Announcements

Project 1a is due Friday … see turn-in instructions

Midterm 1 is next Monday

Tip of the day: The most useful habit for successful computing is that of being perfectly accurate

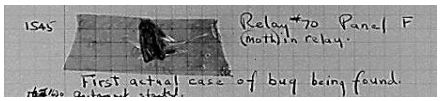
Debugging & Troubleshooting

"To err is human, but it takes a computer to really foul things up"

Debugging

Debugging is the process of finding the error in a faulty (IT) system

- The term was coined by Grace Hopper



Debugging is not algorithmic -- there are no guaranteed-to-work rules to find errors

When You Debug...

There are guidelines for debugging…

Rather than trying things aimlessly and becoming frustrated, think of yourself as solving a mystery Miss Marple or H. Poirot

- Be objective: What are my clues? What is my hypothesis? Do I need more data?
- Consciously 'watch' yourself debug -- its an out-of-body experience
- When stumped, don't become frustrated, but ask, "What am I misunderstanding?"

Debugging Guidelines

Memorize?

- Verify that the error is reproducible
- Determine exactly what the problem is
- Eliminate the "obvious" causes
- Partition the process, separating out the parts that work from the part that doesn't work
- When you reach a dead end, reassess the information you have, trying to identify the mistake you are making

Reproducibility

First step: verify the error is reproducible

- Transient errors are very rare, but they do happen … try again

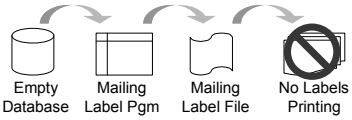
Getting Out and Getting Back In

- Rebooting the operating system is advisable, especially for errors involving peripheral devices (printers, modems)

Determine the Problem

Second step: figure out what's wrong

* Often there is a sequence of steps following an error and propagating it … work backwards looking to see where the error first occurred



Eliminate the Obvious

Third step: eliminate obvious causes

"If the cause were obvious, the problem would have been fixed!"

* There are standard things to check:
  - Inputs
  - Connections
  - "Permissions"
  - Physical connectivity

"Working" in similar situations is usually good enough

Isolate the Problem

Try to "partition" the situation into working and non-working parts

- Form a hypothesis of what's wrong
- Make as few assumptions as possible
- Take nothing for granted

The goal is to eliminate as many things from consideration as possible

On Reaching a Dead End

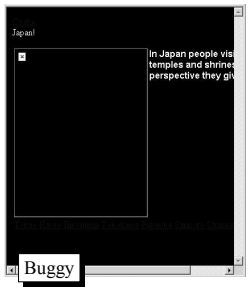
When everything seems to check out, don't get frustrated ... ask, "What am I misunderstanding?"

* Your goal is to see the situation as it is, not as you think it should be
* Step through the process predicting what will happen, looking for places where your prediction is wrong

A Debugging Example

```
<html><head><title>Balance</title></head>
 <body>
  <img align="right" width="234" height="296" src="tang.jpg">
  <img align="right" width="117" height="148" src="tang.jpg">
  <img align="right" width="100" height="110" src="tang.jpg">
  <img align="left"  width="234" height="296" src="tang.jpg">
  <img align="left"  width="117" height="148" src="tang.jpg">
  <img align="left"  width="100" height="110" src="tang.jpg">
</body>
</html>
```

A Buggy Web Page



```
<html><head><title>Japan Page</title></head>
<body bgcolor="black"><font color="white">
  <table>
    <tr>
      <td> <a href="tokyo.html">  Tokyo</a></td>
      <td> <a href="kyoto.html">  Kyoto</a></td>
      <td> <a href="osaka.html">  Osaka</a></td>
      <td> <a href="hiro.html">   Hiroshima</a></td>
      <td> <a href="yoko.html">   Yokohama</a></td>
      <td> <a href="fukuoka.html">Fukuoka</a></td>
      <td> <a href="sapp.html">   Sapporo</a></td>
      <td> <a href="oki.html">    Okinawa</a></td>
      <td> <a href="oni.html">    Onimichi</a></td>
    </tr>
  </ table> <hl>Japan! </hl>
  <h2><font face="Helvetica"><At Heian Shrine</font></h2>
  <img align="left" src="tang.jpeg" width="234" height="296">
  <p> <b>In Japan people visit the temples and shrines
     for the perspective they give. </b></p>
 </body>
</html>
```

Illustration

Summary

Debugging is not algorithmic, but there are guidelines to follow

- It probably pays to memorize them so they come to mind while debugging
- Watch yourself debug -- assess how you are doing, what you need to know
- Being accurate -- avoiding textual mistakes at all -- saves frustration

Notice 6 wrong letters messed up the whole page