

Debugging an Operation

Using a Failed Proof of Correctness as a Guide

Joan Krone – Denison University

Joseph Hollingsworth – Indiana University Southeast

What Went Wrong?

- Student implements an operation
- Student creates a testing harness
a small client program
- Student runs the testing harness
- Receives unexpected output
- The operation doesn't seem to do anything!

Invert

```
Operation Invert (Q: P_Queue);  
           ensures Q = Reverse(#Q);
```

The outgoing Q equals the reverse of the incoming Q

- #Q – incoming
- Q – outgoing

P_Queue is a preemptable queue

- Has one additional operation – Inject

String Theory

Used for mathematically modeling a Queue

➤ Γ - is the alphabet, e.g., $\Gamma = \mathbf{Z}$

String Theory

Used for mathematically modeling a Queue

➤ Γ - is the alphabet, e.g., $\Gamma = \mathbf{Z}$

➤ Example strings:

- $\alpha = \langle 4, 22, 3, 17 \rangle$ *string of 4 integers*
- $\beta = \langle \rangle$ *empty string*

String Theory

Used for mathematically modeling a Queue

➤ Γ - is the alphabet, e.g., $\Gamma = \mathbf{Z}$

➤ Example strings:

- $\alpha = \langle 4, 22, 3, 17 \rangle$ *string of 4 integers*
- $\beta = \langle \rangle$ *empty string*

➤ Concatenation, Length, Reverse

- $\langle -5 \rangle \circ \alpha \circ \langle 101 \rangle = \langle -5, 4, 22, 3, 17, 101 \rangle$
- $|\alpha| = 4$
- $\text{Reverse}(\alpha) = \langle 17, 3, 22, 4 \rangle$
- $\text{Reverse}(\alpha \circ \langle 101 \rangle) = \langle 101, 17, 3, 22, 4 \rangle$

Testing Harness for Invert

Begin

```
q: P_Queue;  
Enqueue(3, q);  
Enqueue(2, q);  
Enqueue(1, q);  
Writeln("q before Invert = ", q);  
Invert(q);  
Writeln("q after Invert = ", q);
```

end;

Output:

q before Invert = <3,2,1>

q after Invert = <3,2,1>

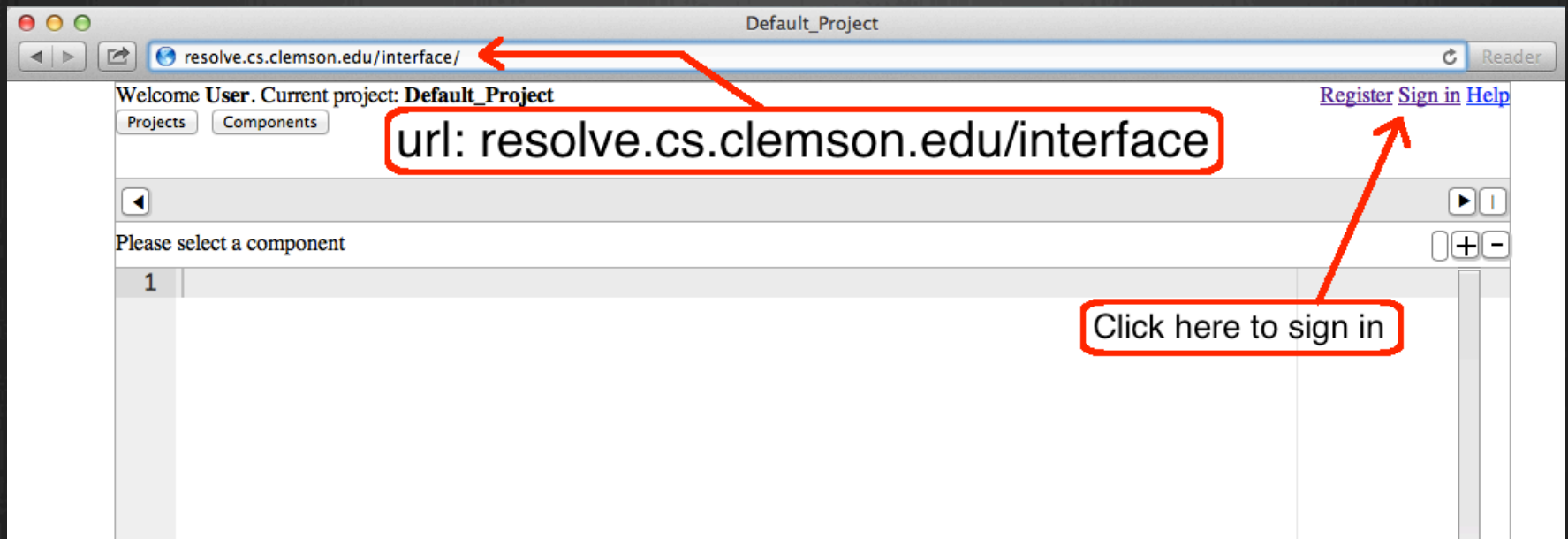
How to Debug? I Could Try ...

- Using one of those genetic algorithms to make random mutations to my code ...

Or Maybe I Should ...

- Do that proof of correctness thingy that my professor showed us.

Accessing the RESOLVE Web Interface



Accessing Components

Welcome **Joseph**. Current project: **Default_Project** [Log out](#) [Help](#)

Projects Components Import Export

Preemptable_Queue Inverting_Capability Recursive_Inverting Defective_Recursive

VCs Verify Save Rename Delete Executable + -

1 **Realization** Defective_Recursive_Inverting_Realiz
2 **for** Inverting_Capability **of** Preemptable_Queue_Template;
3
4 **Recursive Procedure** Invert(**updates** Q: P_Queue);
5 **decreasing** |Q|;
6
7 **Var** E: Entry;
8 **If** (Length(Q) \neq 0) **then**
9 **Dequeue**(E, Q);
10 **Invert**(Q);
11 **Inject**(E, Q);
12 **end**;
13 **end** Invert;
14 **end** Defective_Recursive_Inverting_Realiz;

Click "Components" button to access component library and to add your own.

Components open up in their own tab.