SOFTWARE TESTING Principles and Practices

Naresh Chauhan

OXFORD

UNIVERSITY PRESS

© Oxford University Press 2010 ISBN: 978-0-1980618-47

**Preface**

There is no life without struggles and no software without bugs. Just as one needs to sort out the prob lems in one’s life, it is equally important to check and weed out the bugs in software. Bugs cripple the software in a way problems in life unsettle one. In our life, both joys and sorrows are fl eeting. But a person is best tested in times of crises. One who cultivates an optimistic outlook by displaying an equi poise taking prosperity as well as adversity in his stride and steadily ventures forth on a constructive course is called a sthir pragna. We should follow the same philosophy while testing software too. We need to develop an understanding that unless these bugs appear in our software and until we weed out all of them, our software will not be robust and of superior quality. So, a software test engineer should be an optimist who welcomes the struggles in life and similarly bugs in software, and takes them head on.

Software engineering as a discipline emerged in the late 1960s to guide software development activities in producing quality software. Quality here is not a single-dimensional entity. It has several factors including rigorous software testing. In fact, testing is the critical element of quality and consumes almost half the total development effort. However, it is unfortunate that the quality and testing process does not get its due credit. In software engineering, testing is considered to be a single phase operation performed only after the development of code wherein bugs or errors are removed. However, this is not the case. Testing is not just an intuitive method to remove the bugs, rather it is a systematic process such as software development life cycle (SDLC). The testing process starts as soon as the fi rst phase of SDLC starts. Therefore, even after learning many things about software engineering, there are still some questions and misconceptions regarding the testing process which need to be known, such as the following:

∑ When should testing begin?

∑ How much testing is practically possible?

∑ What are the various techniques to design a good test case (as our knowledge is only limited to black-box and white-box techniques)?

Moreover, the role of software testing as a systematic process to produce quality software is not recognized on a full scale. Many well-proven methods are largely unused in industries today. Companies rely only on the automated testing tools rather than a proper testing methodology. What they need to realize is that Computer-Aided Software Engineering (CASE) environments or tools are there only to assist in the development effort and not meant to serve as silver bullets! Similarly, there are many myths that both students and professionals believe in, which need to be exploded. The present scenario requires *software testing* to be acknowledged as a separate discipline from software engineering. Some universities have already started this course. Therefore, there is a need for a book that explains all these issues for the benefi t of students who will learn software testing and become knowledgeable test engineers as also for the benefi t of test engineers who are already working in the industries and want to hone their testing skills.

**vi** Preface

**ABOUT THE BOOK**

This book treats software testing as a separate discipline to teach the importance of testing process both in academia as well as in the industry. The book stresses on software testing as a systematic process and explains software testing life cycle similar to SDLC and gives insight into the practical importance of software testing. It also describes all the methods/techniques for test case design which is a prime issue in software testing. Moreover, the book advocates the notion of effective software testing in place of exhaustive testing (which is impossible).

The book has been written in a lucid manner and is packed with practical approach of designing the test cases targeting undergraduate and postgraduate students of computer science and engineer ing (B.Tech., M.Tech., MCA), and test engineers. It discusses all the software testing issues and gives insight into their practical importance. Each chapter starts with the learning objectives and ends with a summary containing a quick review of important concepts discussed in the chapter. Some chapters provide solved examples in between the theory to understand the method or technique practically at the same moment. End-chapter exercises and multiple-choice questions are provided to assist instruc tors in classroom teaching and students in preparing better for their exams.

The key feature of the book is a fully devoted case study on Income Tax Calculator which shows how to perform verifi cation and validation at various phases of SDLC. The case study includes ready to-use software and designing of test cases using the techniques described in the book. This material will help both students and testers understand the test design techniques and use them practically.

Apart from the above-mentioned features, the book follows the following methodology in defi ning key concepts in software testing:

∑ Emphasis on software testing as a systematic process

∑ Effective testing concepts rather than exhaustive complete testing

∑ A testing strategy with a complete roadmap has been developed that shows which software test ing technique with how much risk assessment should be adopted at which phase of SDLC

∑ Testing models

∑ Verifi cation and validation as the major components of software testing process. These have been discussed widely expanding in separate chapters.

∑ Software testing life cycle along with bug classifi cation and bug life cycle

∑ Complete categorization of software testing techniques such as static testing and dynamic testing expanding in different chapters

∑ Testing techniques with solved examples to illustrate how to design test cases using these tech niques

∑ Extensive coverage of regression testing, software testing metrics, and test management ∑ Effi cient test suite management to prioritize test cases suitable for a project ∑ The appropriate use of testing tools

∑ Software quality management and test maturity model (TMM)

∑ Testing techniques for two specialized environments: object-oriented software and Web-based software

Preface **vii** l

**ABOUT THE CD**

The CD accompanying the book contains the following:

∑ *Executable fi les* for the examples given in Chapter 5 so that a user can directly implement white box testing on the codes without any extra effort.

∑ *Checklists* for verifi cation of parameters, such as general software design document (SDD), generic code, high level design (HLD), low level design (LLD), and software requirement specifi cation (SRS) document.

∑ *A program on Income Tax Calculator* along with its description in the form of a case study that illustrates all the steps of the software testing process.

**CONTENT AND COVERAGE**

The book has been divided into seven different parts. Each part further consists of various chapters.

Part I (Testing Methodology) introduces concepts such as effective software testing, testing terminology, testing as a process, and development of testing methodology.

*Chapter 1* introduces the concept of effective testing versus complete testing, explains the psychology for performing effective testing, and establishes that software testing is a complete process.

*Chapter 2* discusses the commonly used testing terminology such as error, bug, and failure, explains life cycle of a bug with its various states, phases of software testing life cycle and V testing model, and development of a testing methodology.

*Chapter 3* explains how verifi cation and validation, a part of testing strategy, are performed at various phases of SDLC.

Part II (Testing Techniques) deals with various test case design techniques based on static testing and dynamic testing and verifi cation and validation concepts.

*Chapter 4* covers test case design techniques using black-box testing including boundary value analysis, equivalence class partitioning method, state table based testing, decision table based testing, and cause-effect graphing technique.

*Chapter 5* discusses test case design techniques using white-box testing, including basis path testing, loop testing, data fl ow testing, and mutation testing.

*Chapter 6* deals with the techniques, namely inspection, walkthrough, and reviews, largely used for verifi cation of various intermediate work products resulting at different stages of SDLC.

*Chapter 7* discusses various techniques used in validation testing such as unit testing, integration testing, function testing, system testing, and acceptance testing.

*Chapter 8* describes regression testing that is used to check the effect of modifi cations on other parts of software.

Part III (Managing the Testing Process) discusses how to manage the testing process, various per sons involved in test organization hierarchy, testing metrics to monitor and control the testing process, and how to reduce the number of test cases.

*Chapter 9* covers the concept of introduction of management for test process for its effectiveness. Various persons involved in test management hierarchy are discussed. The test planning

**viii** Preface

for various verifi cation and validation activities are also discussed along with the test result specifi cations.

*Chapter 10* provides an introductory material to understand that measurement is a necessary part of software engineering, known as software metrics.

*Chapter 11* explains how software metrics assist in monitoring and controlling different testing activities.

*Chapter 12* explains the fact that test cases, specially designed for system testing and regression testing, become unmanageable in a way that we cannot test all of them. The problem is how to select or reduce the test cases out of a big test suite. This chapter discusses many such techniques to resolve the problem.

Part IV (Quality Management) covers software quality issues with some standards along with testing process maturity models.

*Chapter 13* discusses various terminologies, issues, and standards related to software quality management to produce high quality software.

*Chapter 14* discusses various test process maturity models, namely test improvement model (TIM), test organization model (TOM), test process improvement (TPI), and test maturity model (TMM).

Part V (Test Automation) discusses the need of testing and provides an introduction to testing tools.

*Chapter 15* explains the need for automation, categories of testing tools, and how to select a testing tool.

Part VI (Testing for Specialized Environment) introduces the testing environment and the issues related to two specialized environments, namely object-oriented software and Web-based software.

*Chapters 16 and 17* discuss the issues, challenges, and techniques related to object-oriented and Web based software, respectively.

Part VII (Tracking the Bug) explains the process and techniques of debugging.

*Chapter 18* covers the debugging process and discusses various methods to debug a software product.

The book concludes with a case study of Income Tax Calculator illustrating the testing of software using verifi cation and validation techniques. In addition, the book contains useful appendices that pro vide various ready-to-use checklists which can be used at the time of verifi cation of an item at various stages of SDLC.

Do send your valuable suggestions, comments, and constructive criticism for further improvement of the book.

Naresh Chauhan

**Step 1**

Introduction to Case Study **513**l

**INCOME TAX CALCULATOR A Case Study**

**Introduction to Case Study**

All the techniques learnt in this book can be practised using a case study. For this purpose, a case study of *Income Tax Calculator* application has been taken. The application has been designed and developed for the readers and all the test case design techniques have been applied on it. However, the application presented and implemented is only for illustrative purposes and it is not claimed that this application is free from defects and can be used practically for calculating the in

come tax of a person. The idea is only to present a working application and show how to perform testing on it.

The case study has been presented in the following sequence:

**Requirement Specifi cations and Verifi cation**

The requirements for the case study have been collected and SRS ver 1.0 was prepared initially. The tax slabs and other details in this case study have been compiled from *www.incometaxindia.gov.in*. This draft of SRS was in a raw form. After this, verifi cation on SRS ver 1.0 was performed and found that many fea

tures were not present in SRS. During verifi cation on SRS, the checklist pre sented in Appendix has been used. The readers are advised that they should also perform verifi cation using checklists and fi nd some more defi ciencies in SRS. In this way, SRS ver 2.0 was prepared as a result of verifi cation on SRS ver 1.0. Another round of verifi cation was performed on SRS ver 2.0 and fi nally we get SRS ver 3.0.

**514** Software Testing: Principles and Practices

The readers are advised to prepare an SDD of this application and perform verifi cation exercises on it to and get a fi nal version of SDD.

**Black-box Testing on SRS ver 3.0**

Once the SRS is prepared, some black-box test cases have been designed using the techniques studied in Chapter 4. The test cases can be executed on the imple mented executable application. The executable application can be directly taken from the CD.

**Source Code**

The application based on SRS ver 3.0 has been implemented in C language. There are two fi les: *TaxCalculator.c* and *Taxcalculator.h*. The readers can get these fi les directly from the CD and use and modify them the way they want.

**White-Box Testing**

The source code of *TaxCalculator.h* has been tested using white-box testing tech niques. All the major white-box testing techniques have been applied on this source code. The test cases can be executed on the implemented executable ap plication. The executable application along with the source code of application can be taken directly from the CD.

The readers should follow this sequence for studying the full case study and learn the testing techniques presented in this book. The case study provides the way to learn the testing concepts and techniques in a practical way.

**Step 2**

Income Tax Calculator SRS ver 1.0 **515**l

**Income Tax Calculator SRS ver 1.0**

A system is proposed to calculate the income tax of a person residing in India, provided his salary, savings, status, and donations are known. The system will accept personal details, income details, savings details and calculate total salary, net tax payable, educational cess, and hence the total tax payable.

Income Tax slabs 2009/2010 for Men

| Income: up to 1.5 lacs | NO INCOME TAX |
| --- | --- |
| Income : 1.5 lacs to 3 lacs | 10% |
| Income : 3 lacs to 5 lacs | 20% |
| Income : above 5 lacs | 30% |

Income Tax slabs 2009/2010 for Women

| Income : up to 1.8 lacs | NO TAX |
| --- | --- |
| Income : 1.8 lacs to 3 lacs | 10% |
| Income : 3 lacs to 5 lacs | 20% |
| Income : above 5 lacs | 30% |

Income Tax slabs 2009/2010 for Senior Citizen

| Income : up to 2.25 lacs | NO TAX |
| --- | --- |
| Income : 2.25 lacs to 3 lacs | 10% |
| Income : 3 lacs to 5 lacs | 20% |
| Income : above 5 lacs | 30% |

In addition to the income tax calculated according to the above income tax slabs, a 3% of education cess will be charged on the total income tax paid (not on the total taxable income). If the taxable income exceeds Rs 10 lacs, a 10% surcharge on the total income tax (not on the total taxable income) is also charged.

**516** Software Testing: Principles and Practices

**Donations with 100% rebate**

The Prime Minister’s National Relief Fund.

The Prime Minister’s Armenia Earthquake Relief Fund.

The Africa (Public Contributions-India) Fund.

The National Foundation for Communal Harmony.

A university or any educational institution of national eminence as may be approved by the prescribed authority. Please note that the prescribed authority in case of a university or a non-technical institution of national eminence is the Director-General (Income-Tac exemption) in concurrence with the Secretary, UGC. In case of any technical institution of national eminence, the prescribed authority is the Director-General (Income-Tax Exemption) in concurrence with the Secretary, All India Council of Techni cal Education.

The Maharashtra Chief Minister’s Earthquake Relief Fund.

Any Zila Saksharta Samiti constituted in any district under the chairman ship of the Collector of that district for the purpose of improvement of pri mary education in villages and towns in such a district and for literacy and post literacy activities.

The National Blood Transfusion Council or any State Blood Transfusion council whose sole objective is the control, supervision, regulation, or en couragement in India of the services related to operation and requirements of blood banks.

Any fund set up by a State Government to provide medical relief to the poor.

The Army Central Welfare Fund or the Indian Naval Benevolent Fund or the Air Force Central Welfare Fund established by the armed forces of the Union for the welfare of the past and present members of such forces or their dependants.

The Andhra Pradesh Chief Minister’s Cyclone Relief Fund, 1996.

The National Illness Assistance Fund.

The Chief Minister’s Relief Fund or the Lieutenant Governor’s Relief Fund in any State or Union Territory.

The Government, or any local authority, institution or association as maybe approved by the Central Government for the purpose of promoting family planning.

**Step 3**

Verifi cation on Income Tax Calculator SRS ver 1.0 **517**l

**Verifi cation on Income Tax Calculator SRS ver 1.0**

*Verifi cation on Income Tax Calculator SRS ver 1.0 is presented here. The reader is advised to use checklists provided in the Appendix while performing verifi cation and use his/her intelligence. The missing features found in this verifi cation are highlighted.*

A system is proposed to calculate the income tax of a person residing in India provided his salary, savings, status, and donations are known. The system will accept personal details, income details, savings details and calculate total salary, net tax payable, educational cess, and hence total tax payable.

*Is the software meant only for salaried person or for anyone? There is no mention about the functional fl ow of the system about how it works. Who will interface with the system? How are the savings considered in calculating the net tax? There is no high-level functional ity diagram representing interfaces and data fl ow.*

Income Tax slabs 2009/2010 for Men

| Income: up to 1.5 lacs | NO INCOME TAX |
| --- | --- |
| Income : 1.5 lacs to 3 lacs | 10% |
| Income : 3 lacs to 5 lacs | 20% |
| Income : above 5 lacs | 30% |

Income Tax slabs 2009/2010 for Women

| Income : up to 1.8 lacs | NO TAX |
| --- | --- |
| Income : 1.8 lacs to 3 lacs | 10% |
| Income : 3 lacs to 5 lacs | 20% |
| Income : above 5 lacs | 30% |

Income Tax slabs 2009/2010 for Senior Citizen

| Income : up to 2.25 lacs | NO TAX |
| --- | --- |
| Income : 2.25 lacs to 3 lacs | 10% |
| Income : 3 lacs to 5 lacs | 20% |
| Income : above 5 lacs | 30% |

**518** Software Testing: Principles and Practices

In addition to the income tax calculated according to the above income tax slabs, a 3% of education cess will be charged on the total income tax paid (not on the total taxable income). If the taxable income exceeds Rs 10 lacs, a 10% sur charge on the total income tax (not on the total taxable income) is also charged

**Donations with 100% rebate**

The Prime Minister’s National Relief Fund.

