Weakest Preconditions (continue)

1. Let $flip \equiv \textbf{if } T \rightarrow x \coloneqq 0 \ \Box\ T \rightarrow x \coloneqq 1 \textbf{ fi}$, $head \equiv x = 0$, and $tail \equiv x = 1$.
   a. What is $M(flip, \emptyset)$?
      $M(flip, \emptyset) = \{\{head\},\ \{tail\}\}$.

   b. What is $wp(flip,\ head \vee tail)$?
      For any state σ (let's assume that x is not defined in σ to simply the notation), we have $M(flip, \sigma) = \{\{\sigma \cup head\},\ \{\sigma \cup tail\}\}$, and it satisfies $head \vee tail$, thus $wp(flip,\ head \vee tail) \Leftrightarrow T$.

   c. What is $wp(flip,\ head)$? And what is $wp(flip,\ tail)$?
      For any state σ, we have $M(flip, \sigma) = \{\{\sigma \cup head\},\ \{\sigma \cup tail\}\}$, it doesn't satisfy $head$ and it doesn't satisfy $tail$; thus $wp(flip,\ head) \Leftrightarrow wp(flip,\ tail) \Leftrightarrow F$.

Calculate wlp for Loop-Free Programs

- We start with the calculations of wlp in loop-free programs because: 1) if a program is loop-free (also error-free), then $wlp \Leftrightarrow wp$ 2) if a program can create runtime errors, then we can add "error-avoiding information" to convert wlp to wp. 3) We will handle wlp for loops in the future.

- The following algorithm takes S and q and calculates a predicate for $wlp(S, q)$. Since this calculation procedure is "robotic" or syntactical, so we use $\equiv$ instead of $=$ or $\Leftrightarrow$ here.
  - $wlp\ (\textbf{skip}, q) \equiv q$.
  - $wlp\ (v \coloneqq e, Q(v)) \equiv Q(e)$, where Q is a predicate function.
    - We have learned this backward assignment rule, and in general this operation that takes us from $Q(v)$ to $Q(e)$ is called syntactic substitution; we will study this carefully in the future.

  - $wlp(S_1; S_2, q) \equiv wlp\ (S_1, wlp(S_2, q))$.
  - $wlp\ (\textbf{if } B \textbf{ then } S_1 \textbf{ else } S_2 \textbf{ fi}, q) \equiv (B \rightarrow wlp(S_1, q)) \wedge (\neg B \rightarrow wlp(S_2, q))$.
  - $wlp\ (\textbf{if } B_1 \rightarrow S_1 \ \Box\ B_2 \rightarrow S_2 \textbf{ fi}, q) \equiv (B_1 \rightarrow wlp(S_1, q)) \wedge (B_2 \rightarrow wlp(S_2, q))$.

2. Calculate the following weakest liberal preconditions.
   a. $wlp(x \coloneqq x + 1,\ x \geq 0) \equiv (x + 1 \geq 0) \Leftrightarrow x \geq -1$

   b. $wlp\ (y \coloneqq y + x;\ x \coloneqq x + 1,\ x \geq 0) \equiv wlp\ (y \coloneqq y + x,\ x + 1 \geq 0) \equiv x + 1 \geq 0$

   c. $wlp\ (y \coloneqq y + x;\ x \coloneqq x + 1,\ x \geq y) \equiv wlp\ (y \coloneqq y + x,\ x + 1 \geq y) \equiv x + 1 \geq y + x \Leftrightarrow y \leq 1$

   d. $wlp\ (x \coloneqq x + 1;\ y \coloneqq y + x,\ x \geq y) \equiv wlp\ (x \coloneqq x + 1,\ x \geq y + x) \equiv x + 1 \geq y + x + 1 \Leftrightarrow y \leq 0$

