Loop Invariant and While Loop Rule (continue)

- A loop invariant for $W \equiv \textbf{while } B \textbf{ do } S \textbf{ od}$ is a predicate p such that $\vDash \{p \wedge B\}\, S\, \{p\}$. It follows that $\vDash \{p\}\, W\, \{p \wedge \neg B\}$.
  - In other words, a loop invariant is a predicate that is True, before and after each iteration of the loop.
  - To indicate the loop invariant, we write an additional clause in the program: $W \equiv \{\textbf{inv } p\} \textbf{ while } B \textbf{ do } S \textbf{ od}$.

- While Loop Rule:
  1. $\{p \wedge B\}\, S\, \{p\}$
  2. $\{p\} \textbf{ while } B \textbf{ do } S \textbf{ od } \{p \wedge \neg B\}$ loop 1

  - The precondition p and postcondition $p \wedge \neg B$ are not the weakest precondition and the strongest postcondition, but they are the "best" precondition and postcondition we can find for a provable triple.

1. Which of the following predicate p can be a loop invariant for $W \equiv \{\textbf{inv } p\} \textbf{ while } k < n \textbf{ do } k := k + 1; s := s + k \textbf{ od}$
   a. $p \equiv T$ Yes.
   b. $p \equiv F$ Yes.
   c. $p \equiv 0 \le k \le n$ Yes, $p \wedge B \Leftrightarrow 0 \le k < n$, so after $k := k + 1$, we still have p

- From the above example, we can see that not all loop invariants provide us useful information: it cannot be too weak or too strong. If we have a loop $W \equiv \textbf{while } B \textbf{ do } S \textbf{ od}$, and we need $\vDash \{p_0\}\, W\, \{q\}$, then we need a loop invariant p such that:
  1) $p_0 \Rightarrow p$
  2) $\vDash \{p \wedge B\}\, S\, \{p\}$
  3) $p \wedge \neg B \Rightarrow q$

  In future classes, we will discuss more on how to find a good loop invariant.

2. Show that p is a loop invariant for $W \equiv \textbf{while } k < n \textbf{ do } k := k + 1; s := s + k \textbf{ od}$, where $p \equiv 0 \le k \le n \wedge s = sum(0, k)$, and $sum(0, k) \equiv \sum_{i=0}^{k} i$. Find a q then create a formal proof for provable triple $\{p\}\, W\, \{q\}$.

   - To show p is loop invariant, we need to prove triple $\{p \wedge k < n\}\, S\, \{p\}$. We can create the following proof.

   1. $\{p[s + k \,/\, s]\}\ s := s + k\, \{p\}$ backward assignment
   2. $\{p[s + k \,/\, s]\,[k + 1 \,/\, k]\}\, k := k + 1\, \{p\,[s + k \,/\, s]\}$ backward assignment
   3. $\{p[s + k \,/\, s]\,[k + 1 \,/\, k]\}\, k := k + 1; s := s + k\, \{p\}$ sequence 2,1
   4. $p \wedge k < n \Rightarrow p[s + k \,/\, s]\,[k + 1 \,/\, k]$ predicate logic

      $\#\ p \wedge k < n \Leftrightarrow 0 \le k < n \wedge s = sum(0, k)$

      $\#\ p[s + k \,/\, s]\,[k + 1 \,/\, k] \equiv \big(0 \le k \le n \wedge s + k = sum(0, k)\big)[k + 1 \,/\, k]$

      $\equiv 0 \le (k + 1) \le n \wedge s + k + 1 = sum(0, k + 1)$

   5. $\{p \wedge k < n\}\, k := k + 1; s := s + k\, \{p\}$ strengthen precondition 4,3
   6. $\{\textbf{inv } p\}\, W\, \{p \wedge k \ge n\}$ loop 5

3. Prove the following triple under partial correctness.

$\{n \geq 0\}$
$k \coloneqq 0;\ s \coloneqq 0;$
$\{\textbf{inv}\ p \equiv 0 \leq k \leq n \wedge s = sum(0,k)\}$
while $k < n$ **do**
    $s \coloneqq s + k + 1;\ k \coloneqq k + 1$
od
$\{s = sum\ (0,n)\}$

- Informally, to prove the above program we need to prove the following triples/predicates:
  $\{n \geq 0\}\ k \coloneqq 0;\ s \coloneqq 0\ \{p\}$
  $\{p \wedge\ k < n\}\ s \coloneqq s + k + 1;\ k \coloneqq k + 1\ \{p\}$ (so that we can prove the loop)
  $p \wedge k \geq n \Rightarrow s = sum(0,n)$

  We can create the following proof.