The Prime Minister’s Armenia Earthquake Relief Fund.

The Africa (Public Contributions-India) Fund.

The National Foundation for Communal Harmony.

A University or any educational institution of national eminence as maybe approved by the prescribed authority. In case of any technical institution of national eminence, the prescribed authority is the Director General (In come-Tax Exemption) in concurrence with the Secretary, All India Council of Technical Education.

The Maharashtra Chief Minister’s Earthquake Relief Fund.

Any Zila Saksharta Samiti constituted in any district under the chairman ship of the Collector of that district for the purpose of improvement of primary education in villages and towns in such a district and for literacy and post literacy activities.

The National Blood Transfusion Council or any State Blood Transfusion council whose sole objective is the control, supervision, regulation, or en couragement in India of the services related to operation and requirements of blood banks.

Any fund set up by a State Government to provide medical relief to the poor.

The Army Central Welfare Fund or the Indian Naval Benevolent Fund or the Air Force Central Welfare Fund established by the armed forces of the Union for the welfare of the past and present members of such forces or their dependants.

The Andhra Pradesh Chief Minister’s Cyclone Relief Fund, 1996.

The National Illness Assistance Fund.

The Chief Minister’s Relief Fund or the Lieutenant Governor’s Relief Fund in any State or Union Territory.

The Government, or any local authority, institution or association as maybe approved by the Central Government for the purpose of promoting family planning.

Verifi cation on Income Tax Calculator SRS ver 1.0 **519**l

What will be the rebate if the donation is not in the above list? In addition, the following items are missing in the SRS which are necessary to avoid bugs and misunderstanding:

1. High-level diagrams depicting external and internal interfaces.

2. The user interaction with the system.

3. Software Functions/Features.

4. Inputs and outputs formats and their ranges.

5. Software/Hardware Requirements.

**520** Software Testing: Principles and Practices

**Step**

**4**

**Income Tax Calculator SRS ver 2.0**

A system is proposed to calculate the income tax of a person residing in India provided his income, savings, status, and donations are known. The system will accept personal details, income details, and savings details, and calculate the total salary, net tax payable, educational cess, and hence the total tax payable. The user gets the information about total tax to be paid.

Personal

details

Income

details

Saving

details

Donation

details

Tax payable

**Tax calculator** details **system**

Tax deductions

details

The system will fi rst accept personal details, income, donations, and savings. For donations, it provides a list of categories in which 100% rebate is provided. The user will look for the option provided and inform the system whether the donation lies in that list. If the donation lies in the list, 100% rebate will be pro

vided, otherwise 50%. The system will check whether the savings are less than Rs 1 lac. If yes, then the whole amount will be deducted from the taxable income. Otherwise, Rs 1 lac will be deducted. Then the system will calculate the total tax and check if it exceeds Rs 10 lacs. If yes, a 10% surcharge on the total income tax (not on the total taxable income) is also charged and a 3% of education cess will be charged on the total income tax paid (not on the total taxable income). Finally, the system will show the net tax as per the following details:

Income Tax Calculator SRS ver 2.0 **521**l

Income Tax slabs 2009/2010 for Men

| Income: up to 1.5 lacs | NO TAX |
| --- | --- |
| Income : 1.5 lacs to 3 lacs | 10% |
| Income : 3 lacs to 5 lacs | 20% |
| Income : above 5 lacs | 30% |

Income Tax slabs 2009/2010 for Women

| Income : up to 1.8 lacs | NO TAX |
| --- | --- |
| Income : 1.8 lacs to 3 lacs | 10% |
| Income : 3 lacs to 5 lacs | 20% |
| Income : above 5 lacs | 30% |

Income Tax slabs 2009/2010 for Senior Citizen

| Income : up to 2.25 lacs | NO TAX |
| --- | --- |
| Income : 2.25 lacs to 3 lacs | 10% |
| Income : 3 lacs to 5 lacs | 20% |
| Income : above 5 lacs | 30% |

**Donations with 100% rebate**

The Prime Minister’s National Relief Fund.

The Prime Minister’s Armenia Earthquake Relief Fund.

The Africa (Public Contributions-India) Fund.

The National Foundation for Communal Harmony.

A university or any educational institution of national eminence as maybe approved by the prescribed authority. In case of any technical institution of national eminence, the prescribed authority is the Director General (In come-Tax Exemption) in concurrence with the Secretary, All India Council of Technical Education.

The Maharashtra Chief Minister’s Earthquake Relief Fund.

Any Zila Saksharta Samiti constituted in any district under the chairman ship of the Collector of that district for the purpose of improvement of pri mary education in villages and towns in such a district and for literacy and post literacy activities.

The National Blood Transfusion Council or any State Blood Transfusion council whose sole objective is the control, supervision, regulation, or en couragement in India of the services related to operation and requirements of blood banks.

Any fund set up by a State Government to provide medical relief to the poor.

The Army Central Welfare Fund or the Indian Naval Benevolent Fund or the Air Force Central Welfare Fund established by the armed forces of the Union for the welfare of the past and present members of such forces or their dependants.

**522** Software Testing: Principles and Practices

The Andhra Pradesh Chief Minister’s Cyclone Relief Fund, 1996.

The National Illness Assistance Fund.

The Chief Minister’s Relief Fund or the Lieutenant Governor’s Relief Fund in any State or Union Territory.

The Government, or any local authority, institution or association as maybe approved by the Central Government for the purpose of promoting family planning.

**Functional Requirements**

Accept personal

details

Accept income

details

Accept savings

and donations

details

User

Accept tax

deduction details

Generate tax

payable details

**Accept Personal Details**

The function will accept the following details to be entered by user.

Name (3 to 15 alphabets with spaces in between)

Date of Birth (dd/mm/yyyy)

Permanent address (3 to 30 characters)

Sex (M/F one alphabet only)

Status: Salaried or not (Y/N one alphabet only)

If the user enters the answer Y (Yes) to the status entry, then the function will display the following three entries, otherwise it will not.

Designation (if Salaried) (3 to 15 alphabets)

Name of the employer (if salaried) (3 to 25 alphabets with spaces)

Address of the employer (if salaried) (3 to 30 characters)

PAN number (10 characters including alphabets and digits 0–9)

Income Tax Calculator SRS ver 2.0 **523**l

TDS circle where annual return/statement under section 206 is to be fi led (3 to 15 alphabets with spaces)

Period: From (dd/mm/yyyy)

To (dd/mm/yyyy)

Assessment year (yyyy-yy)

**Accept Income Details**

The function will enquire whether the user is a salaried person or has some other source of income. If the user is not a salaried person, the system will ask for the source of income. The user may enter various types of source of incomes as given below:

Source of Income: (3 to 20 alphabets with spaces)

Amount: (positive real numbers with maximum two decimal places)

The function will aggregate all the amounts of income as gross total income. If the person is salaried, the function asks for the following details:

1. Gross Salary

(a) Salary as per the provisions contained in the section 17(1)

(b) Value of the prerequisites under section 17(2) (As per form number 12BA, wherever applicable)

(c) Profi ts in lieu of salary under section 17(3) (As per form number 12BA, wherever applicable)

(d) Total (to be calculated by this function)

2. Less allowance to the extent exempt under section 10

This function will add the exempted allowances.

3. Balance

This function will calculate the difference of the gross salary and the ex empted allowances.

4. Deductions

Entertainment allowance

Tax on employment

5. Aggregate

This function will calculate the aggregate of the deductions entered above. 6. Income Chargeable Under The Head ‘Salaries’

The function will calculate the difference of item 3–item 5.

7. Add

Any other item reported by the employee.

User may enter multiple incomes. The function will add all these incomes. 8. Gross Total Income

The function will add item 6 and 7.

All the amounts will be positive real numbers with maximum 2 decimal places.

**524** Software Testing: Principles and Practices

**Accept Savings & Donations Details**

The function will ask the user to enter the total savings and the donations in the following format.

Saving type (3 to 20 alphabets with spaces)

Deductible amount (positive real numbers with maximum two decimal places)

The user may enter multiple savings. The function will add all the deductible amounts in aggregate deductible amount.

**Accept Tax Deduction Details**

If the person is salaried, then this function will accept the details if tax deducted by the employer during the year is in the following format:

Amount of tax deposited (positive real numbers with maximum two decimal places)

Date (dd/mm/yyyy)

Challan Number (5 to 20 characters)

The above details may be entered multiple times. The function will add all the amounts of tax deposited.

Amount of TDS (positive real numbers with maximum two decimal places)

The function will add all the amounts of tax deposited and the amount of TDS in the total tax deducted.

**Generate Tax Payable Details**

This function calculates the tax payable by the person in the following format: *Taxable income* Function will calculate this by taking difference of gross total income in the function Accept Income Details and aggregate deductible amount in the function Accept Savings & Donation Details.

*Tax on taxable income* Function will calculate this using the appropriate slab of user as given above.

*Surcharge* Function will calculate the surcharge as if tax on taxable income exceeds Rs 10 lacs, a 10% surcharge is charged.

*Education cess* Function will calculate the education cess as a 3% of tax on taxable income.

*Tax payable* Function will sum up tax on taxable income, surcharge, and education cess.

*Relief under section 89* User will enter the amount, if applicable.

*Tax payable after relief (if applicable)* Function will deduct relief amount from tax payable.

*Total tax deducted* Displayed from the function Accept Tax Deduction Details. *Tax payable/refundable* The function will fi nd the difference of tax payable and the total tax deducted. If the difference is positive, then this amount is the net tax to be paid by the person, otherwise the amount is due on the government to be refunded.

**Step 5**

Verifi cation on Income Tax Calculator SRS ver 2.0 **525**l

**Verifi cation on Income Tax Calculator SRS ver 2.0**

*Verifi cation on Income Tax Calculator SRS ver 2.0 is presented here. The reader is advised to use checklists while performing verifi cation and use his/her intelligence. The missing features found in this verifi cation are highlighted.*

A system is proposed to calculate the income tax of a person residing in India provided his income, savings, status, and donations are known. The system will accept personal details, income details, savings details, and calculate total salary, net tax payable, educational cess, and hence total tax payable. The user gets the information about total tax to be paid.

Personal

details

Income

details

Saving

details

Donation

details

Tax payable

**Tax calculator** details **system**

Tax deductions

details

The system will fi rst accept personal details, income, donations, and savings. For donations, it provides a list of categories in which 100% rebate is provided. The user will look for the option provided and informs the system whether the donation lies in that list. If the donation lies in the list, 100% rebate will be pro

vided otherwise 50%. The system will check whether the savings are less than Rs 1 lac. If yes, then whole amount will be deducted from the taxable income. Otherwise, Rs 1 lac will be deducted. Then the system will calculate the total tax and checks if it exceeds Rs 10 lacs, a 10% surcharge on the total income tax (not on the total taxable income) is also charged and a 3% of education cess will be charged on the total income tax paid (not on the total taxable income). Finally, the system will show the net tax as per the following details:

**526** Software Testing: Principles and Practices

Income Tax slabs 2009/2010 for Men

| Income: up to 1.5 lacs | NO TAX |
| --- | --- |
| Income : 1.5 lacs to 3 lacs | 10 % |
| Income : 3 lacs to 5 lacs | 20 % |
| Income : above 5 lacs | 30 % |

Income Tax slabs 2009/2010 for Women

| Income : up to 1.8 lacs | NO TAX |
| --- | --- |
| Income : 1.8 lacs to 3 lacs | 10 % |
| Income : 3 lacs to 5 lacs | 20 % |
| Income : above 5 lacs | 30 % |

Income Tax slabs 2009/2010 for Senior Citizen

| Income : up to 2.25 lacs | NO TAX |
| --- | --- |
| Income : 2.25 lacs to 3 lacs | 10 % |
| Income : 3 lacs to 5 lacs | 20 % |
| Income : above 5 lacs | 30 % |

**Donations with 100% rebate**

The Prime Minister’s National Relief Fund.

The Prime Minister’s Armenia Earthquake Relief Fund.

The Africa (Public Contributions-India) Fund.

The National Foundation for Communal Harmony.

A University or any educational institution of national eminence as maybe approved by the prescribed authority. In case of any technical institution of national eminence, the prescribed authority is the Director General (In come-Tax Exemption) in concurrence with the Secretary, All India Council of Technical Education.

The Maharashtra Chief Minister’s Earthquake Relief Fund.

Any Zila Saksharta Samiti constituted in any district under the chairman ship of the Collector of that district for the purpose of improvement of pri mary education in villages and towns in such a district and for literacy and post literacy activities.

The National Blood Transfusion Council or any State Blood Transfusion council whose sole objective is the control, supervision, regulation, or en couragement in India of the services related to operation and requirements of blood banks.

Any fund set up by a State Government to provide medical relief to the poor.

Verifi cation on Income Tax Calculator SRS ver 2.0 **527**l

The Army Central Welfare Fund or the Indian Naval Benevolent Fund or the Air Force Central Welfare Fund established by the armed forces of the Union for the welfare of the past and present members of such forces or their dependants.

The Andhra Pradesh Chief Minister’s Cyclone Relief Fund, 1996.

The National Illness Assistance Fund.

The Chief Minister’s Relief Fund or the Lieutenant Governor’s Relief Fund in any State or Union Territory.

The Government, or any local authority, institution or association as maybe approved by the Central Government for the purpose of promoting family planning.

**Functional Requirements**

Accept personal

details

Accept income

details

Accept savings

and donations

details

User

Accept tax

deduction details

Generate tax

payable details

**Accept Personal Details**

The function will accept the following details to be entered by user.

Name (3 to 15 alphabets with spaces in between)

Date of Birth (dd/mm/yyyy)

Permanent address (3 to 30 characters)

*The address should not contain any character. The allowed characters like alphabets, digits, spaces, and commas should be mentioned.*

Sex (M/F one alphabet only)

Salaried or not (Y/N one alphabet only)

**528** Software Testing: Principles and Practices

Designation (if Salaried) (3 to 15 alphabets)

Name of the employer (if salaried) (3 to 25 alphabets with spaces)

Address of the employer (if salaried) (3 to 30 characters)

PAN number (10 characters including alphabets and digits 0-9)

*Including* means any character can be entered. The word should be *consisting*.

