Towards a quantum programming language

**Operational Semantics**

**Simple Control Structures**

**Loops**

**Reasoning**

**Example**

### Programs

- Terminate or run forever
- Have classical output (even when not terminating)
- Have a post-execution state (when terminating)
- Realize any operation compatible with quantum mechanics

### Simple control operations:

- Measurement operator easy to define
- \( P;Q \): Sequential composition of programs
- \( \text{print} \): Classical output
- \( \text{if/switch} \): Case distinction (branching) based on measurement results

**Examples:**

- \( \text{print a; print b} \)
- \( \text{if (M) print 1} \)
- \( \text{print M} \) (shorthand)
- \( \text{switch (M as m)} \)

### Loops:

- Harder to formalise
- No natural lattice structure → Fixpoint approach fails
- No suitable topology → Limit approach fails
- Axiomatic approach works: State some required properties of loops and show that these define loops uniquely

**Examples:**

- \( \text{while (M)} \)
- \( \text{while (true)} \)

### Reasoning:

- Pre-/postconditions as sets of density operators
- Express equality of programs conditioned on initial states
- Reason about programs conditioned on some classical output

**Examples:**

- \( \{ 1 \} \ P \{ 1 \} \) if the initial state is a random state, so is the post-execution state (e.g. \( P \) is a permutation of basis states)
- \( \{ x \text{ in computational basis} \} \ P \text{-noop} \) if variable \( x \) is in the computational basis, program \( P \) has no effect (e.g. \( P \) might be a dephasing of \( x \))
- \( \{ \text{tr} = \frac{1}{2} \} \ P|a \) Program \( P \) has probability ½ of outputting \( a \)

### Programs modelled as:

- Measurement operators
- Mixture of POVM (for non-terminating) and generalized measurement (for terminating programs)
- Classical output is measurement outcome

**Examples:**

- \( \text{print a; print b} \)
- Equivalent programs outputting \( ab \)
- \( \text{if (M) print 1} \)
- If measurement \( M \) yields true, print 1
- \( \text{switch (M as m)} \) \( \text{print m} \)
- Outputs result of measuring \( M \)
- \( \text{print M} \) (shorthand)

**Examples:**

- \( \text{while (M)} \ P \)
- Run program \( P \) while measurement \( M \) yields true.
- \( \text{while (true)} \ P \)
- Measure \( M \) if nonzero, measure \( N \), output the outcome, and redo from start
- \( \text{while (true)} \ P \text{print x} \)
- Outputs infinite sequence \( x^{\infty} \).

---

### Example

\[
\begin{align*}
H & \quad \text{i:=0; while (i<n) \{ H; x[i]; i:=i+1 \}} \\
U_x & \quad \text{y:=0; y \leftarrow y \oplus f(x); \ \sigma_y; y \leftarrow y \oplus f(x)} \\
U_0 & \quad \text{y:=0; y \leftarrow y \oplus OR(x); \ \sigma_y; y \leftarrow y \oplus OR(x)} \\
\text{Grover} & \quad \text{i:=0; while (i<n) \{ x[i]:=0; i:=i+1 \}; H; H; H; H; k := k+1 \}})\}; \\
& \quad \text{i:=0; while (i<n) \{ print x[i]; i:=i+1 \}}
\end{align*}
\]