

SOFTWARE ERROR DETECTION

through Testing and Analysis

J.C. HUANG

CONTENTS

Preface	ix
1 Concepts, Notation, and Principles	1
1.1 Concepts, Terminology, and Notation	4
1.2 Two Principles of Test-Case Selection	8
1.3 Classification of Faults	10
1.4 Classification of Test-Case Selection Methods	11
1.5 The Cost of Program Testing	12
2 Code-Based Test-Case Selection Methods	14
2.1 Path Testing	16
2.2 Statement Testing	17
2.3 Branch Testing	21
2.4 Howden's and McCabe's Methods	23
2.5 Data-Flow Testing	26
2.6 Domain-Strategy Testing	36
2.7 Program Mutation and Fault Seeding	39
2.8 Discussion	46
Exercises	51
3 Specification-Based Test-Case Selection Methods	53
3.1 Subfunction Testing	55
3.2 Predicate Testing	68
3.3 Boundary-Value Analysis	70
3.4 Error Guessing	71
3.5 Discussion	72
Exercises	73
4 Software Testing Roundup	76
4.1 Ideal Test Sets	77
4.2 Operational Testing	80
4.3 Integration Testing	82
4.4 Testing Object-Oriented Programs	84
4.5 Regression Testing	88

4.6	Criteria for Stopping a Test	88
4.7	Choosing a Test-Case Selection Criterion	90
	Exercises	93
5	Analysis of Symbolic Traces	94
5.1	Symbolic Trace and Program Graph	94
5.2	The Concept of a State Constraint	96
5.3	Rules for Moving and Simplifying Constraints	99
5.4	Rules for Moving and Simplifying Statements	110
5.5	Discussion	114
5.6	Supporting Software Tool	126
	Exercises	131
6	Static Analysis	132
6.1	Data-Flow Anomaly Detection	134
6.2	Symbolic Evaluation (Execution)	137
6.3	Program Slicing	141
6.4	Code Inspection	146
6.5	Proving Programs Correct	152
	Exercises	161
7	Program Instrumentation	163
7.1	Test-Coverage Measurement	164
7.2	Test-Case Effectiveness Assessment	165
7.3	Instrumenting Programs for Assertion Checking	166
7.4	Instrumenting Programs for Data-Flow-Anomaly Detection	169
7.5	Instrumenting Programs for Trace-Subprogram Generation	181
	Exercises	192
	Appendix A: Logico-Mathematical Background	194
	Appendix B: Glossary	213
	Appendix C: Questions for Self-Assessment	220
	Bibliography	237
	Index	253