TDS circle where annual return/ statement under section 206 is to be fi led (3 to 15 alphabets with spaces)

Period: From (dd/mm/yyyy)

To (dd/mm/yyyy)

Assessment year (yyyy-yy)

*It is not clear whether the user can make a wrong entry and move ahead to the next entry or he cannot move ahead until he enters a correct entry.*

**Accept Income Details**

The function will enquire whether the user is a salaried person or has some other source of income. If the user is not a salaried person, the system will ask for the source of income. The user may enter various types of source of incomes as given below:

Source of Income: (3 to 20 alphabets with spaces)

Amount: (positive real numbers with maximum two decimal places)

The function will aggregate all the amounts of income as gross total income. If the person is salaried, the function asks for the following details:

1. Gross Salary

(a) Salary as per the provisions contained in the section 17(1)

(b) Value of the prerequisites under section 17(2) (As per form number 12BA, wherever applicable)

(c) Profi ts in lieu of salary under section 17(3) (As per form number 12BA, wherever applicable)

(d) Total (to be calculated by this function)

2. Less allowance to the extent exempt under section 10

This function will add the exempted allowances.

3. Balance

This function will calculate the difference of the gross salary and the ex empted allowances.

4. Deductions

Entertainment allowance

Tax on employment

Verifi cation on Income Tax Calculator SRS ver 2.0 **529**l

5. Aggregate

This function will calculate the aggregate of the deductions entered above. 6. Income Chargeable Under the Head ‘Salaries’

The function will calculate the difference of item 3 – item 5.

7. ADD

Any other item reported by the employee

User may enter multiple incomes. The function will add all these incomes. 8. Gross Total Income

The function will add item 6 and 7.

All the amounts will be positive decimal numbers with maximum 2 deci mal places.

*It is not clear whether the user can make a wrong entry and move ahead on the next entry or he cannot move ahead until he enters a correct entry.*

**Accept Savings & Donations Details**

The function will ask the user to enter the total savings and the donations in the following format.

Saving Type: (3 to 20 alphabets with spaces)

Deductible amount: (positive real numbers with maximum two decimal places)

The user may enter multiple savings. The function will add all the deductible amounts in aggregate deductible amount.

*It is not clear whether the user can make a wrong entry and move ahead to the next entry or he cannot move ahead until he enters a correct entry.*

**Accept Tax Deduction Details**

If the person is salaried, then this function will accept the details if tax deducted by the employer during the year in the following format:

Amount of tax deposited (positive real numbers with maximum two decimal places)

Date (dd/mm/yyyy)

Challan Number (5 to 20 characters)

The above details may be entered multiple times. The function will add all the amounts of tax deposited.

*Amount of TDS* The function will add all the amounts of tax deposited and amount of TDS in total tax deducted.

*It is not clear whether the user can make a wrong entry and move ahead to the next entry or he cannot move ahead until he enters a correct entry.*

**530** Software Testing: Principles and Practices

**Generate Tax Payable Details**

This function calculates the tax payable by the person in the following format:

*Taxable income* The function will calculate this by taking difference of gross total income in the function Accept Income Details and aggregate deductible amount in the function Accept Savings & Donation Details.

*Tax on taxable income* The function will calculate this using the appropriate slab of user as given above.

*Surcharge* The function will calculate the surcharge as if tax on taxable income exceeds Rs 10 lacs, a 10% surcharge is charged.

*Education cess* The function will calculate the education cess as a 3% of tax on taxable income.

*Tax payable* The function will sum up the tax on taxable income, surcharge, and education cess.

*Relief under section 89* User will enter the amount, if applicable.

*Tax payable after relief (if applicable)* The function will deduct the relief amount from the tax payable.

*Total tax deducted* Displayed from the function Accept Tax Deduction Details.

*Tax payable/refundable* The function will fi nd difference of tax payable and total tax deducted. If difference is positive, then this amount is the net tax to be paid by the person, otherwise the amount is due on the government to be refunded.

User interface requirements and system requirements are not men tioned.

**Step 6**

Income Tax Calculatro SRS ver 3.0 **531**l

**Income Tax Calculator SRS ver 3.0**

A system is proposed to calculate the income tax of a person residing in India provided his income, savings, status, and donations are known. The system will accept personal details, income details, savings details and calculate total salary, net tax payable, educational cess, and hence total tax payable. The user gets the information about total tax to be paid.

Personal

details

Income

details

Saving

details

Donation

details

Tax payable

**Tax calculator** details **system**

Tax deductions

details

The system will fi rst accept personal details, income, donations, and savings. For donations, it provides a list of categories in which 100% rebate is provided. The user will look for the option provided and informs the system whether the donation lies in that list. If the donation lies in the list, 100% rebate will be pro

vided, otherwise 50%. The system will check whether the savings are less than Rs 1 lac. If yes, then the whole amount will be deducted from the taxable income. Otherwise, Rs 1 lac will be deducted. Then the system will calculate the total tax and checks if it exceeds Rs 10 lacs, a 10% surcharge on the total income tax (not on the total taxable income) is also charged and a 3% of education cess will be charged on the total income tax paid (not on the total taxable income). Finally, the system will show the net tax as per the following details:

**532** Software Testing: Principles and Practices

Income Tax slabs 2009/2010 for Men

| Income: up to 1.5 lacs | NO TAX |
| --- | --- |
| Income : 1.5 lacs to 3 lacs | 10% |
| Income : 3 lacs to 5 lacs | 20% |
| Income : above 5 lacs | 30% |

Income Tax slabs 2009/2010 for Women

| Income : up to 1.8 lacs | NO TAX |
| --- | --- |
| Income : 1.8 lacs to 3 lacs | 10% |
| Income : 3 lacs to 5 lacs | 20% |
| Income : above 5 lacs | 30% |

Income Tax slabs 2009/2010 for Senior Citizen

| Income : up to 2.25 lacs | NO TAX |
| --- | --- |
| Income : 2.25 lacs to 3 lacs | 10% |
| Income : 3 lacs to 5 lacs | 20% |
| Income : above 5 lacs | 30% |

**Donations with 100% rebate**

The Prime Minister’s National Relief Fund.

The Prime Minister’s Armenia Earthquake Relief Fund.

The Africa (Public Contributions-India) Fund.

The National Foundation for Communal Harmony.

A University or any educational institution of national eminence as maybe approved by the prescribed authority. In case of any technical institution of national eminence, the prescribed authority is the Director General (In come-Tax Exemption) in concurrence with the Secretary, All India Council of Technical Education.

The Maharashtra Chief Minister’s Earthquake Relief Fund.

Any Zila Saksharta Samiti constituted in any district under the chairman ship of the Collector of that district for the purpose of improvement of primary education in villages and towns in such a district and for literacy and post literacy activities.

The National Blood Transfusion Council or any State Blood Transfusion council whose sole objective is the control, supervision, regulation, or en couragement in India of the services related to operation and requirements of blood banks.

Any fund set up by a State Government to provide medical relief to the poor.

The Army Central Welfare Fund or the Indian Naval Benevolent Fund or the Air Force Central Welfare Fund established by the armed forces of the

Income Tax Calculatro SRS ver 3.0 **533**l

Union for the welfare of the past and present members of such forces or their dependants.

The Andhra Pradesh Chief Minister’s Cyclone Relief Fund, 1996.

The National Illness Assistance Fund.

The Chief Minister’s Relief Fund or the Lieutenant Governor’s Relief Fund in any State or Union Territory.

The Government, or any local authority, institution or association as maybe approved by the Central Government for the purpose of promoting family planning.

**Functional Requirements**

Accept personal

details

Accept income

details

Accept savings

and donations

details

User

Accept tax

deduction details

Generate tax

payable details

**Accept Personal Details (APD)**

The function will accept the following details to be entered by the user. The user cannot move to the next entry unless he enters the correct entry.

Name (3 to 15 alphabets with spaces in between)

Date of Birth (dd/mm/yyyy)

Permanent address (3 to 30 characters) The allowed characters are alpha bets, digits, spaces, and commas only.

Sex (M/F one alphabet only)

Status: Salaried or not (Y/N one alphabet only)

If the user enters the answer Y (Yes) to the status entry, then the function will display the following three entries, otherwise it will not.

**534** Software Testing: Principles and Practices

Designation (if salaried) (3 to 15 alphabets)

Name of the employer (if salaried) (3 to 25 alphabets with spaces)

Address of the employer (if salaried) (3 to 30 characters)

PAN number (10 characters consisting alphabets and digits 0–9)

TDS circle where annual return/statement under section 206 is to be fi led (3 to 15 alphabets with spaces)

Period: From (dd/mm/yyyy)

To (dd/mm/yyyy)

Assessment year (yyyy-yy)

**Accept Income Details (AID)**

The function will enquire whether the user is a salaried person or has some other source of income. If the user is not a salaried person, the system will ask for the source of income. The user may enter various types of source of incomes as given below. The user cannot move to the next entry unless he enters the correct entry.

Source of Income: (3 to 20 alphabets with spaces)

Amount: (positive real numbers with maximum two decimal places)

The function will aggregate all the amounts of income as gross total income. If the person is salaried, the function asks for the following details:

1. Gross Salary

(a) Salary as per the provisions contained in the section 17(1)

(b) Value of the perquisites under section 17(2) (As per form number 12BA, wherever applicable)

(c) Profi ts in lieu of salary under section 17(3) (As per form number 12BA, wherever applicable)

(d) Total (to be calculated by this function)

2. Less allowance to the extent exempt under section 10

This function will add the exempted allowances.

3. Balance

This function will calculate the difference of the gross salary and the ex empted allowances.

4. Deductions

Entertainment allowance (EA)

Tax on employment (TE)

5. Aggregate

This function will calculate the aggregate of the deductions entered above. 6. Income Chargeable Under the Head ‘Salaries’

The function will calculate the difference of item 3 – item 5.

Income Tax Calculatro SRS ver 3.0 **535**l

7. Add

Any other item reported by the employee

User may enter multiple incomes. The function will add all these incomes.

8. Gross Total Income

The function will add item 6 and 7.

All the amounts will be positive real numbers with maximum 2 decimal places.

**Accept Savings & Donations Details (ASD)**

The function will ask the user to enter the total savings and the donations in the following format.

Saving Type: (3 to 20 alphabets with spaces)

Deductible amount: (positive real numbers with maximum two decimal places)

The user may enter multiple savings. The function will add all the deductible amounts in aggregate deductible amount.

The user cannot move to the next entry unless he enters the correct entry.

**Accept Tax Deduction Details**

If the person is salaried, then this function will accept the details if tax deducted by the employer during the year in the following format:

Amount of TDS: (positive real numbers with maximum two decimal places) Amount of tax deposited (positive real numbers with maximum two deci mal places)

Date (dd/mm/yyyy)

Challan Number (5 to 20 characters)

The above details may be entered multiple times. The function will add all the amounts of tax deposited and amount of TDS in total tax deducted.

The user cannot move to the next entry unless he enters the correct entry.

**Generate Tax Payable Details**

This function calculates the tax payable by the person in the following format: *Taxable income* Function will calculate this by taking difference of gross total income in the function Accept Income Details and aggregate deductible amount in the function Accept Savings & Donation Details.

*Tax on taxable income* Function will calculate this using the appropriate slab of user as given above.

*Surcharge* Function will calculate the surcharge as if tax on taxable income exceeds Rs 10 lacs, a 10% surcharge is charged.

*Education cess* Function will calculate the education cess as a 3% of tax on tax able income.

*Tax payable* Function will sum up tax on taxable income, surcharge, and educa tion cess.

**536** Software Testing: Principles and Practices

*Total tax deducted* Displayed from the function Accept Tax Deduction Details. *Tax payable/refundable* The function will fi nd difference of tax payable and total tax deducted. If difference is positive, then this amount is the net tax to be paid by the person, otherwise the amount is due on the government to be refunded.

**USER INTERFACE REQUIREMENTS**

**Personal Detail Screen**

The personal detail screen is displayed wherein the user can enter his various details as given below:

Name

Date of Birth

Permanent address

Sex (M/F)

Status (Salaried or not): Y/N

*If the user presses Y, then the following screen is displayed:*

Designation

Name of the employer

Address of the employer

PAN number

TDS circle where annual return/statement under section 206 is to be fi led Period: From (dd/mm/yyyy)

To (dd/mm/yyyy)

Assessment year (yyyy-yy)

*If the user presses N, then the following screen is displayed:*

PAN number

TDS circle where annual return/statement under section 206 is to be fi led Period: From (dd/mm/yyyy)

To (dd/mm/yyyy)

Assessment year (yyyy-yy)

**Income Details Screen**

The income detail screen is displayed wherein the user can enter his various de tails as given below.

*If the user is not a salaried person, the system will ask for the source of income in the following screen:*

Source of Income:

Amount:

Enter more? (Y/N)

Income Tax Calculatro SRS ver 3.0 **537**l

*If the person is salaried, the function asks for the following details:*

1. Gross Salary

(a) Salary as per the provisions contained in the section 17(1)

(b) Value of the perquisites under section 17(2) (As per form number 12BA, wherever applicable)

(c) Profi ts in lieu of salary under section 17(3) (As per form number 12BA, wherever applicable)

(d) Total (to be calculated by this function)

2. Allowance to the extent exempt under section 10

Enter more? (Y/N)

3. Deductions

Entertainment allowance (EA)

Tax on employment (TE)

4. Income Chargeable Under the Head ‘Salaries’

Displayed by the system

5. Any other item reported by the employee

Enter Income:

Enter more? (Y/N)

6. Gross Total Income: Displayed by the system

**Savings & Donations Details Screen**

The saving detail screen is displayed wherein the user can enter his various de tails as given below:

Saving Type: (3 to 20 alphabets with spaces)

Deductible amount: Enter more? (Y/N)

**Tax Deduction Details Screen**

If the person is salaried, then tax deducted by the employer during the year in the following format is entered:

Amount of tax deposited

Date

Challan Number

Enter more? (Y/N)

Amount of TDS:

**Tax Payable Detail Screen**

This screen is the report screen generated by the system, not modifi able by the user.

**538** Software Testing: Principles and Practices

**Step**

**7**

**Black-Box Testing on Units/Modules**

**of Income Tax Calculator SRS ver 3.0**

*Black-Box Testing on various modules of Income Tax Calculator SRS ver 3.0 is presented here. The details of the various modules have been reproduced to understand the functionality of the module and later, the black-box test cases of modules have been given. The reader is advised to execute these test cases on the running executable application provided in the CD.*

**Accept Personal Details (APD)**

The function will accept the following details to be entered by the user. The user cannot move to the next entry unless he enters the correct entry.

Name (3 to 15 alphabets with spaces in between)

Date of Birth (dd/mm/yyyy)

Permanent address (3 to 30 characters) The allowed characters are alpha bets, digits,

Spaces, and commas only.

Sex (M/F one alphabet only)

Status: Salaried or not (Y/N one alphabet only)

If the user enters the answer Y (Yes) to the status entry, then the function will display the following three entries, otherwise it will not.

Designation (if salaried) (3 to 15 alphabets)

Name of the employer (if salaried) (3 to 25 alphabets with spaces)

Address of the employer (if salaried) (3 to 30 characters)

PAN number (10 characters consisting alphabets and digits 0–9)

TDS circle where annual return/statement under section 206 is to be fi led (3 to 15 alphabets with spaces)

Period: From (dd/mm/yyyy)

To (dd/mm/yyyy)

Assessment year (yyyy-yy)

This module will be tested with equivalence class partitioning methods.

