4/15 6

11/1/16

more testing - discuss assignment

so far, we've mostly been concerned with how to select test *inputs*

equivalence partitions
boundary cases
forcing a statement or branch or error handler to achieve coverage

all concerned with *inputs*

so how can we tell if the program has a bug?

if it crashes or hangs, or otherwise produces no output when output was expected, then obviously there is a bug

but if it does produce output, how do we know if the output is correct?

the code that checks whether the output is correct is called the "test oracle"

4/156 11/1/16

the test oracle might be implemented as an assertion or as regular code

it has to somehow "know" what the correct output ought to be

somes the developer "know" exactly what the result should be, e.g., from the user stories or use cases, and can encode this

sometimes it is necessary to compute the right output value or check the value produced in some way, but keep in mind this code can it self contain bugs

in some application domains, e.g., scientific computing, machine learning, data mining, simulation, optimization, search, it may not be known in advance what the correct output should be or it might be too expensive or impractical to compute on the fly (e.g. NP-complete problems)

We call these "non-testable programs" there are various approaches to testing that do not rely on an oracle (not addressed in this course)

4156 11/1/16

sometimes there is an oracle, in principle, but the tester does not know what it is.
e.g. the requirements are incomplete or not written at all

then tester might conduct "exploratory testing" to figure out what the SW is supposed to do

hard to tell if there's a bug unless SW crashes or hangs

but when the tester can discern a feature, use that feature to drive further testing

distinguish in testing between test-to-pass & test-to-fail

initial testing (e.g., before a demo) is test-to-pass, assure that SW minimally works

most testing is test-to-fail, trying to find bugs in development lab before deployment

4156 11/1/16

that was all concerned with "dynamic testing", or just plain "testing"

also "static testing"

We discussed static testing of requirements earlier, could also be applied to design

now we'll consider static testing of the code

two kinds— static analysis, already discussed, uses a tool to find "code smells", resource leaks, simple bugs or code that is suspicious in some way, w/o actually knowing what SW is supposed to do

now consider manual static testing, aka code review or code inspection

informal code review happens on the fly during pair programming

4156 11/2/16

many organizations also conduct more formal code reviews

why read/review code, why not just run (dynamic) tests?

- find problems earlier
- find problems that dynamic testing can't such as unreadable code
- helps cross-train developers on code written by other people (truck factor)

essential elements of formal code review

- identify problems in code, not author
- follow "rules"

e.g. predefined roles
moderator — often external
recorder

length of time ← 2 hours → standard
or
length of code

- prepare - everyone reads in advance

4156 11/1/16

- wrk report afterwards

summarize problems found
how many
where
what kind
in addition to bug reports
or change requests

report can help later to identify
particular problem areas of code
or common problems across code

basic idea of meeting is walkthrough
code, reading line by line

explain / discuss what code does & why
might be led by author or
another developer

besides moderator, recorder & author(s),
attendees might include
representatives of customer
end-users, testers, customer / technical
QA team support

4156 11/1/16

goal is to find problems,
not to fix in real time

code should already compile
cleanly & pass static analysis
(checkstyle, find bugs)

problems classified as minor, moderate,
severe

minor – author fixes on own
moderate – moderator checks
severe – need another review meeting

might also review test cases
corresponding to the code unit

some other things to test for web/mobile

- will your app work for all current,
past & future browser versions
- ~~[illegible]~~ will your app work for mobile
devices with arbitrary screen sizes
- what if the user turns on "accessibility"
options such as screen reading?

4156 11/1/16

besides the code review report, where do bug reports & change requests go?
where do bug reports from conventional testing go?

email? unfortunately, this may be the case, but then not available to other developers & usually disorganized

task board? perhaps while being fixed, as a task, but once completed it disappears when task board cleared for next iteration

need an "issue tracker"

typical model

create → open
in progress
resolved
closed
possibly reopened
} workflow

assigned to someone
reported (opened) by someone
has some priority

4156 11/1/16

can record time spent
can link to other issues & other materials

answers questions
What issues are open / in progress
who is working on them
which are most important

let's look at some github issues
- ~~phosphor~~ randoop
phosphor?

also bugzilla, jira, etc.