NOVEL LOGICAL REASONING TUTOR
Matthew F. Pfister
the tutor’s
PURPOSE

to evaluate the correctness of conjectures students make about teacher generated code

... independently of the teacher’s aid
defining

“CORRECTNESS”

verifiably true  \rightarrow  verifying compiler

and

nontrivial  \rightarrow  ?

to exclusively evaluate all verifiably true and nontrivial answers as correct
defining
“CORRECTNESS”

Confirm ___ = m; | m is some expression

Set of verifiably true expressions \( E \) ...

Single, teacher-defined key expression \( k \mid k \in E \) ...

Set of variant expressions \( V \subseteq E \mid \forall v \in V, v = k \) is true in the context of the code

Subset of trivial expressions \( T \subseteq E \mid \forall t \in T, t = m \) is true outside of the context of the code

The set of correct answers \( C \) is thus defined as:

\[ C \mid \forall c \in C, c \in E \text{ and } c \in V \text{ and } c \notin T \]
the two types of CONFIRMATIONS

Confirm __________;
Evaluates the conditional given in the blank in the context of the code.

Symbolic Confirm __________;
Evaluates the conditional given in the blank outside of the context of the code. (Except for relevant variable declarations).
Var X, Y, Z : Integer;
/* An elegant solution to the TSP in linear time that is too small to include on a single slide or else I would */

Symbolic Confirm X = Z;
proof rule for

SYMBOLIC CONFIRM

Var X, Y, Z : Integer;
/* An elegant solution to the TSP in linear time that is
too small to include on a single slide or else I would */

Var X`, Z` : Integer;
Read(X`);
Read(Z`);
Confirm X` = Z`;
automated grading of EXPRESSIONS

Consider this example, where $X$, $Y$, and $Z$ are all verifiably correct answers but 0 is not.

Guiding Question: Which variable is equivalent to $Z$?

-- Teacher-defined key expression $k = X$
Var X, Y, Z: Integer;
Read(Z);
X := Z;
Y := Z;
Confirm 0 = Z;

-- Teacher-defined key expression \( k = X \)

UNVERIFIABLE CASE:
The student provides the unverifiable expression 0.
Var X, Y, Z : Integer;
Read(Z);
X := Z;
Y := Z;
Confirm 0 = Z;

Confirm 0 = X;

Symbolic Confirm not(0 = Z);

-- Teacher-defined key expression k = X
Var X, Y, Z : Integer;
Read(Z);
X := Z;
Y := Z;
Confirm 0 = Z;

```
^UNVERIFIED^n
Confirm 0 = X;
```

```
^UNVERIFIED^n
Symbolic Confirm not(0 = Z);
```

```
^VERIFIED^n
-- Teacher-defined key expression k = X
```

**UNVERIFIABLE CASE:**
The student provides the unverifiable expression 0.
Var X, Y, Z : Integer;
Read(Z);
X := Z;
Y := Z;
Confirm X = Z;

EXPECTED CASE:
The student provides the expected expression X.

-- Teacher-defined key expression k = X
Var X, Y, Z : Integer;
Read(Z);
X := Z;
Y := Z;
Confirm X = Z;

Confirm X = X;

Symbolic Confirm not(X = Z);

-- Teacher-defined key expression k = X
Var X, Y, Z : Integer;
Read(Z);
X := Z;
Y := Z;
Confirm X = Z;

^^VERIFIED^^
Confirm X = X;

^^VERIFIED^^
Symbolic Confirm not(X = Z);

^^VERIFIED^^

-- Teacher-defined key expression k = X

EXPECTED CASE:
The student provides the expected expression X.
Var X, Y, Z : Integer;
Read(Z);
X := Z;
Y := Z;
Confirm Y = Z;

-- Teacher-defined key expression k = X

VARIANT CASE: The student provides the variant expression Y.
Var X, Y, Z : Integer;
Read(Z);
X := Z;
Y := Z;
Confirm Y = Z;

Confirm Y = X;

Symbolic Confirm not(Y = Z);

-- Teacher-defined key expression k = X
automated grading of

EXPRESSIONS

Var X, Y, Z : Integer;
Read(Z);
X := Z;
Y := Z;
Confirm Y = Z;
^^VERIFIED^^
Confirm Y = X;
^^VERIFIED^^
Symbolic Confirm not(Y = Z);
^^VERIFIED^^

-- Teacher-defined key expression k = X

VARIANT CASE:
The student provides the variant expression Y.
Var X, Y, Z : Integer;
Read(Z);
X := Z;
Y := Z + 1;
Confirm Y - 1 = Z;

--- Teacher-defined key expression k = X

WHITE RABBIT CASE:
The student provides the verifiable but not variant expression Y - 1.