Black-Box Testing on Units/Modules of Income Tax Calculator SRS ver 3.0 **539**l

Using the information of the module, the following equivalence classes are generated:

C1 = {3 ≤ Name ≤ 15}

C2 = {Name < 3}

C3 = {Name > 15}

C4 = {Name: Any invalid character other than alphabets and spaces be tween the alphabets}

C5 = {Name: Blank}

C6 = {Date of Birth: digits only}

C7 = {Date of Birth: Any invalid character other than digit}

C8 = {Date of Birth: Blank}

C9 = {3 ≤ Permanent address ≤ 30}

C10 = {Permanent address < 3}

C11 = {Permanent address > 30}

C12 = {Permanent address: Any invalid character other than alphabets, dig its, and spaces between them}

C13 = {Permanent address: Blank}

C14 = {Sex: M/F}

C15 = {Sex: any character other than M/F}

C16 = {Sex: Blank}

C17 = {Status: Y/N}

C18 = {Status: any character other than Y/N}

C19 = {Status: Blank}

C20 = {3 ≤ Designation ≤ 15}

C21 = {Designation < 3}

C22 = {Designation > 15}

C23 = {Designation: Any invalid character other than alphabets and spaces between the alphabets}

C24 = {Designation : Blank}

C25 = {3 ≤ Name of Employer ≤ 25}

C26 = {Name of Employer < 3}

C27 = {Name of Employer > 25}

C28 = {Name of Employer: Any invalid character other than alphabets and spaces between the alphabets}

**540** Software Testing: Principles and Practices

C29 = {Name of Employer: Blank}

C30 = {Address of the employer < 3}

C31 = {Address of the employer > 30}

C32 = {Address of the employer: Any invalid character other than alpha bets, digits and spaces between them}

C33 = {Address of the employer: Blank}

C34 = {PAN number: 10 characters consisting of alphabets and digits only} C34 = {PAN number < 10}

C34 = {PAN number > 10}

C34 = {PAN number: 10 characters consisting of any invalid character other than alphabets and digits only}

C35 = {PAN number: Blank}

C36 = {3 ≤ TDS Circle ≤ 15}

C37 = {TDS Circle < 3}

C38 = {TDS Circle > 15}

C39 = {TDS Circle: Any invalid character other than alphabets and spaces between the alphabets}

C40 = {TDS Circle: Blank}

C41 = {Period From: digits only}

C42 = {Period From: Any invalid character other than digit}

C43 = {Period From: Blank}

C44 = {Period To: digits only}

C45 = {Period To: Any invalid character other than digit}

C46 = {Period To: Blank}

After preparing the classes for this function, one test case per class should be designed as given below:

| **Test case ID** | **Class**  **covered** | **Name** | **Date of**  **Birth** | **Permanent Address** | **…** | **Expected Output** |
| --- | --- | --- | --- | --- | --- | --- |
| APD1 | C1 | Harish |  |  |  | Normal behaviour |
| APD2 | C2 | Na |  |  |  | Name too short |
| APD3 | C3 | Abdul Ghaf far Khan |  |  |  | Name too long |
| APD4 | C4 | ;’.,sfh’;.,s |  |  |  | Name should contain only alphabets and spaces. |

APD5 C5 Please enter a valid name entry.

Black-Box Testing on Units/Modules of Income Tax Calculator SRS ver 3.0 **541**l

APD6 C6 Harish 01/07/1984 Normal behaviour

| APD7 | C7 | Harish | Ab/07/qw12 |  |  | Date should contain only digits. |
| --- | --- | --- | --- | --- | --- | --- |
| APD8 | C8 | Harish |  |  |  | Please enter a valid Date entry. |
| …. | … | … | … | … | … | … |

**Accept Income Details (AID)**

The function will enquire whether the user is a salaried person or has some other source of income. If the user is not a salaried person, the system will ask for the source of income. The user may enter various types of source of incomes as given below. The user cannot move to the next entry unless he enters the correct entry.

*Source of Income: (3 to 20 alphabets with spaces)*

*Amount: (positive real numbers with maximum two decimal places)*

These two entries can be tested with equivalence class partitioning methods. The classes are:

C1 = {3 ≤ Source of Income ≤ 20}

C2 = {Source of Income < 3}

C3 = {Source of Income > 20}

C4 = {Source of Income: Blank}

C5 = {Amount: positive real numbers with maximum two decimal places}

C6 = {Amount: negative number or characters other than digit}

C7 = {Amount: Blank }

After preparing the classes, one test case per class should be designed as given below:

| **Test Case ID** | **Class covered** | **Source of Income** | **Amount** | **Expected Output** |
| --- | --- | --- | --- | --- |
| AID1 | C1 | Agriculture |  | Normal behaviour |
| AID2 | C2 | Tc |  | Entry too short |
| AID3 | C3 | Income from consultancy |  | Entry too long |
| AID4 | C4 |  |  | Please enter the source of income |
| AID5 | C5 | Agriculture | 40000.00 | Normal behaviour |
| AID6 | C6 | Agriculture | Rs 23000.00 | Please enter the positive real numbers only |
| AID7 | C7 | Agriculture |  | Please enter amount |

**542** Software Testing: Principles and Practices

The function will aggregate the total amount of income as gross total income. If the person is salaried, the function asks for the following details:

1. Gross Salary

(a) Salary as per the provisions contained in the section 17(1)

(b) Value of the perquisites under section 17(2) (As per form number 12BA, wherever applicable)

(c) Profi ts in lieu of salary under section 17(3) (As per form number 12BA, wherever applicable)

(d) Total (to be calculated by this function)

The entries, a, b, c can also be tested with equivalence classes.

C8 = {a: valid entry}

C9 = {a: invalid entry}

C10 = {a:blank}

C11 = {b: valid entry}

C12 = {b: invalid entry}

C13 = {b:blank}

C14 = {c: valid entry}

C15 = {c: invalid entry}

C16 = {c: blank}

Therefore, the test cases will be as follows.

| **Test Case ID** | **Class covered** | **a** | **b** | **c** | **Expected Output** |
| --- | --- | --- | --- | --- | --- |
| AID8 | C8 | 2000.00 |  |  | Normal Behaviour |
| AID9 | C9 | –1200 |  |  | Please enter a valid entry. |
| AID10 | C10 |  |  |  | Please enter a valid amount. |
| AID11 | C11 | 2000.00 | 1000.00 |  | Normal Behaviour |
| AID12 | C12 | 2000.00 | Rs 200 |  | Please enter a valid entry. |
| AID13 | C13 | 2000.00 |  |  | Please enter a valid amount. |
| AID14 | C14 | 2000.00 | 1000.00 | 3000.00 | Normal Behaviour |
| AID15 | C15 | 2000.00 | 1000.00 | Rs 200 | Please enter a valid entry. |
| AID16 | C16 | 2000.00 | 1000.00 |  | Please enter a valid amount. |

2. Less allowance to the extent exempt under section 10

This function will add the exempted allowances.

3. Balance

This function will calculate the difference of the gross salary and the ex empted allowances.

4. Deductions

Entertainment allowance (EA)

Tax on employment (TE)

Black-Box Testing on Units/Modules of Income Tax Calculator SRS ver 3.0 **543**l

These two entries can be tested with equivalence class partitioning methods. The classes are:

C17 = {EA: valid entry}

C18 = {EA: invalid entry}

C19 = {EA: Blank}

C20 = {TE: valid entry}

C21 = {TE: invalid entry}

C22 = {TE: Blank}

The test cases will be as follows:

| **Test Case ID** | **Class covered** | **EA** | **TE** | **Expected Output** |
| --- | --- | --- | --- | --- |
| AID17 | C17 | 2000.00 |  | Normal behaviour |
| AID18 | C18 | –200 |  | Please enter a valid amount. |
| AID19 | C19 |  |  | Please enter a valid amount. |
| AID20 | C20 | 2000.00 | 100.00 | Normal behaviour |
| AID21 | C21 | 2000.00 | @12.00 | Please enter a valid amount. |
| AID22 | C22 | 2000.00 |  | Please enter a valid amount. |

5. Aggregate

This function will calculate the aggregate of the deductions entered above. 6. Income Chargeable Under the Head ‘Salaries’

The function will calculate the difference of item 3 – item 5.

7. Add

Any other item reported by the employee

User may enter multiple incomes. The function will add all these incomes.

This entry can be tested with equivalence class partitioning methods. The classes are:

C23 = {Income: valid entry}

C24 = {Income: invalid entry}

C25 = {Income: Blank}

The test cases will be:

| **Test Case ID** | **Class covered** | **Income** | **Expected Output** |
| --- | --- | --- | --- |
| AID23 | C23 | 200.00 | Normal behaviour |
| AID24 | C24 | –1000 | Please enter a valid amount. |
| AID25 | C25 |  | Please enter a valid amount. |

8. Gross Total Income

The function will add item 6 and 7.

All the amounts will be positive real numbers with a maximum of two decimal places.

**544** Software Testing: Principles and Practices

**Accept Savings & Donations Details (ASD)**

The function will ask the user to enter the total savings and the donations in the following format.

Saving Type: (3 to 20 alphabets with spaces)

Deductible amount: (positive real numbers with maximum two decimal places)

The user may enter multiple savings. The function will add all the deductible amounts in aggregate deductible amount.

The user cannot move to the next entry unless he enters the correct entry.

These two entries can be tested with equivalence class partitioning methods. The classes are:

C1 = {3 ≤ Saving Type ≤ 20}

C2 = {Saving Type < 3}

C3 = {Saving Type > 20}

C4 = {Saving Type: Blank}

C5 = {Deductible amount: positive real numbers with maximum two deci mal places}

C6 = {Deductible amount: negative number or characters other than digit} C7 = {Deductible amount: Blank }

After preparing the classes, one test case per class should be designed as given below:

| **Test Case**  **ID** | **Class**  **covered** | **Saving**  **Type** | **Deductible**  **Amount** | **Expected Output** |
| --- | --- | --- | --- | --- |
| ASD1 | C1 | LIC |  | Normal behaviour |
| ASD2 | C2 | MF |  | Entry too short |
| ASD3 | C3 | National saving certifi cate |  | Entry too long |
| ASD4 | C4 |  |  | Please enter the saving type |
| ASD5 | C5 | LIC | 2000.00 | Normal behaviour |
| ASD6 | C6 | LIC | Rs 23000.00 | Please enter the positive real numbers only |
| ASD7 | C7 | LIC |  | Please enter amount |

**Accept Tax Deduction Details (ATD)**

If the person is salaried, then this function will accept the details if tax deducted by the employer during the year is in the following format:

Amount of tax deposited (positive real numbers with maximum two decimal places)

Date (dd/mm/yyyy)

Challan Number (5 to 20 characters)

Black-Box Testing on Units/Modules of Income Tax Calculator SRS ver 3.0 **545**l

The above details may be entered multiple times. The function will add the total amounts of tax deposited.

The user cannot move to the next entry unless he enters the correct entry. *Amount of TDS*: (positive real numbers with maximum two decimal places)

Testing of these entries can be done with equivalence class partitioning methods. The classes are:

C1 = {Amount of tax deposited: valid entry}

C2 = {Amount of tax deposited: invalid entry}

C3 = {Amount of tax deposited: Blank}

C4 = {Date : digits only}

C5 = {Date : Any invalid character other than digit}

C6 = {Date : Blank}

C7 = {5 ≤ Challan Number ≤ 20}

C8 = {Challan Number < 5}

C9 = {Challan Number > 20}

C10 = {Challan Number: Blank}

| **Test**  **Case ID** | **Class**  **covered** | **Amount of tax deposited** | **Date** | **Challan**  **Number** | **Expected Output** |
| --- | --- | --- | --- | --- | --- |
| ATD1 | C1 | 2000 |  |  | Normal behaviour |
| ATD2 | C2 | Rs 2000 |  |  | Please enter the valid amount. |
| ATD3 | C3 |  |  |  | Blank Entry. Please enter the valid amount. |
| ATD4 | C4 | 2000 | 12/02/2009 |  | Normal behaviour |
| ATD5 | C5 | 2000 | 12/feb/2009 |  | Please enter a valid date. |
| ATD6 | C6 | 2000 |  |  | Blank entry. Please enter a valid date. |
| ATD7 | C7 | 2000 | 12/02/2009 | SDE345 | Normal behaviour |
| ATD8 | C8 | 2000 | 12/02/2009 | SDE | Challan number too short |
| ATD9 | C9 | 2000 | 12/02/2009 | Sdefrtg5667 89asdf5678 | Challan number too long |
| ATD10 | C10 | 2000 | 12/02/2009 |  | Blank entry. Please enter challan number. |

The testing of date fi eld can be done separately with the help of BVA technique. So, now we will test this fi eld with the following specifi cations:

1 ≤ mm ≤ 12

1≤ dd ≤ 31

2009 ≤ yyyy ≤ 2099

**546** Software Testing: Principles and Practices

**Test Cases Using BVC**

|  | **Month** | **Day** | **Year** |
| --- | --- | --- | --- |
| Min value | 1 | 1 | 2009 |
| Min+ value | 2 | 2 | 2010 |
| Max value | 12 | 31 | 2099 |
| Max– value | 11 | 30 | 2098 |
| Nominal value | 6 | 15 | 2060 |

Using these values, test cases can be designed as shown below:

| **Test Case ID** | **Month** | **Day** | **Year** | **Expected Output** |
| --- | --- | --- | --- | --- |
| 1 | 1 | 15 | 2060 | Normal behaviour |
| 2 | 2 | 15 | 2060 | Normal behaviour |
| 3 | 11 | 15 | 2060 | Normal behaviour |
| 4 | 12 | 15 | 2060 | Normal behaviour |
| 5 | 6 | 1 | 2060 | Normal behaviour |
| 6 | 6 | 2 | 2060 | Normal behaviour |
| 7 | 6 | 30 | 2060 | Normal behaviour |
| 8 | 6 | 31 | 2060 | Invalid input |
| 9 | 6 | 15 | 2009 | Normal behaviour |
| 10 | 6 | 15 | 2010 | Normal behaviour |
| 11 | 6 | 15 | 2098 | Normal behaviour |
| 12 | 6 | 15 | 2099 | Normal behaviour |
| 13 | 6 | 15 | 2060 | Normal behaviour |

**Test Cases Using Robust Testing**

|  | **Month** | **Day** | **Year** |
| --- | --- | --- | --- |
| Min– value | 0 | 0 | 2008 |
| Min value | 1 | 1 | 2009 |
| Min+ value | 2 | 2 | 2010 |
| Max value | 12 | 31 | 2099 |
| Max– value | 11 | 30 | 2098 |
| Max+ value | 13 | 32 | 3000 |
| Nominal value | 6 | 15 | 2060 |

Using these values, test cases can be designed as shown below:

| **Test Case ID** | **Month** | **Day** | **Year** | **Expected Output** |
| --- | --- | --- | --- | --- |
| 1 | 0 | 15 | 2060 | Invalid date |
| 2 | 1 | 15 | 2060 | Normal behaviour |
| 3 | 2 | 15 | 2060 | Normal behaviour |
| 4 | 11 | 15 | 2060 | Normal behaviour |
| 5 | 12 | 15 | 2060 | Normal behaviour |
| 6 | 13 | 15 | 2060 | Invalid date |
| 7 | 6 | 0 | 2060 | Invalid date |

