Program Debugging (Special Topic)

TOPICS

- Engineering Methodology
- Requirements, Algorithm, Example, Coding
- Testing and Debugging
- Testing methodologies

More Detail

- Figure out what problem you are solving.
- Read the specification very carefully.
- Figure out how to solve the problem, on paper.
- Make an example of the centerline case.
- Add corner cases and error handling later.
- Incremental development is more effective.
- Test each piece by itself.

Think before coding!

Testing and Debugging

- Test individual methods before using them.
- Not just centerline, also boundary conditions.
- Debug your code in a methodical fashion.
- Thoroughly investigate any anomalies in behavior.
- Make sure to test all the branches in your code.
- Anything can be debugged, given enough time.

If you haven’t tested it, it doesn’t work!
Print Debugging

- Add a print statement after every operation.
- Check the values against your example.
- Run several times with different input data.
- Remove or comment out print statements.
- Verify against test suite, if available.
- Check against the assignment specification.

Tedious, but effective, and guaranteed to improve assignment scores.

Debugging Tools

- Eclipse makes debugging easy.
- Run the program in debug mode (demo).
- Learn how to set breakpoints.
- Single stepping: ‘step into’ versus ‘step over’.
- Can examine current values of all variables.
- More advanced features available.

This is how professional programmers debug their code.

Eclipse Debugging (1)

Starting a debugging session.

Eclipse Debugging (2)

Setting breakpoints in code.
Eclipse Debugging (3)

Examining the values of variables.

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>3.0</td>
</tr>
<tr>
<td>B</td>
<td>4.0</td>
</tr>
<tr>
<td>C</td>
<td>5.0</td>
</tr>
</tbody>
</table>

Testing Methodologies

- Unit Testing: verify that each method works.
- Module Testing: verify that each class works.
- Integration: verify that classes interact correctly.
- System Testing: verify that system works as desired.
- Black Box versus White Box testing
- Centerline testing versus Corner Cases
- Checking that Error Conditions are handled
- Test Driven Development: what is it?