   e. $wlp\ (\textbf{if } y \geq 0 \textbf{ then } x \coloneqq y \textbf{ fi}, x \geq 0)$
      $\equiv wlp\ (\textbf{if } y \geq 0 \textbf{ then } x \coloneqq y \textbf{ else skip fi}, x \geq 0)$
      $\equiv (y \geq 0 \rightarrow y \geq 0) \wedge (y < 0 \rightarrow x \geq 0)$
      $\Leftrightarrow T \wedge (y < 0 \rightarrow x \geq 0)$
      $\Leftrightarrow y < 0 \rightarrow x \geq 0$
      $\Leftrightarrow y \geq 0 \vee x \geq 0$

f. $wlp\ (\mathbf{if}\ y \geq 0 \rightarrow x \coloneqq y \ \square\ x < 0 \rightarrow x \coloneqq y + 1\ \mathbf{fi}, x \geq 0)$ $\equiv (y \geq 0 \rightarrow y \geq 0) \wedge (x < 0 \rightarrow y + 1 \geq 0)$
$\Leftrightarrow x < 0 \rightarrow y \geq -1 \Leftrightarrow x \geq 0 \vee y \geq -1$

Avoid Runtime Error in Expressions

- Runtime errors can appear while evaluating expressions. To avoid such errors while calculating $\sigma(e)$, we define **domain predicate** $D(e)$ such that $\sigma \vDash D(e)$ implies $\sigma(e) \neq \perp_e$.
  - For example, to avoid runtime error, we can define $D\big(b\big[b[k]\big]\big) \equiv 0 \leq k < size(b) \wedge 0 \leq b[k] < size(b)$.
  - Here is another example, we can define $D\ (x/y + u/v) \equiv y \neq 0 \wedge v \neq 0$.

- Let us define $D(e)$ using the following algorithm.
  - If e contains no operations that can fail, then $D(e) \equiv T$
  - $D(b[e]) \equiv D(e) \wedge 0 \leq e < size(b)$.
  - $D(e_1/e_2) \equiv D(e_1 \,\%\, e_2) \equiv D(e_1) \wedge D(e_2) \wedge e_2 \neq 0$.
  - $D(sqrt(e)) \equiv D(e) \wedge e \geq 0$.
  - $D(op\ e) \equiv D(e)$.
  - $D(e_1\ op\ e_2) \equiv D(e_1) \wedge D(e_2)$ for op other than / and %.
  - $D\big(f(e_1, e_2, \dots)\big) \equiv D(e_1) \wedge D(e_2) \wedge \dots$ for $f()$ other than $sqrt()$.
  - $D\ (\mathbf{if}\ B\ \mathbf{then}\ e_1\ \mathbf{else}\ e_2\ \mathbf{fi}) \equiv D(B) \wedge \big(B \rightarrow D(e_1)\big) \wedge \big(\neg B \rightarrow D(e_2)\big)$.

3. Calculate domain predicate $D(e)$ for the following expressions.
   a. $D\big(b\big[b[k]\big]\big)$ $\equiv D(b[k]) \wedge 0 \leq b[k] < size(b)$
   $\equiv D(k) \wedge 0 \leq k < size(b) \wedge 0 \leq b[k] < size(b)$
   $\equiv T \wedge 0 \leq k < size(b) \wedge 0 \leq b[k] < size(b)$
   $\Leftrightarrow 0 \leq k < size(b) \wedge 0 \leq b[k] < size(b)$

   b. Let $B \equiv 0 \leq k < size(b)$.
   $D(\mathbf{if}\ B\ \mathbf{then}\ b[k]\ \mathbf{else} - 1\ \mathbf{fi})$
   $\equiv D(B) \wedge \big(B \rightarrow D(b[k])\big) \wedge \big(\neg B \rightarrow D(-1)\big)$
   $\Leftrightarrow T \wedge \big(B \rightarrow D(b[k])\big) \wedge (\neg B \rightarrow T)$
   $\Leftrightarrow B \rightarrow D(b[k])$
   $\equiv 0 \leq k < size(b) \rightarrow 0 \leq k < size(b)$
   $\Leftrightarrow T$

Avoid Runtime Error in Statements

- To avoid runtime errors in the execution of S, we can define domain predicate $D(S)$ that gives a sufficient condition that avoids runtime errors. We will discuss how to avoid divergence in the future.