8 6 1 2060 Normal behaviour

Black-Box Testing on Units/Modules of Income Tax Calculator SRS ver 3.0 **547**l

9 6 2 2060 Normal behaviour

| 10 | 6 | 30 | 2060 | Normal behaviour |
| --- | --- | --- | --- | --- |
| 11 | 6 | 31 | 2060 | Invalid input |
| 12 | 6 | 32 | 2060 | Invalid date |
| 13 | 6 | 15 | 2008 | Invalid date |
| 14 | 6 | 15 | 2009 | Normal behaviour |
| 15 | 6 | 15 | 2010 | Normal behaviour |
| 16 | 6 | 15 | 2098 | Normal behaviour |
| 17 | 6 | 15 | 2099 | Normal behaviour |
| 18 | 6 | 15 | 3000 | Invalid date |
| 19 | 6 | 15 | 2060 | Normal behaviour |

**Test Cases Using Worst Case Testing**

|  | **Month** | **Day** | **Year** |
| --- | --- | --- | --- |
| Min value | 1 | 1 | 2009 |
| Min+ value | 2 | 2 | 2010 |
| Max value | 12 | 31 | 2099 |
| Max– value | 11 | 30 | 2098 |
| Nominal value | 6 | 15 | 2060 |

Using these values, test cases can be designed as shown below:

| **Test Case ID** | **Month** | **Day** | **Year** | **Expected Output** |
| --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 2009 | Normal behaviour |
| 2 | 1 | 1 | 2010 | Normal behaviour |
| 3 | 1 | 1 | 2060 | Normal behaviour |
| 4 | 1 | 1 | 2098 | Normal behaviour |
| 5 | 1 | 1 | 2099 | Normal behaviour |
| 6 | 1 | 2 | 2009 | Normal behaviour |
| 7 | 1 | 2 | 2010 | Normal behaviour |
| 8 | 1 | 2 | 2060 | Normal behaviour |
| 9 | 1 | 2 | 2098 | Normal behaviour |
| 10 | 1 | 2 | 2099 | Normal behaviour |
| 11 | 1 | 15 | 2009 | Normal behaviour |
| 12 | 1 | 15 | 2010 | Normal behaviour |
| 13 | 1 | 15 | 2060 | Normal behaviour |
| 14 | 1 | 15 | 2098 | Normal behaviour |
| 15 | 1 | 15 | 2099 | Normal behaviour |
| 16 | 1 | 30 | 2009 | Normal behaviour |
| 17 | 1 | 30 | 2010 | Normal behaviour |
| 18 | 1 | 30 | 2060 | Normal behaviour |
| 19 | 1 | 30 | 2098 | Normal behaviour |
| 20 | 1 | 30 | 2099 | Normal behaviour |
| 21 | 1 | 31 | 2009 | Normal behaviour |

22 1 31 2010 Normal behaviour

**548** Software Testing: Principles and Practices

231 31 2060 Normal behaviour

| 24 | 1 | 31 | 2098 | Normal behaviour |
| --- | --- | --- | --- | --- |
| 25 | 1 | 31 | 2099 | Normal behaviour |
| 26 | 2 | 1 | 2009 | Normal behaviour |
| 27 | 2 | 1 | 2010 | Normal behaviour |
| 28 | 2 | 1 | 2060 | Normal behaviour |
| 29 | 2 | 1 | 2098 | Normal behaviour |
| 30 | 2 | 1 | 2099 | Normal behaviour |
| 31 | 2 | 2 | 2009 | Normal behaviour |
| 32 | 2 | 2 | 2010 | Normal behaviour |
| 33 | 2 | 2 | 2060 | Normal behaviour |
| 34 | 2 | 2 | 2098 | Normal behaviour |
| 35 | 2 | 2 | 2099 | Normal behaviour |
| 36 | 2 | 15 | 2009 | Normal behaviour |
| 37 | 2 | 15 | 2010 | Normal behaviour |
| 38 | 2 | 15 | 2060 | Normal behaviour |
| 39 | 2 | 15 | 2098 | Normal behaviour |
| 40 | 2 | 15 | 2099 | Normal behaviour |
| 41 | 2 | 30 | 2009 | Invalid date |
| 42 | 2 | 30 | 2010 | Invalid date |
| 43 | 2 | 30 | 2060 | Invalid date |
| 44 | 2 | 30 | 2098 | Invalid date |
| 45 | 2 | 30 | 2099 | Invalid date |
| 46 | 2 | 31 | 2009 | Invalid date |
| 47 | 2 | 31 | 2010 | Invalid date |
| 48 | 2 | 31 | 2060 | Invalid date |
| 49 | 2 | 31 | 2098 | Invalid date |
| 50 | 2 | 31 | 2099 | Invalid date |
| 51 | 6 | 1 | 2009 | Normal behaviour |
| 52 | 6 | 1 | 2010 | Normal behaviour |
| 53 | 6 | 1 | 2060 | Normal behaviour |
| 54 | 6 | 1 | 2098 | Normal behaviour |
| 55 | 6 | 1 | 2099 | Normal behaviour |
| 56 | 6 | 2 | 2009 | Normal behaviour |
| 57 | 6 | 2 | 2010 | Normal behaviour |
| 58 | 6 | 2 | 2060 | Normal behaviour |
| 59 | 6 | 2 | 2098 | Normal behaviour |
| 60 | 6 | 2 | 2099 | Normal behaviour |
| 61 | 6 | 15 | 2009 | Normal behaviour |
| 62 | 6 | 15 | 2010 | Normal behaviour |
| 63 | 6 | 15 | 2060 | Normal behaviour |
| 64 | 6 | 15 | 2098 | Normal behaviour |

656 15 2099 Normal behaviour

Black-Box Testing on Units/Modules of Income Tax Calculator SRS ver 3.0 **549**l

666 30 2009 Normal behaviour

| 67 | 6 | 30 | 2010 | Normal behaviour |
| --- | --- | --- | --- | --- |
| 68 | 6 | 30 | 2060 | Normal behaviour |
| 69 | 6 | 30 | 2098 | Normal behaviour |
| 70 | 6 | 30 | 2099 | Normal behaviour |
| 71 | 6 | 31 | 2009 | Invalid date |
| 72 | 6 | 31 | 2010 | Invalid date |
| 73 | 6 | 31 | 2060 | Invalid date |
| 74 | 6 | 31 | 2098 | Invalid date |
| 75 | 6 | 31 | 2099 | Invalid date |
| 76 | 11 | 1 | 2009 | Normal behaviour |
| 77 | 11 | 1 | 2010 | Normal behaviour |
| 78 | 11 | 1 | 2060 | Normal behaviour |
| 79 | 11 | 1 | 2098 | Normal behaviour |
| 80 | 11 | 1 | 2099 | Normal behaviour |
| 81 | 11 | 2 | 2009 | Normal behaviour |
| 82 | 11 | 2 | 2010 | Normal behaviour |
| 83 | 11 | 2 | 2060 | Normal behaviour |
| 84 | 11 | 2 | 2098 | Normal behaviour |
| 85 | 11 | 2 | 2099 | Normal behaviour |
| 86 | 11 | 15 | 2009 | Normal behaviour |
| 87 | 11 | 15 | 2010 | Normal behaviour |
| 88 | 11 | 15 | 2060 | Normal behaviour |
| 89 | 11 | 15 | 2098 | Normal behaviour |
| 90 | 11 | 15 | 2099 | Normal behaviour |
| 91 | 11 | 30 | 2009 | Normal behaviour |
| 92 | 11 | 30 | 2010 | Normal behaviour |
| 93 | 11 | 30 | 2060 | Normal behaviour |
| 94 | 11 | 30 | 2098 | Normal behaviour |
| 95 | 11 | 30 | 2099 | Normal behaviour |
| 96 | 11 | 31 | 2009 | Invalid date |
| 97 | 11 | 31 | 2010 | Invalid date |
| 98 | 11 | 31 | 2060 | Invalid date |
| 99 | 11 | 31 | 2098 | Invalid date |
| 100 | 11 | 31 | 2099 | Invalid date |
| 101 | 12 | 1 | 2009 | Normal behaviour |
| 102 | 12 | 1 | 2010 | Normal behaviour |
| 103 | 12 | 1 | 2060 | Normal behaviour |
| 104 | 12 | 1 | 2098 | Normal behaviour |
| 105 | 12 | 1 | 2099 | Normal behaviour |
| 106 | 12 | 2 | 2009 | Normal behaviour |

107 122 2010 Normal behaviour

**550** Software Testing: Principles and Practices

108 12 2 2060 Normal behaviour

| 109 | 12 | 2 | 2098 | Normal behaviour |
| --- | --- | --- | --- | --- |
| 110 | 12 | 2 | 2099 | Normal behaviour |
| 111 | 12 | 15 | 2009 | Normal behaviour |
| 112 | 12 | 15 | 2010 | Normal behaviour |
| 113 | 12 | 15 | 2060 | Normal behaviour |
| 114 | 12 | 15 | 2098 | Normal behaviour |
| 115 | 12 | 15 | 2099 | Normal behaviour |
| 116 | 12 | 30 | 2009 | Normal behaviour |
| 117 | 12 | 30 | 2010 | Normal behaviour |
| 118 | 12 | 30 | 2060 | Normal behaviour |
| 119 | 12 | 30 | 2098 | Normal behaviour |
| 120 | 12 | 30 | 2099 | Normal behaviour |
| 121 | 12 | 31 | 2009 | Normal behaviour |
| 122 | 12 | 31 | 2010 | Normal behaviour |
| 123 | 12 | 31 | 2060 | Normal behaviour |
| 124 | 12 | 31 | 2098 | Normal behaviour |
| 125 | 12 | 31 | 2099 | Normal behaviour |

**Generate Tax Payable Details**

This function calculates the tax payable by the person in the following format: *Taxable income* The function will calculate this by taking the difference of gross total income in the function Accept Income Details and aggregate deductible amount in the function Accept Savings & Donation Details.

*Tax on taxable income* The function will calculate this using the appropriate slab of user as given above.

The following conditions are there for calculating the appropriate tax slab of a person:

Is Sex Male?

Age > 65?

Income: up to 1.5 lacs

Income: up to 1.8 lacs

Income : 1.5 lacs to 3 lacs

Income : 1.8 lacs to 3 lacs

Income : 3 lacs to 5 lacs

Income : above 5 lacs

Income : up to 2.25 lacs

Income : 2.25 lacs to 3 lacs

Black-Box Testing on Units/Modules of Income Tax Calculator SRS ver 3.0 **551**l

The following outputs will be there:

No tax

10%

20%

30%

The test cases for these conditions and outputs can be designed using a deci sion table, as given below:

|  |  | **R1** | **R2** | **R3** | **R4** | **R5** | **R6** | **R7** | **R8** | **R9** | **R10** | **R11** | **R12** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Condition Stub** | C1: Is Sex Male? | T | T | T | T | F | F | F | F | I | I | I | I |
| C2: Age > 60? | F | F | T | F | F | F | F | F | T | T | T | T |
| C3: Income: up to 1.5 lacs | T | F | F | F | I | F | F | F | I | F | F | F |
| C4: Income: up to 1.8 lacs | F | I | F | F | T | F | F | F | I | F | F | F |
| C5: Income : 1.5 lacs to 3 lacs | F | T | F | F | F | I | F | F | I | I | F | F |
| C6: Income : 1.8 lacs to 3 lacs | F | F | F | F | F | T | F | F | I | I | F | F |
| C7: Income : 3 lacs to 5 lacs | F | F | T | F | F | F | T | F | F | F | T | F |
| C8: Income : above 5 lacs | F | F | F | T | F | F | F | T | F | F | F | T |
| C9: Income : up to 2.25 lacs | F | I | F | F | F | I | F | I | T | F | F | F |
| C10: Income : 2.25 lacs to 3 lacs | F | I | F | F | F | I | F | F | F | T | F | F |
| **Action Stub** | A1: No Tax | X |  |  |  | X |  |  |  | X |  |  |  |
| A2: 10% |  | X |  |  |  | X |  |  |  | X |  |  |
| A3: 20% |  |  | X |  |  |  | X |  |  |  | X |  |
| A4: 30% |  |  |  | X |  |  |  | X |  |  |  | X |

*Surcharge* The function will calculate the surcharge as: If tax on taxable income exceeds Rs 10 lacs, a 10% surcharge is charged.

*Education cess* The function will calculate the education cess as a 3% of tax on taxable income.

*Tax payable* The function will sum up the tax on taxable income, surcharge, and education cess.

*Relief under section 89* User will enter the amount, if applicable.

*Tax payable after relief (if applicable)* The function will deduct relief amount from tax payable.

*Total Tax deducted* Displayed from the function Accept Tax Deduction Details.

*Tax payable/refundable* The function will fi nd the difference of tax payable and the total tax deducted. If difference is positive, then this amount is the net tax to be paid by the person, otherwise the amount is due on the government to be refunded.

**552** Software Testing: Principles and Practices

**Step**

**8**

**White-Box Testing on Units/Modules**

**of Income Tax Calculator**

*Here, we will discuss how the modules of Income Tax Calculator is tested us ing white-box testing techniques. The modules of TaxCalculator.h (refer CD) have been reproduced here for understanding their functionality and coding details. The reader is advised to refer to the full code for implementation details of the applica tion and execute these test cases on the running executable application provided in the CD.*

**BASIS PATH TESTING ON income\_details\_non\_sal()**

**Source Code of income\_details\_non\_sal()**

fl oat income\_details\_non\_sal()

{

1’ char source[20]=“abc”;

2’ fl oat amount,total =0;

3’ int fl ag1=1,fl ag2=1,i;

4’ char income\_ch=‘y’;

1 while((income\_ch==‘y’)||( income\_ch==‘Y’))

2 {

3 while(fl ag1==1)

4 {

5 printf(“\nEnter SOURCE\t:”);

6 gets(source);

7 for(i=0;i<strlen(source);i++)

8 {

9 if(((toascii(source[i]) >= 65) && (toascii(source[i]) <= 122)) || (toascii(source[i]) == 32))

10 {

11 fl ag1=0;

12 }

White-Box Testing on Units/Modules of Income Tax Calculator **553**l

13 else

14 {

15 printf(“\nSource can contain only charcter Error at position number %d”,i);

16 fl ag1=1;

17 break;

18 }

19 }//end for

20 if((strlen(source)<3)||(strlen(source)>20))

21 {

22 printf(“\nSource can contain a max of 20 characters”); 23 fl ag1=1;

24 }

25 }

26 while(fl ag2==1)

27 {

28 printf(“\nEnter Amount\t:”);

29 scanf(“%f”,&amount);

30 if(amount>0)

31 {

32 fl ag2=0;

33 }

34 else

35 {

36 printf(“\nAmount cannot be less than or equal to 0”);

37 fl ag2=1;

38 }

39 }

40 printf(“\n\nPress any key to proceed”);

41 getch();

42 clrscr();

43 patt(“INCOME Details”);

44 printf(“\nSOURCE\t:%s”,source);

45 printf(“\nAMOUNT\t:%f”,amount);

46 total=total+amount;

47 printf(“\nDo you want to enter more(y/n)\t:”);

48 income\_ch=getche();

49 fl ag1=1;

50 fl ag2=1;

51 }

52 printf(“\nTotal\t\t:%f”,total);

53 return(total);

