Debugging

Types of bugs

Syntax Errors

* These are easy to find, compiler or interpreter finds them
* Things such as forgotten parenthesis

Runtime Errors

* Found when the code is running
* Things such as divide by 0

Semantic Errors, or errors in meaning

* What code is intended to do it does not do
* Such as logic errors, using a > when you needed a >=
  + Makes the code execute the wrong if-else branch for some cases

So how do I find the source of a bug?

* Reread the requirements and see if we are doing what they say
  + Large source of error we misread
* Look for a prior report of an error
  + If no such one exist, make one
* Examine the code for possible logic errors
* Check your tests for faults
  + This can drive you crazy!
* Add print statements to aid in tracing
* Run a debugger to investigate the problem
* Talk to management, they may be able to help

Steps to debug a program

* Run the program to get near the problem point
  + In case of a crash we run it until it crashes
* Locate one or two places in the code before the problem
  + Place a breakpoint
* Run or rerun the program so it stops at the breakpoint
* Examine the variables to see if you can see what’s wrong
* Step through code line-by-line to follow the execution path, called ‘stepping’
* Step through code function-call-by-function-call, called ‘nexting’
* Identify code changes that could help you fix it

Debuggers

* Tools that allow us to stop a program midstream and:
  + Check values
  + Follow code paths
  + Watch how values change
* There are high level and low level debuggers
* We will look at two in this lecture
  + PyCharm Debugger
  + Pbd

With pdb

The main ways to use it:

Postmortem tool (cannot be done in PyCharm)

When program crashes and terminates, we can run it again with a special pdb flag

Python 3 –m pdb myprogram.py

We will get a stacktrace and pdb prompt

WE can use this to examine the program state when it crashes

An interactive tool step/next style

As instrumentation to debug and trace starting at a particular code location

Pdb.settrace() stop program at that point

Display

List

Testing

* Bug – error in code
* A fault is a result of a defect during a code execution  
  A fault products a failure in operation

“System under test”

* The element which the behavior the tests will validate

Stimulus

* Causes the system under test to react in some way
  + Example:   
    Program inputs
  + Function arguments
  + Causes a system response, or reaction
* Examples:
  + Return values of function
  + Printed outputs

Facts of testing

* Start small first
* Plan testing early in development
  + Typically before any code is written
  + Developing testing process along with code design
  + Great testing depends on good requirements and design
  + 80% of the problems occur in 20% of the code
  + Testing is most effective when the code author does not do it
  + Complete, exhaustive testing is impossible
  + “Resisting testing is futile; you and your code will be test”

Eval(str) – evaluates string as code

Def foo(x,y):

If x==0:

Return y

Else:

Foo(x-1, y+3)

Execution diagram

Foo(x=2, y=0):

Foo(x=1, y=3)



Foo(x=0,y=6)



Return 6

Return 6

Return 6

[8:1:-3]

Start at char 8

Go to char 1

By going back by 3