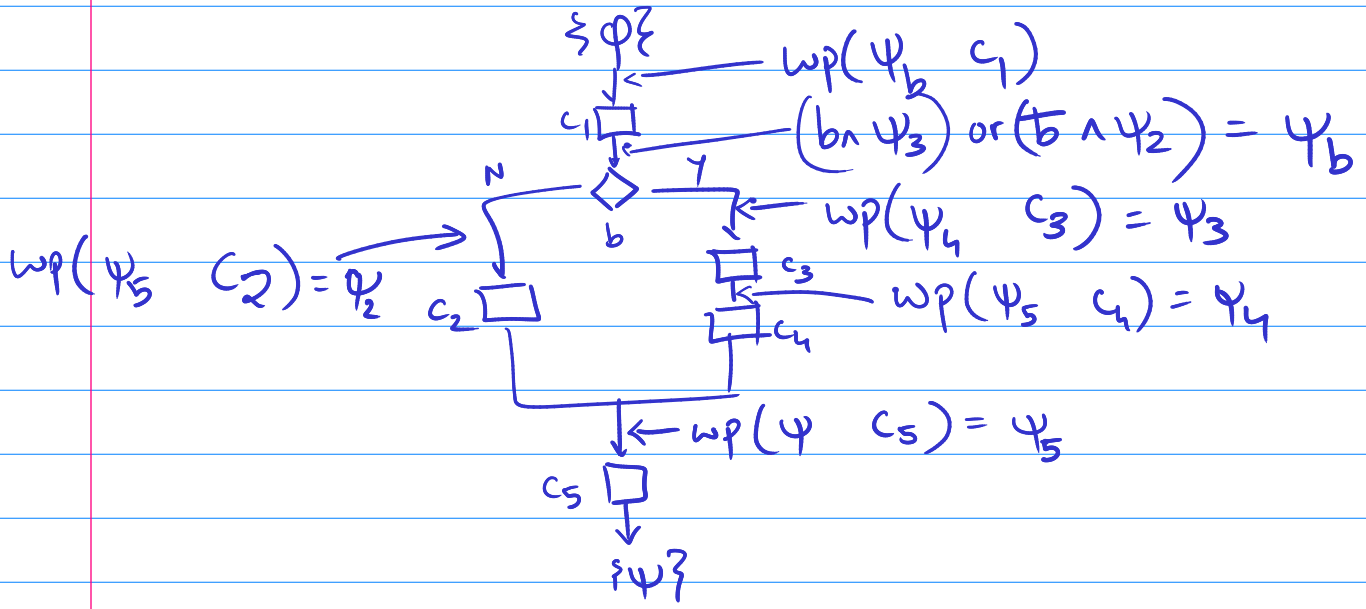
Axiom 3:

$$\frac{\{\varphi\}\ P\ \{\psi\} \qquad \psi \Rightarrow \psi'}{\{\varphi\}\ P\ \{\psi'\}}$$

Axiom 4 $\dfrac{\{\varphi\}\ P\ \{\psi\} \qquad \{\psi\}\ Q\ \{\chi\}}{\{\varphi\}\ P;Q\ \{\chi\}}$