- Let us define $D(S)$ using the following algorithm.
  - $D(\mathbf{skip}) \equiv T$
  - $D(v \coloneqq e) \equiv D(e)$
  - $D(b[e_1] \coloneqq e_2) \equiv D(b[e_1]) \wedge D(e_2)$
  - $D(S_1; S_2) \equiv D(S_1) \wedge wp\big(S_1, D(S_2)\big)$
    - $D(S_1)$ guarantees the execution of S_1 is error-free, $D(S_2)$ guarantees the execution of S_2 is error-free.
    - $wp\big(S_1, D(S_2)\big)$ guarantees that S_2 is executed in an acceptable state.

  - $D(\mathbf{if}\ B\ \mathbf{then}\ S_1\ \mathbf{else}\ S_2\ \mathbf{fi}) \equiv D(B) \wedge \big(B \rightarrow D(S_1)\big) \wedge \big(\neg B \rightarrow D(S_2)\big)$.
  - $D(\mathbf{if}\ B_1 \rightarrow S_1 \ \square\ B_2 \rightarrow S_2\ \mathbf{fi}) \equiv D(B_1 \vee B_2) \wedge (B_1 \vee B_2) \wedge \big(B_1 \rightarrow D(S_1)\big) \wedge \big(B_2 \rightarrow D(S_2)\big)$.

- $D(\textbf{while } B \textbf{ do } S_1 \textbf{ od}) \equiv D(B) \wedge \big(B \to D(S_1)\big)$.
- $D(\textbf{do } B_1 \to S_1 \,\square\, B_2 \to S_2 \textbf{ od}) \equiv D(B_1 \vee B_2) \wedge \big(B_1 \to D(S_1)\big) \wedge \big(B_2 \to D(S_2)\big)$.

<u>Calculate wp for Loop-Free Programs</u>

- $wp(S,q) \equiv D(S) \wedge wlp(S,q) \wedge D\big(wlp(S,q)\big)$

4. Calculate $w_1 \Leftrightarrow wp(x \coloneqq b[k],\ sqrt(x) \geq 1)$.

   - $D(x \coloneqq b[k]) \equiv 0 \leq k < size(b)$
   - $wlp(x \coloneqq b[k],\ sqrt(x) \geq 1) \equiv\ sqrt(b[k]) \geq 1$
   - $D\big(wlp(x \coloneqq b[k],\ sqrt(x) \geq 1)\big) \equiv D(sqrt(b[k]) \geq 1) \equiv b[k] \geq 0 \wedge 0 \leq k < size(b)$
   - $w_1 \equiv 0 \leq k < size(b) \wedge sqrt(b[k]) \geq 1 \wedge b[k] \geq 0 \wedge 0 \leq k < size(b)$
     $\Leftrightarrow sqrt(b[k]) \geq 1 \wedge b[k] \geq 0 \wedge 0 \leq k < size(b)$
     $\Leftrightarrow b[k] \geq 1 \wedge 0 \leq k < size(b)$

5. Calculate $w_0 \Leftrightarrow wp(x \coloneqq y; z \coloneqq v/x\,, z > x + 2)$.
   - Let $w \equiv wlp(x \coloneqq y; z \coloneqq v/x\,, z > x + 2)$.
     Then $w \equiv wlp(x \coloneqq y,\ v/x > x + 2) \equiv\ v/y > y + 2$.
   - $D(w) \equiv D(v/y > y + 2) \equiv y \neq 0$
   - $D(x \coloneqq y; z \coloneqq v/x)$ $\equiv D(x \coloneqq y) \wedge wp\big(x \coloneqq y, D(z \coloneqq v/x)\big)$
     $\equiv T \wedge wp(x \coloneqq y, x \neq 0)$
     $\equiv T \wedge D(x \coloneqq y) \wedge wlp(x \coloneqq y, x \neq 0) \wedge D\big(wlp(x \coloneqq y, x \neq 0)\big)$
     $\equiv T \wedge T \wedge y \neq 0 \wedge D(y \neq 0) \Leftrightarrow y \neq 0$
   - $w_0 \Leftrightarrow y \neq 0 \wedge v/y > y + 2$