

Multi Criteria Test Case Prioritization based on Additional Strategy

By

Robeala Abid

A research thesis submitted to the Department of Computer Science,
Capital University of Science and Technology, Islamabad
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE



**DEPARTMENT OF COMPUTER SCIENCE
CAPITAL UNIVERSITY OF SCIENCE AND TECHNOLOGY
ISLAMABAD
2017**



**CAPITAL UNIVERSITY OF SCIENCE & TECHNOLOGY
ISLAMABAD**

CERTIFICATE OF APPROVAL

Multi Criteria Test Case Prioritization based on Additional Strategy

by

Robeala Abid

MCS153010

THESIS EXAMINING COMMITTEE

S No	Examiner	Name	Organization
(a)	External Examiner	Dr. Rizwan Faiz	Riphah International, Islamabad
(b)	Internal Examiner	Dr. Muhammad Tanvir Afzal	CUST, Islamabad
(c)	Supervisor	Dr. Aamer Nadeem	CUST, Islamabad

Dr. Aamer Nadeem

Thesis Supervisor

September, 2017

Dr. Nayyer Masood

Head

Department of Computer Science

Dated : September, 2017

Dr. Muhammad Abdul Qadir

Dean

Faculty of Computing

Dated : September, 2017

Certificate

This is to certify that **Miss Robeala Abid** has incorporated all observations, suggestions and comments made by the external evaluators as well as the internal examiners and thesis supervisor. The title of her Thesis is: **Multi Criteria Test Case Prioritization based on Additional Strategy.**

Dr. Aamer Nadeem
(Thesis Supervisor)

Copyright ©2017 by Robeala Abid

All rights reserved. Reproduction in whole or in part in any form requires the prior written permission of Robeala Abid (MCS-153010) or designated representative.

I dedicate my dissertation work to my family, teachers and friends. A special feeling of gratitude is for my loving parents for their love, endless support and encouragement.

ACKNOWLEDGMENT

I am thankful to my Creator Allah Subhana-Watala to have guided me throughout this work and strength. You gave me each day. Whosoever helped me throughout the course of my thesis, whether my parents or any other individual was Your will, so indeed none be worthy of praise but You. I am profusely thankful to my beloved parents who raised me when I was not capable of walking and continued to support me throughout in every department of my life. Without their support help and prayers I could not even think of doing something in life. This work which initially seemed to be a nightmare finally becomes a reality with the kind support and help of many individuals.

I would also like to express special thanks to my supervisor Dr. Aamer Nadeem for his guidance throughout my thesis, for his patience, motivation, enthusiasm, and immense knowledge. I am highly thankful to him and have warm feelings and wishes for him that can't be expressed in words. I can safely say that I haven't learned any other CS subject in such depth than the ones which he has taught. He is the unrivalled supervisor, most respectable and virtuous man. Working under his supervision was honor, pleasure and an experience that I had truly treasured.

I also want to express my gratitude for Zubia Abid, Aneeqa Abid, Zain-ul-Abideen and Anis-ul-Abideen for their ethical backing. I am highly thankful to my friend Faiza Farooq for her help and guidance. I am really blessed to have such a good friend in my life. I would also like to thank CSD research group for being on my thesis guidance.

Finally, I want to say thanks to each and every one who helped me. ALLAH Almighty bless you all.

Ameen

Robeala Abid

Declaration

It is declared that this is an original piece of my own work, except where otherwise acknowledged in text and references. This work has not been submitted in any form for another degree or diploma at any university or other institution for tertiary education and shall not be submitted by me in future for obtaining any degree from this or any other University or Institution.

Robeala Abid

MCS-153010

Abstract

Software testing is performed for assessing quality of program. The main aim of testing is to identify bugs in software program. When software is modified, it is retested to ensure that no new faults have been introduced in the previously tested code and it still works correctly. Such a testing is known as regression testing. The cost of regression testing is very high because the original program has large number of test cases. It is not feasible to execute all test cases for regression testing. The number of test cases increases as the software tends to evolve, thus increasing the cost of regression testing. Test suite minimization, test case selection and test case prioritization are cost reduction techniques used to reduce the cost of regression testing. Prioritization is better than selection and minimization techniques because it does not eliminate any test case. Whereas in selection and minimization technique, some test cases are eliminated. In this thesis, our focus is on prioritization technique. Prioritization technique assigns priority to each test case on the basis of covering some criteria. Priorities are assigned to test cases by using some criterion and are prioritized on the basis of maximum coverage of that criterion. These criteria can be structural entities of source code like statements, branches, functions or can be specification based like interactions between GUI events or requirements etc. Two general strategies named as “Total” and “Additional” are used to assign priorities. “Total” strategy assigns priority on the basis of total number of entities covered by a test case. Whereas, “Additional” strategy chooses the test case with high coverage degree and covering entities that are not so far covered by previously chosen test cases. “Additional” strategy is better than total strategy, as “Total” strategy does not count the number of additional entities which remain uncovered in the previous selection. This can result in repetition of entities or some entities can be missed. Based on these strategies, prioritization techniques can be single criterion based or multiple criteria based. A number of different single criterion and multiple criteria based prioritization techniques have been introduced. Multiple criteria based prioritization techniques are more effective than single criterion based prioritization techniques because these techniques are using more than one coverage criteria and are also increasing the rate of fault detection. The existing multiple criteria based prioritization techniques combine multiple criteria in such a way that “Additional” strategy can not be applied on them.

In this thesis, we propose a new multiple criteria based test case prioritization algorithm. Our algorithm considers two criteria to prioritize test cases using “Additional” strategy. One criterion is considered as primary and other is considered as secondary. Primary criterion is used to prioritize the test cases whereas secondary criterion is used to break the tie among test cases when two or more test cases provide equal coverage of entities of first criterion.