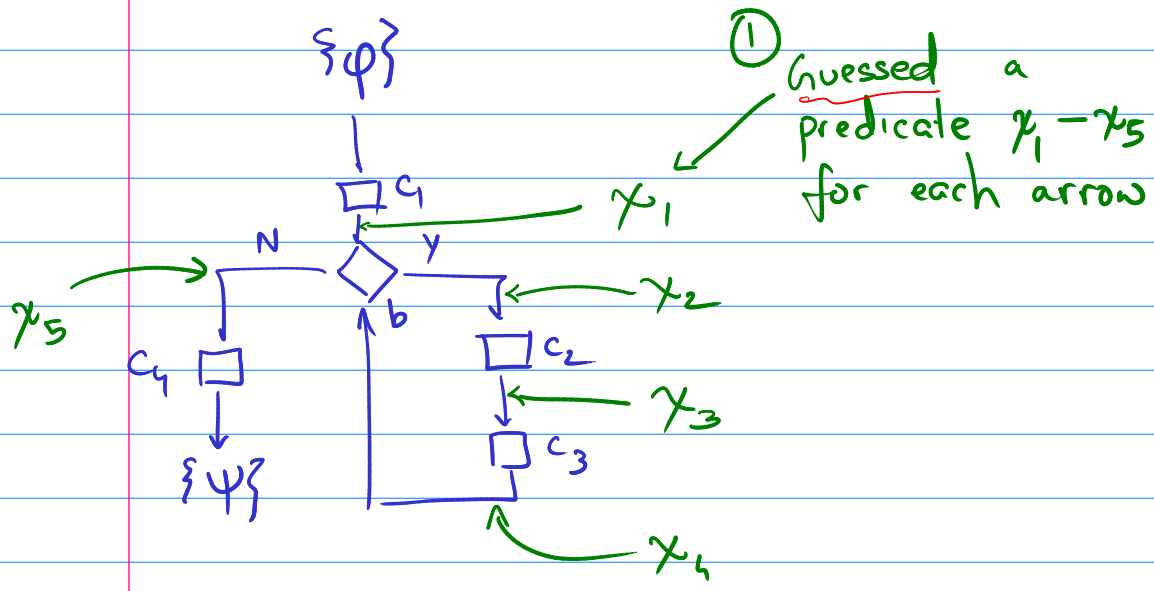
Axiom 5: $\dfrac{}{\{\varphi\}\ \text{skip}\ \{\varphi\}}$

Axiom 6: $\dfrac{\{\varphi \wedge b\}\ P\ \{\psi\} \qquad \{\varphi \wedge \neg b\}\ Q\ \{\psi\}}{\{\varphi\}\ \text{if } b \text{ then } P \text{ else } Q\ \{\psi\}}$



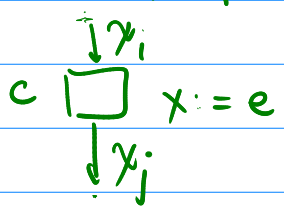
Finally, check whether $\varphi \Rightarrow wp(\psi_b\ c_1)$

Guess-and-check approach to verifying code with loops

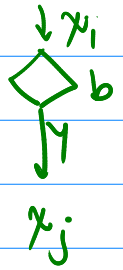


① Guessed a predicate $\chi_1 - \chi_5$ for each arrow

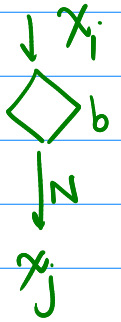
② Verify, for each (Verification conditions)



$\{\chi_i\}\ x := e\ \{\chi_j\}$



$\chi_i \wedge b \Rightarrow \chi_j$



$\chi_i \wedge \neg b \Rightarrow \chi_j$

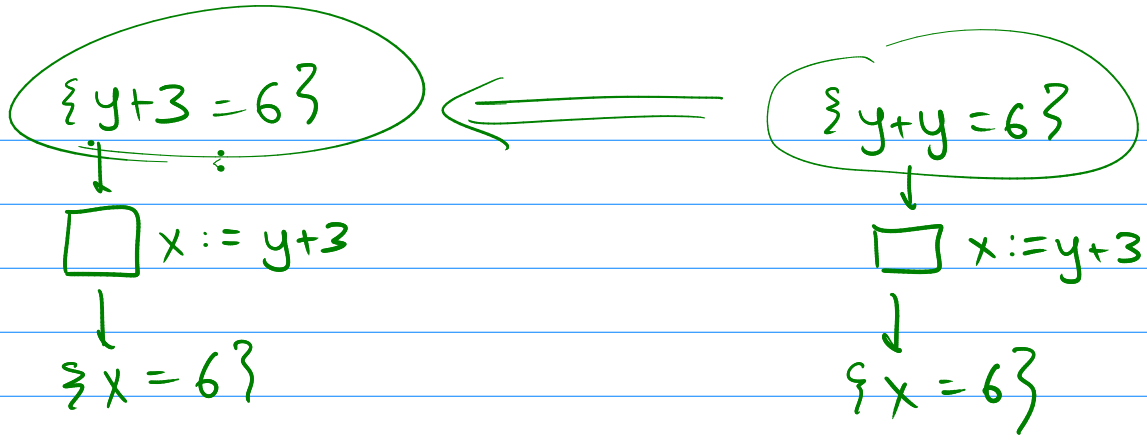
Question: Is it not sufficient to check whether

$\chi_i \stackrel{?}{=} \chi_j[e/x]$?