Guiding Question:
Which variable is equivalent to Z?
Var X, Y, Z : Integer;
Read(Z);
X := Z;
Y := Z + 1;
Confirm Y - 1 = Z;

Confirm Y - 1 = X;

Symbolic Confirm not(Y - 1 = Z);

-- Teacher-defined key expression k = X

automated grading of EXPRESSIONS

WHITE RABBIT CASE:
The student provides the verifiable but not variant expression Y - 1.
Var X, Y, Z : Integer;
Read(Z);
X := Z;
Y := Z + 1;
Confirm Y - 1 = Z;  -- verifiable
^^VERIFIED^^
Confirm Y - 1 = X;  -- not variant
^^UNVERIFIED^^
Symbolic Confirm not(Y - 1 = Z);
^^VERIFIED^^

-- Teacher-defined key expression k = X

**WHITE RABBIT CASE:**
The student provides the verifiable but not variant expression Y - 1.
Var X, Y, Z : Integer;
Read(Z);
X := Z;
Y := Z;
Confirm Z = Z;

TRIVIAL CASE:
The student provides the trivial expression Z.

-- Teacher-defined key expression k = X
Var X, Y, Z : Integer;
Read(Z);
X := Z;
Y := Z;
Confirm Z = Z;

Confirm Z = X;

Symbolic Confirm not(Z = Z);

-- Teacher-defined key expression k = X

TRIVIAL CASE:
The student provides the trivial expression Z.
Var X, Y, Z : Integer;
Read(Z);
X := Z;
Y := Z;
Confirm Z = Z;  
^^VERIFIED^^
Confirm Z = X;  
^^VERIFIED^^
Symbolic Confirm not(Z = Z);  
^^UNVERIFIED^^

-- Teacher-defined key expression k = X
summary of USE CASES

<table>
<thead>
<tr>
<th></th>
<th>Verifiable Check</th>
<th>Variant Check</th>
<th>Non-Trivial Check</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNVERIFIABLE CASE</td>
<td>Red</td>
<td>Red</td>
<td>Green</td>
</tr>
<tr>
<td>EXPECTED CASE</td>
<td>Green</td>
<td>Green</td>
<td>Green</td>
</tr>
<tr>
<td>VARIANT CASE</td>
<td>Green</td>
<td>Green</td>
<td>Green</td>
</tr>
<tr>
<td>WHITE RABBIT CASE</td>
<td>Green</td>
<td>Red</td>
<td>Green</td>
</tr>
<tr>
<td>TRIVIAL CASE</td>
<td>Green</td>
<td>Green</td>
<td>Red</td>
</tr>
</tbody>
</table>

with an additional symbolic check of the student’s answer to the key, we can tell *how variant* an answer really is...
When symbolically comparing correct expressions to keys...

**Code Independent Variance (evaluate to true):**
Symbolic Confirm $Y = X$; -- innocent
Symbolic Confirm $X + 1 - 1 + 1 - 1 = X$; -- not innocent

**Code Dependent Variance (evaluates to false):**
$X := Y$
Symbolic Confirm $X = Y$;
3 broad categories of problems

PROBLEMS

Expressions: which we have seen in depth.
  Confirm ___ = m;

Operators: which are so trivial, string matching is appropriate.
  Confirm m _ n;

Conditionals: the holy grail of evaluating understanding.
  Confirm _______;

defining
“CORRECTNESS”

Confirm _____;

Set of verifiably true conditionals $E$ ...

Single, teacher-defined key conditional $k \mid k \in E$ ...

Subset of variant conditionals $V \subseteq E \mid \forall v \in V, v \rightarrow k$ is true in the context of the code

Subset of trivial conditionals $T \subseteq E \mid \forall t \in T, t$ is true outside of the context of the code

The set of correct answers $C$ is thus defined as:

$$C \mid \forall c \in C, c \in E \text{ and } c \in V \text{ and } c \notin T$$