Lecture 11

##by prof guttag

Testing and Debugging.

Definitions:

* validation : process designed to uncover problems and increase confidence(that our program does what it is designed to do). we can increase our confidence but never be 100% confident that it will do what it is designed to do.
  + testing
  + reasoning – have we tested it on enough inputs?
* debugging : process if ascertaining why the program is not working?? it has two aspects:
  + functioning
  + performance debugging
* defensive programming : writing programs such that they facilitate both validation and debugging

Testing and debugging are not the same:

* in testing we see the input and output pair
* in debugging we study the events that led to an error

Types of testing:

* unit testing: validating each module separately
* integration testing: we put our program together and see if the whole thing works.

why is testing hard? :

exhaustive testing is never feasible for an interesting program.

we need a test suite:

* it is small enough so we can test it in a small amount of time
* but large enough so as to give us some confidence

Myths about bugs:

1. bugs crawl into programs. if there is a bug then it is there because you put it there

And if you find a bug then it probably means that there are more of them.

Goal of debugging is: NOT TO ELIMINATE ONE BUG. BUT TO MOVE TOWARDS A BUG-FREE PROGRAM.

The two things that help is in debugging:

1. print statement
2. reading the code

and be systematic. if we do this then we will reduce search space thus localizing the source of the bug.

HOW CAN WE BE SYSTEMATIC?

1. Study the program text and ask yourself how could it have produced this result?
2. Is it part of a family? is this a mistake that I have made all through the program? if yes then fix it all through the program before moving on to other bugs.
3. How to fix them?

Scientific method of debugging

1. study available data
   1. test results(all of them, right and wrong)
   2. code
2. Form a hypothesis consistent with all the data
3. Design and run a repeatable experiment.

**REPEATABLE EXPERIMENT:**

it should have the potential to refute the hypothesis. it should have some chance to show us that the hypothesis is wrong.

It can have intermediate results.

**We should know what the expected result should be.**

if the experiments are random then we wanna figure out what is a repeatable experiment.

**HOW TO DESIGN THE EXPERIMENT ITSELF?**

Find the simplest input that may cause the bug to arise.

And find out the part of code that is most probably causing the bug. and preferably use binary search. so as to eliminate half the code when we do one check.

* to create a **copy of the list** res and store it into tmp we can do: tmp = res[:]