54 }

**554** Software Testing: Principles and Practices

**DD Graph for Module income\_details\_non\_sal()** ,– ,

N1N2

1 4

1

R6

R7

R4

N3

N4

N5

N6

N8

N11

N12

N13

N14

N16

N19

2

3

4,5,6

7

8

9

10,11,12 13 17–

19

20

21 24–

25

26

27 29–

30

31 33–

34 38–

N7

N9

N10

R3

N15

N17

N18

52 54–

R1

R2

N23

N20N21

39

40 50– 51

R5

N22

White-Box Testing on Units/Modules of Income Tax Calculator **555**l

**Cyclomatic Complexity of income\_details\_non\_sal()**

1. V(G) = *e* – *n* + 2*P*

= 29 – 23 +2

= 8

2. V(G) = Number of predicate nodes + 1

= 7 + 1

= 8

3. V(G) = No. of regions

= 8

**Independent Paths of income\_details\_non\_sal()**

Since the cyclomatic complexity of the graph is 8, there will be 8 independent paths in the graph, as shown below:

1. N1N2N23

2. N1N2 N3 N4 N15 N21 N22 N2 N23

3. N1N2 N3 N4 N5 N6 N12 N14 N4 N15 N21 N22 N2 N23

4. N1N2 N3 N4 N5 N6 N7 N8 N10 N12 N14 N4 N15 N21 N22 N2 N23

5. N1N2 N3 N4 N5 N6 N7 N8 N9 N11 N6 N12 N14 N4 N15 N21 N22 N2 N23 6. N1N2 N3 N4 N5 N6 N7 N8 N9 N11 N6 N12 N13 N14 N4 N15 N21 N22 N2 N23 7. N1N2 N3 N4 N15 N16 N17 N19 N20 N15 N21 N22 N2 N23

8. N1N2 N3 N4 N15 N16 N17 N18 N20 N15 N21 N22 N2 N23

**Test Case Design on income\_details\_non\_sal() from the list of**

**Independent Paths**

| **Test Case ID** | **Inputs** | | **Expected Output** | **Independent paths**  **covered by Test Case** |
| --- | --- | --- | --- | --- |
| **Source** | **Amount** |
| 1 | Agriculture | 400000 | Source Agriculture  Amount 400000  Do you want to enter more(y/n) Y | 1, 2, 3, 5, 8 |
|  | Others | 100000 | Source Others  Amount 100000  Do you want to enter more(y/n) N |
|  |  |  | Total 500000 |  |
| 2 | 1234 |  | Source can contain only character. | 1, 2, 3, 4 |
| 3 | Agriculture and others |  | Source can contain a max of 20 charac ters. | 1, 2, 3, 6 |
| 4 | Agriculture | 0 | Amount cannot be less than or equal to 0 | 1, 2, 3, 5, 7 |

**556** Software Testing: Principles and Practices

**DATA FLOW TESTING ON income\_details\_non\_sal()**

**Defi nition Nodes and Usage Nodes**

| **Variable** | **Defi ned At** | **Used At** |
| --- | --- | --- |
| source  amount  total  fl ag1  fl ag2  i  income\_ch | 1’, 6  29  2’, 46  3’, 11, 16, 23, 49  3’, 32, 37, 50  7  4’, 48 | 7, 9, 20, 44  30, 45, 46  46, 52  3  26  7, 9, 15  1 |

**du and dc paths**

| **Variable** | **du Path(beg-end)** | **dc?** |
| --- | --- | --- |
| source | 1’–7 | No |
| 1’–9 | No |
| 1’–20 | No |
| 1’–44 | No |
| 6–7 | Yes |
| 6–9 | Yes |
| 6–20 | Yes |
| 6–44 | Yes |
| amount | 29–30 | Yes |
| 29–45 | Yes |
| 29–46 | Yes |
| total | 2’–46 | Yes |
| 2’–52 | No |
| 46–46 | Yes |
| 46–52 | Yes |
| fl ag1 | 3’–3 | Yes |
| 11–3 | No |
| 16-3 | No |
| 23–3 | Yes |
| 49–3 | Yes |

White-Box Testing on Units/Modules of Income Tax Calculator **557**l

| fl ag2 | 3’–26 | Yes |
| --- | --- | --- |
| 32–26 | Yes |
| 37–26 | Yes |
| 50–26 | Yes |
| i | 7–7 | Yes |
| 7–9 | Yes |
| 7–15 | Yes |
| income\_ch | 4’–1 | Yes |
| 48–1 | Yes |

**BASIS PATH TESTING ON income\_details\_sal()**

**Source code of income\_details\_sal()**

double income\_details\_sal()

{

1’ fl oat t\_d, d1, d2, sal1, sal2, sal3, t\_sal, sal\_all, sal\_all\_tot=0, ei, t\_ei=0, net\_t\_sal=0, bal;

2’ char sal\_ch=‘y’,ei\_ch=‘y’;

3’ int f1=1,f2=1,f3=1,f4=1;

4’ double gross;

1 while(f2==1)

2 {

3 printf(“\n1.\tGROSS SALARY\t:”);

4 printf(“\n\ta) Salary as per the provisions contained in the sec tion 17(1)\t:”);

5 scanf(“%f”,&sal1);

6 printf(“\n\tb) Value of the perquisites under section 17(2)\n(As per form number 12BA , wherever applicable )\t:”);

7 scanf(“%f”,&sal2);

8 printf(“\n\tc) Profi ts in lieu of salary under section 17(3)\n(As per form number 12BA , wherever applicable )\t:”);

9 scanf(“%f”,&sal3);

10 if((sal1<0)||(sal2<0)||(sal3<0))

11 {

12 f2=1;

13 }

14 else

15 {

16 f2=0;

**558** Software Testing: Principles and Practices

17 }

18 }

19 t\_sal=sal1+sal2+sal3;

20 printf(“\n\td)\tTotal\n\t\t\t\t:%f”,t\_sal);

21 sal\_all\_tot=0;

22 while((sal\_ch==’y’)||(sal\_ch==’Y’))

23 {

24 while(f1==1)

25 {

26 printf(“\n2.\tAllowance to the extent exempt under section 10\t:”);

27 scanf(“%f”,&sal\_all);

28 if(sal\_all<0)

29 {

30 printf(“\nEnter correct value”);

31 }

32 else

33 {

34 f1=0;

35 }

36 }

37 sal\_all\_tot=sal\_all\_tot+sal\_all;

38 printf(“\nEnter more?(Y/N)\t:”);

39 sal\_ch=getche();

40 if((sal\_ch==’y’)||(sal\_ch==’Y’)||(sal\_ch==’n’)||(sal\_ch==’N’)) 41 {

42 f1=0;

43 }

44 else

45 {

46 printf(“\nPlease enter y or n”);

47 }

48 }

49 printf(“\nTotal allowance\t\t:%f”,sal\_all\_tot);

50 bal=t\_sal-sal\_all\_tot;

51 printf(“\nBalance\t:%f”,bal);

52 while(f3==1)

53 {

54 printf(“\n3.\tDeductions\t:”);

55 printf(“\n\tEntertainment allowance(EA)\t:”);

56 scanf(“%f”,&d1);

White-Box Testing on Units/Modules of Income Tax Calculator **559**l

57 printf(“\tTax on employment (TE)\t:”);

58 scanf(“%f”,&d2);

59 if((d1<0)||(d2<0))

60 {

61 f3=1;

62 }

63 else

64 {

65 f3=0;

66 }

67 }

68 t\_d=d1+d2;

69 printf(“\nTotal deductions\t:%f”,t\_d);

70 net\_t\_sal=bal-t\_d;

71 printf(“\n4.\tINCOME CHARGABLE UNDER THE HEAD SALARIES’\t:%f”,net\_t\_sal); 72 while((ei\_ch==’y’)||(ei\_ch==’Y’))

73 {

74 while(f4==1)

75 {

76 printf(“\n5.\tAny other income reported by the Employee\t:”); 77 printf(“\n\t\tEnter Income\t:”);

78 scanf(“%f”,&ei);

79 if(ei<0)

80 {

81 f4=1;

82 }

83 else

84 {

85 f4=0;

86 }

87 }

88 t\_ei=t\_ei+ei;

89 printf(“\n\t\tEnter more?(Y/N)\t:”);

90 ei\_ch=getche();

91 }

92 gross=net\_t\_sal+t\_ei;

93 printf(“\n6.Gross Total Income:\t%f”,gross);

94 return(gross);

95 }

**560** Software Testing: Principles and Practices

**DD Graph for Module income\_details\_sal()**

N1

1 –4 ′

R1

1

2 9–

10

11 13– 14 17–

N2

N3

N4

N5

N6

N7

N8

R9

18

19

N10

20 21– 22

N9 N23 49

R2

23 24

N11

N12

N24 N25

50 51– 52

68 71–

N26

25 27–

N13

N27N32 53 58–

72

N14

28

R3 R6 59

N28

73

N33

29 31–

N15

N29

60 62–

N34

74

N16

R4

32 35–

R10

63 66–

N35

75 78–

N17

36

N19

N30N36 67

79

R8

37 39– 40

N31

80 82–

N37

N18

R5

N38

R11

N20 N21

41 43– 44 47– 48

N22

N39

83 86– 87

88 91– 92

93 95–

R7

N40 N41

N42

White-Box Testing on Units/Modules of Income Tax Calculator **561**l

**Cyclomatic Complexity of income\_details\_sal()**

1. V(G) = *e* – *n* + 2*P*

= 52 – 42 +2

= 12

2. V(G) = Number of predicate nodes + 1

= 11+ 1

= 12

3. V(G) = No. of Regions

= 12

**Independent Paths of income\_details\_sal()**

Since the cyclomatic complexity of the graph is 12, there will be 12 indepen dent paths in the graph as shown below:

1. N1 N2 N8 N9 N10 N23 N24 N25 N26 N32 N41 N42

2. N1 N2 N3 N4 N6 N7 N2 N8 N9 N10 N23 N24 N25 N26 N32 N41 N42

3. N1 N2 N3 N4 N5 N7 N2 N8 N9 N10 N23 N24 N25 N26 N32 N41 N42

4. N1 N2 N8 N9 N10 N11 N12 N18 N19 N22 N21 N10 N23 N24 N25 N26 N32 N41 N42

5. N1 N2 N8 N9 N10 N11 N12 N13 N14 N16 N17 N12 N18 N19 N22 N21 N10 N23 N24 N25 N26 N32 N41 N42

6. N1 N2 N8 N9 N10 N11 N12 N13 N14 N15 N17 N12 N18 N19 N22 N21 N10 N23 N24 N25 N26 N32 N41 N42

7. N1 N2 N8 N9 N10 N11 N12 N18 N19 N20 N21 N10 N23 N24 N25 N26 N32 N41 N42

8. N1 N2 N8 N9 N10 N23 N24 N25 N27 N28 N30 N31 N25 N26 N32 N41 N42 9. N1 N2 N8 N9 N10 N23 N24 N25 N27 N28 N29 N31 N25 N26 N32 N41 N42

10. N1 N2 N8 N9 N10 N23 N24 N25 N27 N28 N30 N31 N25 N26 N32 N33 N34 N40 N32 N41 N42

11. N1 N2 N8 N9 N10 N23 N24 N25 N27 N28 N30 N31 N25 N26 N32 N33 N34 N35 N36 N38 N39 N34 N40 N32 N41 N42

12. N1 N2 N8 N9 N10 N23 N24 N25 N27 N28 N30 N31 N25 N26 N32 N33 N34 N35 N36 N37 N39 N34 N40 N32 N41 N42

**562** Software Testing: Principles and Practices

**Test Case Design on income\_details\_sal() from the list of Independent Paths**

| **Test**  **Case**  **ID** | **Inputs** | | | | | | | | **Expected Output** | **Independent path covered by Test Case** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Sal1** | **Sal2** | **Sal3** | **Sal\_all** | **Enter**  **more?** | **EA** | **TE** | **Other**  **Income** |
| 1 | 2000 | 4000 | 9000 |  |  |  |  |  | Total 15000 | 1, 2, 5, 7, 8, 10, 11 |
|  |  |  | 2000 | Y |  |  |  |  |
|  |  |  | 1000 | n |  |  |  | Total Allowance 3000  Balance 12000 |
|  |  |  |  |  | 200 | 300 |  | Total deductions 500  Income under Head Salaries 11500 |
|  |  |  |  |  |  |  | 12000 | Enter more? Y |
|  |  |  |  |  |  |  | 12000 | Enter more? N  Gross Total Income:  35500 |
| 2 | 2000 | –300 | 500 |  |  |  |  |  |  | 2, 3 |
| 3 | 2000 | 4000 | 9000 |  |  |  |  |  | Total 15000 | 2, 6 |
|  |  |  | –300 |  |  |  |  | Enter correct value |
| 4 | 2000 | 4000 | 9000 |  |  |  |  |  | Total 15000 | 2, 4, 5, 7 |
|  |  |  | 1000 | t |  |  |  | Please enter y or n |
| 5 | 2000 | 4000 | 9000 |  |  |  |  |  | Total 15000 | 2, 5, 7, 9 |
|  |  |  | 1000 | n |  |  |  | Total Allowance 1000  Balance 14000 |
|  |  |  |  |  | –100 | 200 |  |  |
| 6 | 2000 | 4000 | 9000 |  |  |  |  |  | Total 15000 | 2, 5, 7, 8, 10, 12 |
|  |  |  | 2000 | Y |  |  |  |  |
|  |  |  | 1000 | n |  |  |  | Total Allowance 3000  Balance 12000 |
|  |  |  |  |  | 200 | 300 |  | Total deductions 500  Income under Head Salaries 11500 |
|  |  |  |  |  |  |  | –1000 |  |

White-Box Testing on Units/Modules of Income Tax Calculator **563**l

**DATA FLOW TESTING ON income\_details\_sal()**

**Defi nition nodes and Usage nodes**

| **Variable** | **Defi ned At** | **Used At** |
| --- | --- | --- |
| t\_d | 68 | 69, 70 |
| d1 | 56 | 59, 68 |
| d2 | 58 | 59, 68 |
| sal1 | 5 | 10, 19 |
| sal2 | 7 | 10, 19 |
| sal3 | 9 | 10, 19 |
| t\_sal | 19 | 20 |
| sal\_all | 27 | 28, 37 |
| sal\_all\_tot | 21, 37 | 37, 49, 50 |
| ei | 78 | 79, 88 |
| t\_ei | 1’, 88 | 88, 92 |
| net\_t\_sal | 1’, 70 | 71, 92 |
| bal | 50 | 70 |
| sal\_ch | 2’, 39 | 40 |
| ei\_ch | 2’, 90 | 72 |
| f1 | 3’, 34, 42 | 24 |
| f2 | 3’, 12, 16 | 1 |
| f3 | 3’, 61, 65 | 52 |
| f4 | 3’, 81, 85 | 74 |
| gross | 92 | 93 |