↑ syntactic equality) Not always

$\chi_i \Rightarrow \chi_j[e/x]$.

Immediately submit to SMT solver



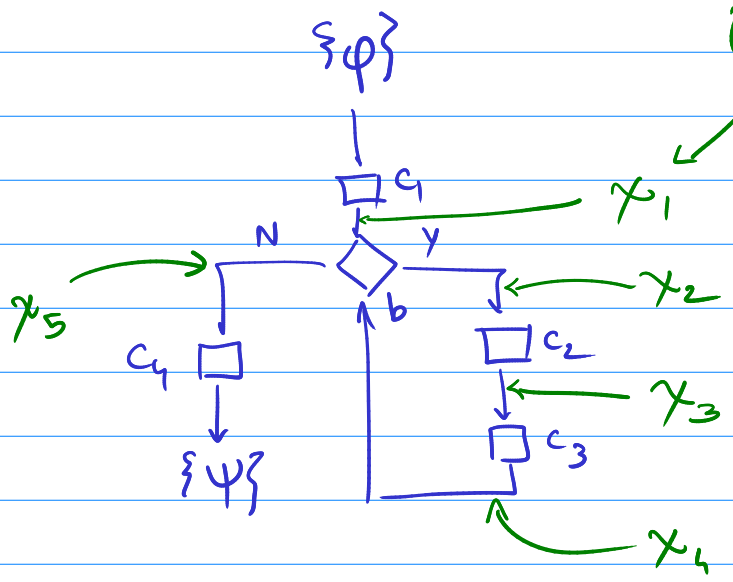
$(x=6)[y+3/x] = y+3=6$

$$\dfrac{y+y=6 \Rightarrow y+3=6 \qquad \dfrac{}{\{y+3=6\}\ x:=y+3\ \{x=6\}}\ \text{Axiom 1}}{\{y+y=6\}\ x:=y+3\ \{x=6\}}\ \text{Axiom 2}$$

Step 1: Observe that $\{y+3=6\}\ x:=y+3\ \{x=6\}$, by Axiom 1.

Step 2. Apply Axiom 2 to step 1 & the observation that $y+y=6 \Rightarrow y+3=6$.

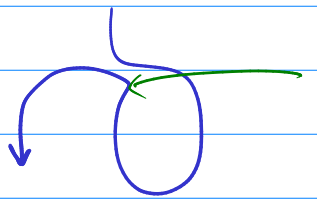
It follows that $\{y+y=6\}\ x:=y+3\ \{x=6\}$.



