CS 261 Lab #2

In which we delve into pointers
“Pointers” are just memory addresses. They “point” to some location in the computer’s memory.
foo,
*bar;

&foo is 1000
&bar is 1004

foo = 10;
bar = &foo;

bar == 1000
*bar == 10

*bar = 20;

<table>
<thead>
<tr>
<th>Memory address</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>“foo”</td>
<td>1000</td>
</tr>
<tr>
<td>1001</td>
<td>?</td>
</tr>
<tr>
<td>1002</td>
<td>?</td>
</tr>
<tr>
<td>1003</td>
<td>?</td>
</tr>
<tr>
<td>“bar”</td>
<td>1004</td>
</tr>
<tr>
<td>1000</td>
<td>1000</td>
</tr>
</tbody>
</table>
Breakpoints tell the computer to stop execution at specific points. You can see variables’ values while your program is still running.

Execute one line at a time with the step into and step over commands.

Resume normal execution of your program with the continue command.

“gutter”  
breakpoint
Debugging strategy

Identify a line just before the problem area. Set a breakpoint here.

Use the debugger to verify your assumptions—problems often start earlier than we think.

Step through the problematic areas, examining variables until you find the problem.
Debugging in Visual Studio

Insert breakpoints by **clicking in the gutter**.

Start debugging by pressing **F5**.

After a breakpoint is hit, step through the program with **F10** or **F11**. **F11 steps into** functions; **F10 steps over** them.

Hover over a variable to see its value (but only **after** the line has has executed).
Three exercises

Download from
http://classes.engr.oregonstate.edu/eecs/winter2015/cs261-001/lab2.php

factorial.c — We’ll go through the steps to debug this program together.

pointers.c — Create pointers, point them at existing variables, and print their contents.

swap.c — Debug a program that uses pointers to swap variable values.

When you’re finished, you can start working on Assignment #1 with your partner.