**du and dc paths**

| **Variable** | **du Path(beg-end)** | **dc?** |
| --- | --- | --- |
| t\_d | 68–69 | Yes |
| 68–70 | Yes |
| d1 | 56–59 | Yes |
| 56–68 | Yes |
| d2 | 58–59 | Yes |
| 58–68 | Yes |
| sal1 | 5–10 | Yes |
| 5–19 | Yes |
| sal2 | 7–10 | Yes |
| 7–19 | Yes |
| sal3 | 9–10 | Yes |
| 9–19 | Yes |
| t\_sal | 19–20 | Yes |
| sal\_all | 27–28 | Yes |
| 27–37 | Yes |

sal\_all\_tot 21–37 No

**564** Software Testing: Principles and Practices

21–49 No

|  | 21–50 | No |
| --- | --- | --- |
|  | 37–37 | No |
|  | 37–49 | Yes |
|  | 37–50 | Yes |
| ei | 78–79 | Yes |
| 78–88 | Yes |
| t\_ei | 1’–88 | No |
| 1’–92 | No |
| 88–88 | No |
| 88–92 | Yes |
| net\_t\_sal | 1’–71 | No |
| 1’–92 | No |
| 70–71 | Yes |
| 70–92 | Yes |
| bal | 50–70 | Yes |
| sal\_ch | 2’–40 | No |
| 39–40 | Yes |
| ei\_ch | 2’–72 | Yes |
| 90–72 | Yes |
| f1 | 3’–24 | Yes |
| 34–24 | Yes |
| 42–24 | Yes |
| f2 | 3’–1 | Yes |
| 12–1 | Yes |
| 16–1 | Yes |
| f3 | 3’–52 | Yes |
| 61–52 | Yes |
| 65–52 | Yes |
| f4 | 3’–74 | Yes |
| 81–74 | Yes |
| gross | 85–74 | Yes |
| 92–93 | Yes |

**BASIS PATH TESTING ON MODULE savings()**

**Source Code of module savings()**

fl oat savings()

{

1 char saving\_type[20];

2 fl oat amount,total=0;

3 int fl ag1=1,fl ag2=1,i;

4 char sav\_ch=‘y’;

5 while((sav\_ch==‘y’)||(sav\_ch==‘Y’))

6 {

White-Box Testing on Units/Modules of Income Tax Calculator **565**l

7 while(fl ag1==1)

8 {

9 printf(“\nEnter Saving type\t:”);

10 gets(saving\_type);

11 for(i=0;i<strlen(saving\_type);i++)

12 {

13 if(((toascii(saving\_type[i])>= 65) && (toascii(saving\_type[i]) <= 122)) || (toascii(saving\_type[i]) == 32))

14 {

15 fl ag1=0;

16 }

17 else

18 {

19 printf(“\nSaving type can contain only charcter Error at posi tion number %d”,i);

20 fl ag1=1;

21 break;

22 }

23 }

24 if((strlen(saving\_type)<3)||(strlen(saving\_type)>20))

25 {

26 printf(“\nPlease enter between 3 to 20 characters ”);

27 fl ag1=1;

28 }

29 }

30 while(fl ag2==1)

31 {

32 printf(“\nEnter Amount\t:”);

33 scanf(“%f”,&amount);

34 if(amount>0)

35 {

36 fl ag2=0;

37 }

38 else

39 {

40 printf(“\nAmount cannot be less than or equal to 0”);

41 fl ag2=1;

42 }

43 }

44 printf(“\n\nPress any key to proceed”);

45 getch();

46 clrscr();

47 patt(“SAVING Details”);

48 printf(“\nSAVING TYPE\t:%s”,saving\_type);

49 printf(“\nAMOUNT\t\t\t:%f”,amount);

50 total=total+amount;

51 printf(“\nDo you want to enter more(y for yes)\t:”);

52 sav\_ch=getche();

53 fl ag1=1;

**566** Software Testing: Principles and Practices

54 fl ag2=1;

55 }

56 printf(“\nTotal\t\t:%f”,total);

57 return(total);

58 }

**DD Graph for Module savings()**

N2

1–4 5

6

N1

N3

56 57– N23

N4

N5

7

8 10 –

R2

R3

N8

11 12 13

N6 N7

14 16– 17 21–

N9

N10

R7

N12

23 24

N11

R4

N13

N14

R1

25 28–

29

N15

30

R5

31 33– 34

35 37– 38 42–

N16

N17

N18

N19

N20

43

R6

44 54– 55

N21

N22

White-Box Testing on Units/Modules of Income Tax Calculator **567**l

**Cyclomatic Complexity of savings()**

1. V(G) = *e* – *n* + 2*P*

= 29 – 23 +2

= 8

2. V(G) = Number of predicate nodes + 1

= 7 + 1

= 8

3. V(G) = No. of regions

= 8

**Independent Paths of savings()**

Since the cyclomatic complexity of the graph is 8, there will be 8 independent paths in the graph, as shown below:

1. N1 N2 N23

2. N1 N2 N3 N4 N15 N21 N22 N2 N23

3. N1 N2 N3 N4 N5 N6 N12 N14 N4 N15 N21 N22 N2 N23

4. N1 N2 N3 N4 N5 N6 N7 N8 N10 N12 N14 N4 N15 N21 N22 N2 N23

5. N1 N2 N3 N4 N5 N6 N7 N8 N9 N11 N6 N12 N14 N4 N15 N21 N22 N2 N23 6. N1 N2 N3 N4 N5 N6 N12 N13 N14 N4 N15 N21 N22 N2 N23

7. N1 N2 N3 N4 N15 N16 N17 N19 N20 N15 N21 N22 N2 N23

8. N1 N2 N3 N4 N15 N16 N17 N18 N20 N15 N21 N22 N2 N23

**Test Case Design on savings() from the list of Independent Paths**

| **Test**  **Case ID** | **Inputs** | | | **Expected Output** | **Independent path**  **covered by Test Case** |
| --- | --- | --- | --- | --- | --- |
| **Saving Type** | **Amount** | **Enter more?** |  |  |
| 1 | NSC | 5000 |  | Saving type NSC  Amount 5000 | 1, 2, 3, 5, 8 |
|  |  | y |  |
| PPF | 1200 |  | Saving type PPF  Amount 1200 |
|  |  | n | Total 6200 |

2 123 Saving type can contain only character

2, 3, 4

**568** Software Testing: Principles and Practices

3 PF Please enter between 3 to

20 characters

2, 3, 6

| 4 | PPF | 0 |  | Amount cannot be less than or equal to 0. | 2, 3, 7 |
| --- | --- | --- | --- | --- | --- |

**DATA FLOW TESTING ON MODULE SAVINGS()**

**Defi nition and Usage nodes**

| **Variable** | **Defi ned At** | **Used At** |
| --- | --- | --- |
| saving\_type | 10 | 11, 13, 24, 48 |
| amount | 33 | 34, 49, 50 |
| total | 2, 50 | 50, 56, 57 |
| fl ag1 | 3, 15, 20, 27, 53 | 7 |
| fl ag2 | 3, 36, 41, 54 | 30 |
| i | 11 | 11, 13, 19 |
| sav\_ch | 4, 52 | 5 |

**du and dc paths**

| **Variable** | **du Path(beg-end)** | **dc?** |
| --- | --- | --- |
| saving\_type | 10–11 | Yes |
| 10–13 | Yes |
| 10–24 | Yes |
| 10–48 | Yes |
| amount | 33–34 | Yes |
| 33–49 | Yes |
| 33–50 | Yes |
| total | 2–50 | No |
| 50–50 | No |
| 50–56 | Yes |
| 50–57 | Yes |
| fl ag1 | 3–7 | Yes |
| 15–7 | No |
| 20–7 | No |
| 27–7 | Yes |
| 53–7 | Yes |
|  | 3–30 | Yes |
| fl ag2 | 36–30 | Yes |
|  | 41–30 | Yes |

54–30 Yes

White-Box Testing on Units/Modules of Income Tax Calculator **569**l

11–11 No

| i | 11–13 | Yes |
| --- | --- | --- |
|  | 11–19 | Yes |
| sav\_ch | 4–5 | Yes |
| 52–5 | Yes |

**BASIS PATH TESTING ON MODULE tax\_ded()**

**Source code of tax\_ded()**

fl oat tax\_ded()

{

1 char challan[20];

2 int fl ag=1,fl ag1=1,fl ag2=1,fl ag3=1,fl ag4=1,i,dd,mm,yyyy,choice=‘y’;

3 fl oat amount\_tax,tot\_amount\_tax=0,amount\_tds,tot;

4 while((choice==‘y’)||(choice==‘Y’))

5 { fl ag=1,fl ag1=1,fl ag2=1,fl ag3=1,fl ag4=1;

6 while(fl ag4==1)

7 {

8 printf(“\nEnter amount of Tax deposited\t:”);

9 scanf(“%f”,&amount\_tax);

10 if(amount\_tax<0)

11 {

12 printf(“\nAMOUNT CANNOT BE NEGATIVE”);

13 fl ag4=1;

14 }

15 else

16 {

17 fl ag4=0;

18 }

19 }

20 while(fl ag3==1)

21 {

22 fl ag3=0;

23 printf(“\nEnter DATE:\t:”);

24 printf(“\n\tEnter day(dd)\t:”);

25 scanf(“%d”,&dd);

26 if((dd>31)||(dd<=0))

27 {

28 fl ag3=1;

29 printf(“\nWrong date”);

30 }

**570** Software Testing: Principles and Practices

31 printf(“\n\tEnter month\t:”);

32 scanf(“%d”,&mm);

33 if((mm<=0)||(mm>12)||((mm==2)&&(dd>29)))

34 {

35 printf(“\nWrong date/month”);

36 fl ag3=1;

37 }

38 printf(“\n\tEnter year\t:”);

39 scanf(“%d”,&yyyy);

40 if((yyyy<2009)||(yyyy>2099))

41 {

42 printf(“\nPlease enter a year between 2009 and 2099”);

43 fl ag3=1;

44 }

45 }

46 while(fl ag1==1)

47 {

48 printf(“\nEnter Challan Number\t:”);

49 gets(challan);

50 for(i=0;i<strlen(challan);i++)

51 {

52 if(((toascii(challan[i]) >= 65) && (toascii(challan[i]) <= 122)) || (toascii(challan[i]) == 32) || ((toascii(challan[i])

>= 48) && (toascii(challan[i]) <= 57)))

53 {

54 fl ag1=0;

55 }

56 else

57 {

58 printf(“\nChallan Number can contain only charcter Error at position number %d”,i);

59 fl ag1=1;

60 break;

61 }

62 }//end of for

63 if((strlen(challan)<5)||(strlen(challan)>20))

64 {

65 printf(“\nPlease enter correct Challan Number having 5 to 20 characters”);

66 fl ag1=1;

67 }

68 }

69 printf(“\nAMOUNT OF TAX DEPOSITED\t:%f”,amount\_tax);

White-Box Testing on Units/Modules of Income Tax Calculator **571**l

70 printf(“\nCHALLAN NUMBER\t\t:%s”,challan);

71 printf(“\nDATE\t\t\t:%d/%d/%d”,dd,mm,yyyy);

72 fl ag=1;

73 while(fl ag)

74 {

75 printf(“\nDo you want to enter more ?(y/n)”);

76 choice=getche();

77 if((choice==‘y’)||(choice==‘Y’)||(choice==‘N’)||(choice==‘n’)) 78 {

79 fl ag=0;

80 }

81 else

82 {

83 printf(“\nPlease Enter y or n”);

84 fl ag=1;

85 }

86 }//end of inner while

87 tot\_amount\_tax=tot\_amount\_tax+amount\_tax;

88 }

89 while(fl ag2==1)

90 {

91 printf(“\nEnter amount of TDS \t:”);

92 scanf(“%f”,&amount\_tds);

93 if(amount\_tds<0)

94 {

95 printf(“\nTDS AMOUNT CANNOT BE NEGATIVE”);

96 fl ag2=1;

97 }

98 else

99 {

100 fl ag2=0;

101 }

102 }

103 printf(“\n\nPress any key to proceed”);

104 getch();

105 clrscr();

106 patt(“TAX DEDUCTION”);

107 printf(“\nAMOUNT OF TAX DEPOSITED\t:%f”,tot\_amount\_tax);

108 printf(“\nAMOUNT OF TDS\t:%f”,amount\_tds);

109 tot=tot\_amount\_tax+amount\_tds;

110 printf(“\nTotal\t\t:%f”,tot);

111 return(tot);

112 }

**572** Software Testing: Principles and Practices **DD Graph for Module tax\_ded()**

N2

N4 N5

1–3 4

5

6

7–9

N1 N3

20

21–25

26

N13

27–30

N10 N11

N12

46

47–49 50

51

N21 N22 N23 N24

N6

R1

R13

N14

N25

N7

10

11–14

R3

N15

31–30 33

R7

52

N26

53–55

N8

N16

N27

R8

R2

15–18 N9

19

R4

N18

N19

34–37

N17

38–39

40

41–44

R6

56–61 62

63

64–67

N28

N29

N30

R14

45

N20

N31

68

R15

N39

N40

R12

87

88

R5

69–72 73

74–76

N32 N33 N34

89

90–92

N41

N42

77

78–80

N35

N36

R10

N37

R9

93

94–97

N43

N44

N38

81–85 86

N45

N46

N47

98–101

R11

102

103–111

White-Box Testing on Units/Modules of Income Tax Calculator **573**l

**Cyclomatic Complexity of tax\_ded()**

1. V(G) = *e* – *n* + 2*P*

= 61 – 47 +2

= 16

2. V(G) = Number of predicate nodes + 1

= 15 + 1

= 16

3. V(G) = No. of regions

= 16

**Independent Paths of tax\_ded()**

Since the cyclomatic complexity of the graph is 16, there will be 16 independent paths in the graph as shown below:

1. N1 N2 N41 N47

2. N1 N2 N3 N4 N10 N21 N32 N33 N39 N40 N2 N41 N47

3. N1 N2 N3 N4 N5 N6 N8 N9 N4 N10 N21 N32 N33 N39 N40 N2 N41 N47 4. N1 N2 N3 N4 N5 N6 N7 N9 N4 N10 N21 N32 N33 N39 N40 N2 N41 N47 5. N1 N2 N3 N4 N10 N11 N12 N14 N15 N17 N18 N20 N10 N21 N32 N33 N39 N40 N2 N41 N47

6. N1 N2 N3 N4 N10 N11 N12 N13 N14 N15 N17 N18 N20 N10 N21 N32 N33 N39 N40 N2 N41 N47

7. N1 N2 N3 N4 N10 N11 N12 N13 N14 N15 N16 N17 N18 N20 N10 N21 N32 N33 N39 N40 N2 N41 N47

8. N1 N2 N3 N4 N10 N11 N12 N13 N14 N15 N16 N17 N18 N19 N20 N10 N21 N32 N33 N39 N40 N2 N41 N47

9. N1 N2 N3 N4 N10 N21 N22 N23 N29 N31 N21 N32 N33 N39 N40 N2 N41 N47 10. N1 N2 N3 N4 N10 N21 N22 N23 N29 N30 N31 N21 N32 N33 N39 N40 N2 N41 N47

11. N1 N2 N3 N4 N10 N21 N22 N23 N24 N25 N27 N28 N23 N29 N31 N21 N32 N33 N39 N40 N2 N41 N47

12. N1 N2 N3 N4 N10 N21 N22 N23 N24 N25 N26 N28 N23 N29 N31 N21 N32 N33 N39 N40 N2 N41 N47

13. N1 N2 N3 N4 N10 N21 N32 N33 N34 N35 N37 N38 N33 N39 N40 N2 N41 N47 14. N1 N2 N3 N4 N10 N21 N32 N33 N34 N35 N36 N38 N33 N34 N35 N37 N38 N33 N39 N40 N2 N41 N47