1. $\{n \geq 0\}\ k \coloneqq 0\ \{n \geq 0 \wedge k = 0\}$	forward assignment
2. $\{n \geq 0 \wedge k = 0\}\ s \coloneqq 0\ \{n \geq 0 \wedge k = 0\ \wedge\ s = 0\}$	forward assignment
3. $\{n \geq 0\}\ k \coloneqq 0;\ s \coloneqq 0\ \{n \geq 0 \wedge k = 0 \wedge s = 0\}$	sequence 1,2
4. $n \geq 0 \wedge k = 0 \wedge s = 0 \rightarrow p$	predicate logic
# Where $p \equiv\ 0 \leq k \leq n\ \wedge\ s = sum\ (0\ ,k\ )$	
5. $\{n \geq 0\}\ k \coloneqq 0;\ s \coloneqq 0\ \{p\}$	weaken postcondition 3,4
6. $\{p[k+1\ /\ k]\}\ k \coloneqq k + 1\ \{p\}$	backward assignment
7. $\{p[k+1\ /\ k][s+k+1\ /\ s]\}\ s \coloneqq s + k + 1\ \{p[k+1\ /\ k]\}$	backward assignment
8. $\{p[k+1\ /\ k][s+k+1\ /\ s]\}\ s \coloneqq s + k + 1;\ k \coloneqq k + 1\{p\}$	sequence 7,6
9. $p \wedge\ k < n \rightarrow p[k+1\ /\ k][s+k+1\ /\ s]$	predicate logic
# $p \wedge\ k < n \Leftrightarrow 0 \leq k < n\ \wedge\ s = sum\ (0\ ,k)$	
# $p[k+1\ /\ k][s+k+1\ /\ s] \equiv \big(0 \leq k+1 \leq n \wedge s = sum(0,k+1)\big)\ [s+k+1\ /\ s]$	
$\equiv 0 \leq k+1 \leq n \wedge s+k+1 = sum(0,k+1)$	
10. $\{p \wedge\ k < n\}\ s \coloneqq s + k + 1;\ k \coloneqq k + 1\{p\}$	strengthen precondtion 9, 8
11. $\{\textbf{inv}\ p\}\ W\ \{p \wedge k \geq n\}$	loop 10
# Where $W \equiv$ **while** $k < n$ **do** $s \coloneqq s + k + 1;\ k \coloneqq k + 1$ **od**	
12. $\{n \geq 0\}\ k \coloneqq 0;\ s \coloneqq 0; W\ \{p \wedge k \geq n\}$	sequence 5,11
13. $p \wedge\ k \geq n \rightarrow s = sum\ (0,n)$	predicate logic
# $p \wedge\ k \geq n \Leftrightarrow 0 \leq k = n\ \wedge\ s = sum\ (0\ ,k)$	
14. $\{n \geq 0\}\ k \coloneqq 0;\ s \coloneqq 0; W\ \{sum\ (0,n)\}$	weaken postcondition 12, 13

Full Proof Outlines

- A formal proof can be very long and contains repetitive information. A ***proof outline*** is a way to write out all the information that you would need to generate full formal proof, but with less repetition, so they're much shorter, and they don't mask the overall structure of the program the way a full proof does.

- Before studying how to write a ***full proof outline***, let us look at an example. Here we give the full proof outline for the program we proved in Question 3; I use a different color to show the parts that are added to the program.

4. Give a full proof outline for the program in Question 3.

$\{n \geq 0\}$
$k \coloneqq 0; \{n \geq 0 \wedge k = 0\}\ s \coloneqq 0; \{n \geq 0 \wedge k = 0 \wedge s = 0\}$
$\{\textbf{inv}\ p \equiv 0 \leq k \leq n \wedge s = sum(0\,, k)\}$
while $k < n$ **do**
    $\{p \wedge\ k < n\}$
    $\{p\ [k+1\ /\ k][s+k+1\ /\ s]\}\ s \coloneqq s + k + 1; \{p[k+1\ /\ k]\}\ k \coloneqq k + 1\ \{p\}$
od
$\{p \wedge k \geq n\}$
$\{s = sum\ (0, n)\}$

- To get a full proof outline, we annotate statements with their preconditions and postconditions, so that :
  - *Every statement S in the program is wrapped in a triple.*
  - *Every triple $\{p\}\ S\ \{q\}$ in the full proof outline is provable.*
  - *Every pair of adjacent conditions $\{p\}\{q\}$ has that $p \Rightarrow q$.*

- Here are the rules to create proof outlines for individual statements:
  - Assignment and **skip** statements that are proved by axiom are handled just as they are in the axiom:
    - $\{p\}\ v \coloneqq e\ \{q\}$
    - $\{p\}$ **skip** $\{p\}$

