

Black Box Software Testing

(Academic Course -Fall 2001)

Cem Kaner, J.D., Ph.D.

Florida Institute of Technology

Section: 1 :

Opening Exercise: triangle

Contact Information:

kaner@kaner.com

www.kaner.com (testing practitioners)

www.badsoftware.com (software law)

www.testingeducation.org (education research)

Copyright (c) Cem Kaner 2001.

I grant permission to make digital or hard copies of this work for personal or classroom use, without fee, provided that (a) copies are not made or distributed for profit or commercial advantage, (b) copies bear this notice and full citation on the first page, and if you distribute the work in portions, the notice and citation must appear on the first page of each portion. Abstracting with credit is permitted. The proper citation for this work is Cem Kaner, *A Course in Black Box Software Testing (Academic Version)*, Fall 2001, www.testing-education.org.

To copy otherwise, to republish or post on servers, or to distribute to lists requires prior specific permission and a fee. Request permission to republish from kaner@kaner.com.

Copyright (c) Cem Kaner 2001.

About Cem Kaner

➤ I'm in the business of improving software customer satisfaction

I've worked as a programmer, tester, writer, teacher, user interface designer, software salesperson, organization development consultant, as a manager of user documentation, software testing, and software development, and as an attorney focusing on the law of software quality. These have provided many insights into relationships between computers, software, developers, and customers.

➤ Current employment

- Professor of Computer Sciences, Florida Institute of Technology
- Private practice in the Law Office of Cem Kaner

Books

- *Lessons Learned in Software Testing: A Context-Driven Approach* (with James Bach & Bret Pettichord, 2001).
- *Testing Computer Software* (1988; 2nd edition with Hung Nguyen and Jack Falk, 1993). This received the *Award of Excellence* in the Society for Technical Communication's *Northern California Technical Publications Competition* and has the lifetime best sales of any book in the field.
- *Bad Software: What To Do When Software Fails* (with David Pels). Ralph Nader called this book “a how-to book for consumer protection in the Information Age.”

Education

- J.D. (law degree, 1993). Elected to the American Law Institute, 1999.
- Ph.D. (experimental psychology, 1984) (trained in *measurement theory* and in *human factors*, the field concerned with making hardware and software easier and safer for humans to use).
- B.A. (primarily mathematics and philosophy, 1974).
- Certified in Quality Engineering (American Society for Quality, 1992). Examiner (1994, 1995) for the California Quality Awards.
- I also co-founded and/or co-host the *Los Altos Workshops on Software Testing*, the *Software Test Managers' Roundtable*, the *Austin Workshop on Test Automation*, the *Workshop on Model-Based Testing*, and the *Workshop on Heuristic & Exploratory Techniques*.

Notice

- **These course notes are copyrighted.**
- **For uses outside those listed in the license on the first page, please request permission from me at kaner@kaner.com.**
- **These notes were originally developed in co-authorship with Hung Quoc Nguyen. James Bach has contributed substantial material. I also thank Jack Falk, Elizabeth Hendrickson, Doug Hoffman, Bob Johnson, Brian Lawrence, Melora Svoboda, and the participants in the Los Altos Workshops on Software Testing and the Software Test Managers' Roundtables. Additional acknowledgements appear below.**
- **These notes include legal information, but you are not my legal client. I do not provide legal advice in the course. I may use a question of yours as a teaching tool and answer in a way that would “normally” be true but I cannot explore enough details in a classroom to respond with a competent legal opinion. My answer could be completely inappropriate for your particular situation. I cannot accept responsibility for actions that you might take in response to my comments in this course. If you need legal advice, please consult your own attorney.**

Contents

1	Opening exercise: triangle	16	Paradigm: stress
2	An example test series	17	Paradigm: exploratory
3	Boundaries & equivalence	18	Questioning & exploring
4	Reusable test matrices	19	Exploratory testing in pairs
5	Completeness is impossible	20	Paradigm: user
6	Overview of development	21	Paradigm: scenario
7	Black box testing group	22	Paradigm: stochastic
8	Bug advocacy	23	Combination testing
9	Intro to black box design	24	Managing GUI automation
10	Techniques and paradigms	25	Automation architectures
11	Paradigm: function testing	26	Intro to test documentation
12	Paradigm: regression	27	Scripting test cases
13	Paradigm: domain testing	28	Docs requirements analysis
14	Paradigm: spec-driven	29	Metrics & measurement
15	Paradigm: risk-based	30	Status reporting

Testing Computer Software

Part 1.

An Introductory Exercise

Introductory Exercise

- **The program reads three integer values from a card. The three values are interpreted as representing the lengths of the sides of a triangle. The program prints a message that states whether the triangle is scalene, isosceles, or equilateral.**
 - From Glen Myers, *The Art of Software Testing*
- *Write a set of test cases that would adequately test this program.*
- *Please write your name on your answer so that I can return it to you. Hand it in when you are done.*

List 10 tests that you'd run that aren't in Myers' list.

- 1.
- 2.
- 3.
- 4.
- 5.
- 6.
- 7.
- 8.
- 9.
- 10.

(Do or finish this AFTER we complete Part 2)