A While language

A.1 Basic language

```
⟨program⟩ → ⟨statement⟩ (; ⟨program⟩)*

⟨statement⟩ → skip
       | ⟨variable⟩ := ⟨aexp⟩
       | if ⟨bexp⟩ then ⟨statement⟩ else ⟨statement⟩
       | while ⟨bexp⟩ do ⟨statement⟩
       | ( ⟨statement⟩ )

⟨bexp⟩ → true | false | ! ⟨bexp⟩ | ⟨bexp⟩ & ⟨bexp⟩ | ⟨bexp⟩ || ⟨bexp⟩
       | ⟨aexp⟩ = ⟨aexp⟩
       | ⟨aexp⟩ <= ⟨aexp⟩ | ⟨aexp⟩ < ⟨aexp⟩
       | ⟨aexp⟩ >= ⟨aexp⟩ | ⟨aexp⟩ > ⟨aexp⟩
       | ( ⟨bexp⟩ )

⟨aexp⟩ → ⟨integer⟩ | ⟨variable⟩
       | ⟨aexp⟩ + ⟨aexp⟩
       | ⟨aexp⟩ - ⟨aexp⟩
       | ⟨aexp⟩ * ⟨aexp⟩
       | ( ⟨aexp⟩ )
```

A.2 Annotations and comments

Comments begun with a `%` symbol and ended with a newline character `\n` may be inserted at any point within a While program. Pre/postconditions and intermediate assertions expressed in an extended language of boolean expressions may also be written within While programs between curly braces, in an extended language of boolean expressions, according to the following rules:

```
⟨statement⟩ → (⟨assertion⟩)? ⟨statement⟩ (⟨assertion⟩)?

⟨assertion⟩ → { ⟨bexpExtended⟩ }

⟨bexpExtended⟩ → ⟨bexp⟩
       | ⟨bexp⟩ -> ⟨bexp⟩
       | forall ⟨variable⟩ ⟨bexp⟩
       | exists ⟨variable⟩ ⟨bexp⟩
```

A.3 Language extensions

- Assigning array elements:

  ```
  ⟨statement⟩ → ⟨variable⟩ [ ⟨aexp⟩ ] := ⟨aexp⟩

  ⟨aexp⟩ → ⟨variable⟩ [ aexp ]
  ```

- Declaring and calling procedures:

  $$\begin{array}{l} \langle \textit{statement} \rangle \rightarrow \texttt{proc}\ \langle \textit{variable} \rangle\ \texttt{(}\ \langle \textit{variableList} \rangle\ \texttt{)}\ \texttt{(}\ \langle \textit{variableList} \rangle\ \texttt{)}\ \langle \textit{statement} \rangle \\ \qquad \mid\ \texttt{call}\ \langle \textit{variable} \rangle\ \texttt{(}\ \langle \textit{variableList} \rangle\ \texttt{)}\ \texttt{(}\ \langle \textit{aexpList} \rangle\ \texttt{)} \end{array}$$

  Here, $\langle \textit{variableList} \rangle$ and $\langle \textit{aexpList} \rangle$ are (possibly empty) comma-separated lists of variable names or expresssions respectively:

  $\langle \textit{variableList} \rangle \rightarrow \langle \textit{variable} \rangle?\ (\texttt{,}\ \langle \textit{variable} \rangle)^*$

  $\langle \textit{aexpList} \rangle \rightarrow \langle \textit{aexp} \rangle?\ (\texttt{,}\ \langle \textit{aexp} \rangle)^*$

- Declaring a block

  $\langle \textit{statement} \rangle \rightarrow \texttt{begin}\ \langle \textit{variableList} \rangle\ \langle \textit{statement} \rangle\ \texttt{end}$

B Hoare logic

B.1 title

Partial correctness

- Skip axiom $\dfrac{}{\{P\}\ \texttt{skip}\ \{P\}}$
- Assignment axiom $\dfrac{}{\{P[e/x]\}\ x\texttt{:=}e\ \{P\}}$
- Composition rule $\dfrac{\{P\}\ S_1\ \{R\}\ ,\ \{R\}\ S_2\ \{Q\}}{\{P\}\ S_1\texttt{;}S_2\ \{Q\}}$
- Conditional rule $\dfrac{\{b \wedge P\}\ S_1\ \{Q\}\ ,\ \{\neg b \wedge P\}\ S_2\ \{Q\}}{\{P\}\ \texttt{if}\ b\ \texttt{then}\ S_1\ \texttt{else}\ S_2\ \{Q\}}$
- Iteration rule $\dfrac{\{b \wedge I\}\ S\ \{I\}}{\{P\}\ \texttt{while}\ b\ \texttt{do}\ S\ \{\neg b \wedge I\}}$
- Consequence rule 1 $\dfrac{\{P\}\ S\ \{Q\}\ , P^+ \rightarrow P}{\{P^+\}\ S\ \{Q\}}$
- Consequence rule 2 $\dfrac{\{P\}\ S\ \{Q\}\ , Q \rightarrow Q^-}{\{P\}\ S\ \{Q^-\}}$

B.2 Extensions

- Parameterless Rule of Invocation $\dfrac{\{P\}\ S\ \{Q\}\ ,\texttt{proc}\ f\texttt{()()}S}{\{P\}\ \texttt{call}\ f\texttt{()()}\ \{Q\}}$
- Rule of Invocation $\dfrac{\{P\}\ S\ \{Q\}\ ,\texttt{proc}\ f\texttt{(}\boldsymbol{x}\texttt{)(}\boldsymbol{y}\texttt{)}\ S}{\{P\}\ \texttt{call}\ f\texttt{(}\boldsymbol{x}\texttt{)(}\boldsymbol{y}\texttt{)}\ \{Q\}}$

- Rule of Substitution $\dfrac{\{P\}\ \texttt{call}\ f\texttt{(}\boldsymbol{x}\texttt{)(}\boldsymbol{y}\texttt{)}\ \{Q\}}{\{P[\boldsymbol{a}/\boldsymbol{x},\boldsymbol{e}/\boldsymbol{y}]\}\ \texttt{call}\ f\texttt{(}\boldsymbol{a}\texttt{)(}\boldsymbol{e}\texttt{)}\ \{Q[\boldsymbol{a}/\boldsymbol{x},\boldsymbol{e}/\boldsymbol{y}]\}}$
- Rule of Declaration $\dfrac{\{P\}\ S[\boldsymbol{z}/\boldsymbol{x}]\ \{Q\}}{\{P\}\ \texttt{begin}\ \boldsymbol{x}\ S\ \texttt{end}\ \{Q\}}$

C Weakest preconditions

Skip $wp(\texttt{skip}, Q) = Q$

Assignment $wp(x\texttt{:=}e, Q) = Q[e/x]$

Composition $wp(S_1\texttt{;}S_2, Q) = wp(S_1, wp(S_2, Q))$

Conditional $wp(\texttt{if}\ b\ \texttt{then}\ S_1\ \texttt{else}\ S_2, Q) = (b \to wp(S_1, Q)) \wedge (\neg b \to wp(S_2, Q))$

Iteration $wlp(I, \texttt{while}\ b\ \texttt{do}\ S, Q) = I \wedge ((b \wedge I) \to wp(S, I)) \wedge ((\neg b \wedge I) \to Q)$