  - Sequence statement that combines $\{p\}\ S_1\ \{q\}$ and $\{q\}\ S_2\ \{r\}$ and getting $\{p\}\ S_1; S_2\ \{r\}$ is written as:
    - $\{p\}\ S_1;\ \{q\}\ S_2\ \{r\}$
  - Sequence statement that combines $\{p\}\ S_1\ \{q_1\}$ and $\{p_2\}\ S_2\ \{r\}$ (so $q_1 \Rightarrow p_2$) and getting $\{p\}\ S_1; S_2\ \{r\}$ is written as:
    - $\{p\}\ S_1;\ \{q_1\}\{p_2\}\ S_2\ \{r\}$

  - To express the while loop rule that loops through $\{p \wedge B\}\ S\ \{p\}$ to get $\{\textbf{inv}\ p\}$ **while** B **do** S **od** $\{p \wedge \neg B\}$, we write:
    - $\{\textbf{inv}\ p\}$ **while** B **do** $\{p \wedge B\}\ S\ \{p\}$ **od** $\{p \wedge \neg B\}$

  - For conditional rule 1 we write:
    - $\{p\}$ **if** B **then** $\{p \wedge B\}\ S_1\ \{q_1\}$ **else** $\{p \wedge \neg B\}\ S_2\ \{q_2\}$ **fi** $\{q_1 \vee q_2\}$
  - For conditional rule 2 we write:
    - $\{(B \rightarrow p_1) \wedge (\neg B \rightarrow p_2)\}$ **if** B **then** $\{p_1\}\ S_1\ \{q_1\}$ **else** $\{p_2\}\ S_2\ \{q_2\}$ **fi** $\{q_1 \vee q_2\}$
  - For nondeterministic conditional rule we write:
    - $\{p\}$ **if** $B_1 \rightarrow \{p \wedge B_1\}\ S_1\ \{q_1\}\ \square\ B_2 \rightarrow \{p \wedge B_2\}\ S_2\ \{q_2\}$ **fi** $\{q_1 \vee q_2\}$

  - To strengthen the precondition of $\{p\}\ S\ \{q\}$ with $p_1 \Rightarrow p$, we write:
    - $\{p_1\}\{p\}\ S\ \{q\}$
  - To weaken the postcondition of $\{p\}\ S\ \{q\}$ to $q \Rightarrow q_1$, we write:
    - $\{p\}\ S\ \{q\}\{q_1\}$

5. Create a full proof outline of the following formal proof.

1. $\{T\}\ k \coloneqq 0\ \{k = 0\}$	forward assignment
2. $\{k = 0\}\ x \coloneqq 1\ \{x = 1 \wedge k = 0\}$	forward assignment
3. $\{T\}\ k \coloneqq 0; x \coloneqq 1\ \{x = 1 \wedge k = 0\}$	sequence 1,2
4. $x = 1 \wedge k = 0 \rightarrow k \geq 0 \wedge x = 2^k$	predicate logic
5. $\{T\}\ k \coloneqq 0; x \coloneqq 1\ \{k \geq 0 \wedge x = 2^k\}$	weaken postcondition 3,4

- We can create a full proof outline following the proof: there is assignment, assignment, sequence and weaken postcondition; so, we can create:
  $\{T\}\ k \coloneqq 0; \{k = 0\}\ x \coloneqq 1\ \{x = 1 \wedge k = 0\}\{k \geq 0 \wedge x = 2^k\}$

6. Create formal proof for the following full proof outline:

$$\{T\}\{0 \geq 0 \wedge 1 = 2^0\}\ k \coloneqq 0; \{k \geq 0 \wedge 1 = 2^k\}\ x \coloneqq 1\ \{k \geq 0 \wedge x = 2^k\}$$

   - There are two statements, and they are in sequence.

$1.\{k \geq 0 \wedge 1 = 2^k\}\ x \coloneqq 1\ \{k \geq 0 \wedge x = 2^k\}$	backward assignment
$2.\{0 \geq 0 \wedge 1 = 2^0\}\ k \coloneqq 0\ \{k \geq 0 \wedge 1 = 2^k\}$	backward assignment
$3.\{0 \geq 0 \wedge 1 = 2^0\}\ k \coloneqq 0;\ x \coloneqq 1\ \{k \geq 0 \wedge x = 2^k\}$	sequence 2,1
$4.T \rightarrow 0 \geq 0 \wedge 1 = 2^0$	predicate logic
$5.\{T\}\ k \coloneqq 0;\ x \coloneqq 1\ \{k \geq 0 \wedge x = 2^k\}$	strengthen precondition 4,3

- Both Examples 5 and 6 are proving the triple $\{T\}\ k \coloneqq 0;\ x \coloneqq 1\ \{k \geq 0 \wedge x = 2^k\}$, but we have different formal proofs and different proof outlines. The reason here is that the forward assignment and the backward assignment are equally powerful.
  In general, a program can be proved in different ways. For each formal proof, there is a corresponding full proof outline.