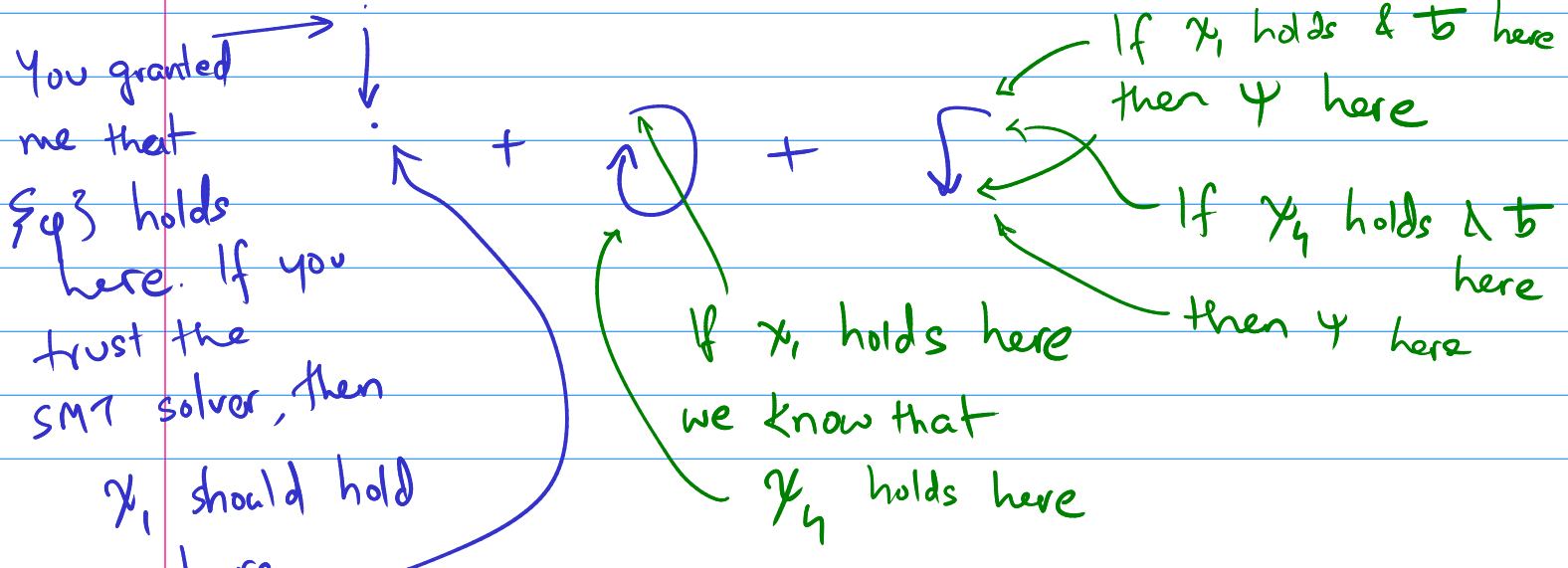
① Guessed a predicate $\chi_1 - \chi_5$ for each arrow

Remember: How you have to guess the induction hypothesis while doing induction.

Question: How do we know that the program never crashes if the control flow path looks like



=

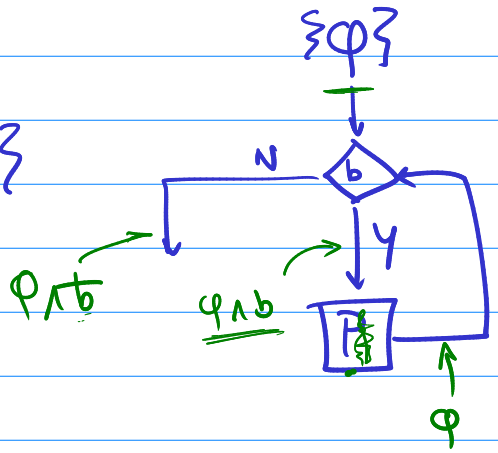


Floyd Hoare logic

Axiom 7: $\{\varphi \wedge b\}\ P\ \{\varphi\}$

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

Invariant

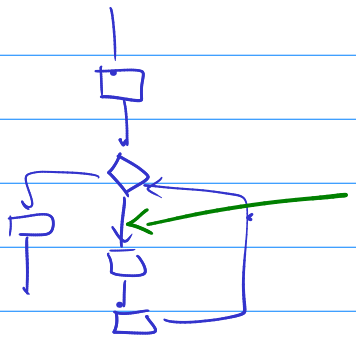


For relative completeness of Hoare logic, see

Glynn Winskel's textbook

"Formal Semantics of Programming Languages".

Invariant vs Inductive Invariant.



say φ always holds here.

Doesn't matter how you know.

φ is an "invariant"

A invariant is a predicate φ which is always true at some program point

A loop invariant φ is inductive if it satisfies $\{\varphi \wedge b\}\ P\ \{\varphi\}$

while (b) do P.