15. N1 N2 N3 N4 N10 N21 N32 N33 N39 N40 N2 N41 N42 N43 N45 N46 N41 N47 16. N1 N2 N3 N4 N10 N21 N32 N33 N39 N40 N2 N41 N42 N43 N44 N46 N41 N47

**574** Software Testing: Principles and Practices

**Test Case Design on tax\_ded() from the list of Independent Paths**

| **Test**  **Case ID** | **Inputs** | | | | | | | **Expected**  **Output** | **Independent path covered by Test Case** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Tax**  **amount** | **day** | **month** | **year** | **Challan number** | **Enter**  **more?** | **TDS** |
| 1 | 5000 | 12 | 12 | 2009 | Dfert56 |  |  | AMOUNT OF TAX DEPOS ITED 5000  CHALLAN  NUMBER  Dfert56  DATE 12 12  2009 | 1, 2, 3, 5, 9,  12, 14, 15 |
|  |  |  |  |  | Y |  |  |
| 2000 | 13 | 12 | 2009 | Hgyt65 |  |  | AMOUNT OF TAX DEPOS ITED 2000  CHALLAN  NUMBER  Hgyt65  DATE 13 12  2009 |
|  |  |  |  |  | n |  |  |
|  |  |  |  |  |  | 1200 | AMOUNT OF TAX DEPOS ITED 7000  AMOUNT OF TDS 1200  Total 8200 |
| 2 | –200 |  |  |  |  |  |  | AMOUNT  CANNOT BE NEGATIVE | 2, 3, 4 |
| 3 | 3000 | 32/0/-1 |  |  |  |  |  | Wrong date | 2, 3, 6 |
| 4 | 3000 | 31 | -1/0/13 |  |  |  |  | Wrong date/  month | 2, 3, 7 |
| 5 | 3000 | 29 | 2 |  |  |  |  | Wrong date/  month | 2, 3, 7 |

6 3000 31 5 2008 Please enter a

year between

2, 3, 8

2009 and 2099

White-Box Testing on Units/Modules of Income Tax Calculator **575**l

7 4000 31 5 2009 !@#$ Challan number

can contain only characters

2, 3, 5, 11

| 8 | 4000 | 31 | 5 | 2009 | rty |  |  | Please enter correct challan number having 5 to 20 char  acters | 2, 3, 5, 10 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 9 | 4000 | 31 | 5 | 2009 | Rty456 | n | –234 | TDS AMOUNT CANNOT BE NEGATIVE | 2, 3, 5, 12,  14, 16 |
| 10 | 4000 | 31 | 5 | 2009 | Rty456 | t |  | Please enter y or n | 2, 3, 5, 13 |

**DATA FLOW TESTING ON MODULE tax\_ded()**

**Defi nition and Usage nodes**

| **Variable** | **Defi ned At** | **Used At** |
| --- | --- | --- |
| challan | 49 | 50, 52, 63, 70 |
| fl ag | 2, 5, 72, 79, 84 | 73 |
| fl ag1 | 2, 5, 54, 59, 66, | 46 |
| fl ag2 | 2, 5, 96, 100 | 89 |
| fl ag3 | 2, 5, 22, 28, 36, 43 | 20 |
| fl ag4 | 2, 5, 13, 17 | 6 |
| i | 50 | 50, 52, 58 |
| dd | 25 | 26, 33 |
| mm | 32 | 33 |
| yyyy | 39 | 40 |
| choice | 2, 76 | 77 |
| amount\_tax | 9 | 10, 69, 87 |
| tot\_amount\_tax | 3, 87 | 87, 107, 109 |
| amount\_tds | 92 | 93, 108, 109 |
| tot | 109 | 110, 111 |

**576** Software Testing: Principles and Practices

**du and dc paths**

| **Variable** | **du Path(beg-end)** | **dc?** |
| --- | --- | --- |
| challan | 49–50 | Yes |
| 49–52 | Yes |
| 49–63 | Yes |
| 49–70 | Yes |
| fl ag | 2–73 | No |
| 5–73 | No |
| 72–73 | Yes |
| 79–73 | Yes |
| 84–73 | Yes |
| fl ag1 | 2–46 | No |
| 5–46 | Yes |
| 54–46 | No |
| 59–46 | No |
| 66–46 | Yes |
| fl ag2 | 2–89 | No |
| 5–89 | Yes |
| 96–89 | Yes |
| 100–89 | Yes |
| fl ag3 | 2–20 | No |
| 5–20 | Yes |
| 22–20 | No |
| 28–20 | No |
| 36–20 | No |
| 43–20 | Yes |
| fl ag4 | 2–6 | No |
| 5–6 | Yes |
| 13–6 | Yes |
| 17–6 | Yes |
| i | 50–50 | No |
| 50–52 | Yes |
| 50–58 | Yes |
| dd | 25–26 | Yes |
| 25–33 | Yes |
| mm | 32–33 | Yes |
| yyyy | 39–40 | Yes |

choice 2–77 No

White-Box Testing on Units/Modules of Income Tax Calculator **577**l

76–77 Yes

| amount\_tax | 9–10 | Yes |
| --- | --- | --- |
| 9–69 | Yes |
| 9–87 | Yes |
| tot\_amount\_tax | 3–87 | No |
| 3–107 | No |
| 3–109 | No |
| 87–87 | No |
| 87–107 | Yes |
| 87–109 | Yes |
| amount\_tds | 92–93 | Yes |
| 92–108 | Yes |
| 92–109 | Yes |
| tot | 109–110 | Yes |
| 109–111 | Yes |

**BASIS PATH TESTING ON MODULE male\_tax()**

**Source code of male\_tax()**

fl oat male\_tax(fl oat taxable)

{

fl oat t;

1 if(taxable<=150000)

2 {

3 return 0;

4 }

5 else if((taxable>150000)&&(taxable<=300000))

6 {

7 t=10\*taxable/100;

8 }

9 else if((taxable>300000)&&(taxable<=500000))

10 {

11 t=20\*taxable/100;

12 }

13 else

14 {

15 t=30\*taxable/100;

16 }

17 return t;

18 }

**578** Software Testing: Principles and Practices

**DD Graph for Module male\_tax()**

N1

1

N2 N3 N4 N5 2–4 5–8 9–12 13–16

R1 R2 R3

N6

**Cyclomatic Complexity of male\_tax()**

1. V(G) = *e* – *n* + 2*P*

= 8 – 6 + 2

= 4

17–18

2. V(G) = Number of predicate nodes + 1

= 3 + 1

= 4

3. V(G) = No. of regions

= 4

**Independent Paths of male\_tax()**

Since the cyclomatic complexity of the graph is 4, there will be 4 independent paths in the graph, as shown below:

1. N1 N2 N6

2. N1 N3 N6

3. N1 N4 N6

4. N1 N5 N6

**Test Case Design on male\_tax() from the list of Independent Paths**

| **Test Case ID** | **Input**  **Taxable income** | **Expected Output** | **Independent path covered by Test Case** |
| --- | --- | --- | --- |
| 1 | 147000 | return 0 | 1 |
| 2 | 123000 | return 12300 | 2 |
| 3 | 340000 | return 68000 | 3 |
| 4 | 620000 | return 186000 | 4 |

White-Box Testing on Units/Modules of Income Tax Calculator **579**l

**BASIS PATH TESTING OF MODULE fem\_tax()**

**Source code of fem\_tax()**

fl oat fem\_tax(fl oat taxable)

{

fl oat t;

1 if(taxable<=180000)

2 {

3 return 0;

4 }

5 else if((taxable>180000)&&(taxable<=300000))

6 {

7 t=10\*taxable/100;

8 }

9 else if((taxable>300000)&&(taxable<=500000))

10 {

11 t=20\*taxable/100;

12 }

13 else

14 {

15 t=30\*taxable/100;

16 }

17 return t;

18 }

**DD Graph for Module fem\_tax()**

1

N1

N2 N3 N4 N5 2–4 5–8 9–12 13–16

R1 R2 R3

N6

**Cyclomatic Complexity of fem\_tax()** 1. V(G) = *e* – *n* + 2*P*

= 8 – 6 + 2

= 4

17–18

**580** Software Testing: Principles and Practices

2. V(G) = Number of predicate nodes + 1

= 3 + 1

= 4

3. V(G) = No. of regions

= 4

**Independent Paths of fem\_tax()**

Since the cyclomatic complexity of the graph is 4, there will be 4 independent paths in the graph, as shown below:

1. N1 N2 N6

2. N1 N3 N6

3. N1 N4 N6

4. N1 N5 N6

**Test Case Design on fem\_tax() from the list of Independent Paths**

| **Test Case ID** | **Input**  **Taxable income** | **Expected Output** | **Independent path covered by Test Case** |
| --- | --- | --- | --- |
| 1 | 180000 | return 0 | 1 |
| 2 | 270000 | return 27000 | 2 |
| 3 | 430000 | return 86000 | 3 |
| 4 | 620000 | return 186000 | 4 |

**BASIS PATH TESTING OF module senior\_tax()**

**Source Code of senior\_tax()**

fl oat senior\_tax(fl oat taxable)

{

fl oat t;

1 if(taxable<=225000)

2 {

3 return 0;

4 }

5 else if((taxable>225000)&&(taxable<=300000))

6 {

7 t=10\*taxable/100;

8 }

9 else if((taxable>300000)&&(taxable<=500000))

10 {

White-Box Testing on Units/Modules of Income Tax Calculator **581**l

11 t=20\*taxable/100;

12 }

13 else

14 {

15 t=30\*taxable/100;

16 }

17 return t;

18 }

**DD Graph for Module senior\_tax()**

1

N1

N2 N3 N4 N5 2–4 5–8 9–12 13–16

R1 R2 R3

N6

17–18

**Cyclomatic Complexity of senior\_tax()**

1. V(G) = *e* – *n* + 2*P*

= 8 – 6 + 2

= 4

2. V(G) = Number of predicate nodes + 1

= 3 + 1

= 4

3. V(G) = No. of regions

= 4

**Independent Paths of senior\_tax()**

Since the cyclomatic complexity of the graph is 4, there will be 4 independent paths in the graph as shown below:

1. N1 N2 N6

2. N1 N3 N6

3. N1 N4 N6

4. N1 N5 N6

**582** Software Testing: Principles and Practices

**Test case design on senior\_tax from the list of independent paths**

| **Test Case ID** | **Input Taxable income** | **Expected Output** | **Independent path covered by**  **Test Case** |
| --- | --- | --- | --- |
| 1 | 225000 | return 0 | 1 |
| 2 | 228000 | return 22800 | 2 |
| 3 | 430000 | return 86000 | 3 |
| 4 | 620000 | return 186000 | 4 |

**BASIS PATH TESTING OF MODULE gtd()**

**Source code of gtd()**

void gtd(fl oat gross\_income,fl oat net\_deductions,fl oat tax\_ded)

{

1 fl oat taxable\_income,tax,edu\_cess,surcharge=0,difference,total\_tax; 2 taxable\_income=gross\_income-net\_deductions;

3 printf(“\nTaxable Income\t\t:%f”,taxable\_income);

4 if((s==‘m’)||(s==‘M’))

5 {

6 tax=male\_tax(taxable\_income);

7 printf(“\nTax\t\t\t:%f”,tax);

8 }

9 else if((s==‘f’)||(s==‘F’))

10 {

11 tax=fem\_tax(taxable\_income);

12 printf(“\n\Tax\t\t\t:%f”,tax);

13 }

14 else if((2009-yr1)>65)

15 {

16 tax=senior\_tax(taxable\_income);

17 printf(“\nTax\t\t\t:%f”,tax);

18 }

19 if(tax>1000000)

20 {

21 surcharge=10\*tax/100;

22 printf(“\nSurcharge\t\t:%f”,surcharge);

23 }

24 edu\_cess=3\*tax/100;

25 printf(“\nEducational cess\t:%f”,edu\_cess);

26 total\_tax=tax+surcharge+edu\_cess;

27 printf(“\nTax payable\t\t:%f”,total\_tax);

28 printf(“\nTotal tax deducted\t:%f”,tax\_ded);

White-Box Testing on Units/Modules of Income Tax Calculator **583**l

29 difference=total\_tax-tax\_ded;

30 if(difference>0)

31 {

32 printf(“\nTax payable\t\t:%f”,difference);

33 }

34 else if(difference<0)

35 {

36 printf(“\nRerfundable\t%f”,-1\*difference);

37 }

38 else

39 {

40 printf(“\nNo tax”);

41 }

42 }

**DD Graph for Module gtd()**

1–3 N1

4 N2

N3 N4 N5

5–8 9–13 14–18

R1 R2

R3

19

20–23 24–29 30

N6 N7 N8 N9

N11 N12

N10 34–39 38–41 31–33

R4 R5

N13

42

**584** Software Testing: Principles and Practices

**Cyclomatic Complexity of gtd()**

1. V(G) = *e* – *n* + 2*P*

= 17 – 13 + 2

= 6

2. V(G) = Number of predicate nodes + 1

= 5 + 1

= 6

3. V(G) = No. of regions

= 6

**Independent Paths of gtd()**

Since the cyclomatic complexity of the graph is 6, there will be 6 independent paths in the graph as shown below:

1. N1 N2 N3 N6 N8 N9 N10 N13

2. N1 N2 N4 N6 N8 N9 N10 N13

3. N1 N2 N5 N6 N8 N9 N10 N13

4. N1 N2 N3 N6 N7 N8 N9 N10 N13

5. N1 N2 N3 N6 N7 N8 N9 N11 N13

6. N1 N2 N3 N6 N7 N8 N9 N12 N13

**Test case design on gtd() from the list of independent paths**

| **Test**  **Case ID** | **Inputs** | | | | | **Expected Output** | **Independent**  **path covered by Test Case** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Gross\_**  **income** | **Net\_**  **deductions** | **Tax\_ded** | **Status** | **Age** |
| 1 | 320000 | 2000 | 1000 | male | 34 | Tax 30000  Educational cess 900  Tax payable 30900  Total Tax deducted 1000 Tax payable 29900 | 1 |
| 2 | 340000 | 2000 | 1000 | female | 34 | Tax 32000  Educational cess 960  Tax payable 32960  Total Tax deducted 1000 Tax payable 31960 | 2 |

3 420000 2000 1000 male 67 Tax 84000

Educational cess 2520

Tax payable 86520 Total Tax deducted 1000 Tax payable 85520

1, 3

White-Box Testing on Units/Modules of Income Tax Calculator **585**l

4 12000000 2000 1000 male 34 Tax 3629400

Surcharge 362940

Educational cess 108882 Tax payable 4101222 Total Tax deducted 1000 Tax payable 4100222

1, 4

| 5 | 320000 | 2000 | 32000 | male | 34 | Tax 30000  Educational cess 900  Tax payable 30900  Total Tax deducted 32000 Refundable 1100 | 1, 5 |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 6 | 320000 | 2000 | 30900 | male | 34 | Tax 30000  Educational cess 900  Tax payable 30900  Total Tax deducted 30900 No tax | 1, 6 |

*To*

*my parents*

*who have made me capable to struggle in this world*