Our algorithm performs better than the existing techniques. We have evaluated and compared our technique with existing single and multiple criteria based prioritization techniques. We have used three subject programs: Triangle program, Next Date program and Commission program for evaluation of these techniques. APFD (Average Percentage of Fault Detection) metric is used to make comparison of these techniques. The APFD values shows that our technique yields better results than the existing techniques.

Contents

List of Figures	xii
List of Tables	xiii
Chapter 1 : Introduction	1
1.1 Regression Testing	2
1.2 Prioritization Approach	5
1.2.1 Prioritization Criteria	5
1.2.2 Prioritization Algorithms	6
1.3 Problem Statement	7
1.4 Research Questions	8
1.5 Research Methodology	8
1.6 Research Contributions	9
1.7 Thesis Organization	10
Chapter 2 : Literature Review	11
2.1 Single Criterion based Prioritization Techniques	12
2.1.1 White Box Prioritization Techniques	12
2.1.2 Black Box Prioritization Techniques	15
2.1.3 Analysis and Comparison	17
2.2 Multiple Criteria based Prioritization Techniques	19
2.2.1 Analysis and Comparison	20
Chapter 3 : Proposed Approach	23
3.1 Proposed Prioritization Algorithm	23
3.1.1 Multiple Criteria based Prioritization Algorithm	25
3.1.2 Example of Proposed Prioritization Algorithm	27
Chapter 4 : Implementation	30
4.1 Implementation details	30
4.2 Usage Details	31
Chapter 5 : Results and Discussion	33
5.1 Subject programs	33
5.2 Priority Lists	34
5.3 Comparison	38

Chapter 6 : Conclusion	51
6.1 Future Work	53
References	54
Appendix A : Source Code of Subject Programs	59
Appendix B : Test Data for Subject Programs	65

List of Figures

Figure 1-1: Regression Testing basic Concept	2
Figure 1-2: Techniques of Regression Testing	5
Figure 3-1: Context Digram of Proposed Prioritization Algorithm.....	24
Figure 3-2: Multiple Criteria based Prioritization Algorithm.....	26
Figure 4-1: Class digram of Proposed Algorithm.....	30
Figure 4-2: Output of Example Program on Console	32
Figure 5-1: Graphical Representation of APFD	40
Figure 5-2: Graphical Representation of fault detection of test cases for Triangle Program	44
Figure 5-3: Graphical Representation of fault detection of test cases for Commission Program.	46
Figure 5-4: Graphical Representation of fault detection of test cases for NextDate Program	49

List of Tables

Table 2-1: Comparison of Single Criterion based prioritization techniques	17
Table 2-2: Comparison of Multiple Criteria based prioritization techniques	21
Table 3-1: Example Data	27
Table 5-1: Subject Programs summary	34
Table 5-2: Priority Lists for Triangle Program	35
Table 5-3: Priority Lists for Commission Program	36
Table 5-4: Priority Lists for NextDate Program	37
Table 5-5: APFDs of Subject Programs.....	39
Table 5-6: Fault detection of test cases for Triangle Program.....	41
Table 5-7: Fault detection of test cases for Commission Program.....	45
Table 5-8: Fault detection of test cases for NextDate Program	46

Chapter 1 : Introduction

Software testing is performed for evaluating quality of software and making improvements in it. The main aim of testing is to identify errors in the software product and to fix them. It additionally guarantees that whether customer's requirements are being refined by the software (Chauhan and Singh, 2014). Testing of software is a critical and expensive phase of software development (Jeffrey and Gupta, 2008). Quality and consistency of software can be ensured through testing. Testing should be able to use least resources, as already the software design and development phase exceed the resources to much extent (Feldt et al., 2016). Testing identifies the errors or defects in the software product, but it does not guarantee that software under test is error free (Quadri and Farooq, 2010). Faults can occur at any stage of development process and must be identified and removed to control their further transmission in the next steps of development phases (Baresi and Pezz`e, 2006). According to Burnstein (2006), "Software Testing is generally described as a group of procedures carried out to evaluate some aspect of a piece of software" or Software Testing can be described as "a process used for revealing defects in software, and for establishing that the software has attained a specified degree of quality with respect to selected attributes". According to the report of "National Institute for Standards and Technology (NIST)", US economy is utilizing almost \$60 billion per year on software testing (Bryce and Colbourn, 2006). Effective testing can spare roughly \$22 billion. Therefore advanced software testing is required to reduce the cost. Software can have different types of errors such as design error, input error, hardware error, statement error, specification error etc. Different testing types are used to remove these errors (Singh et al., 2010). There are different types of software testing (Hooda et al., 2015), which include white box testing, black box testing (khan et al., 2012), grey box testing (Saxena and Singh, 2014), and model based testing (Elbaum et al., 2002).

The most important part of SQA (Software Quality Assurance) is software testing. On making modifications in any software, new test suites are created to test the modified portion of software (Jeffrey and Gupta, 2008). In System Development Life Cycle (SDLC), maintenance phase is more expensive because during this phase, the software undergoes some changes and is

updated continuously. The updated software also needs testing to identify faults in the software. So, the maximum cost of maintenance phase is consumed on regression testing (Hooda et al., 2015).

1.1 Regression Testing

When software is revised, it is retested to ensure that no new faults have occurred in the earlier tested code and it still works correctly. Such testing is known as regression testing (Elbaum et al., 2002). Formally; consider P is an original program, P' as its modified form, T is test suite to test program P, the objective of regression testing is to utilize the test suite T such that it will be able to detect the errors found in P'. Following figure explains this concept more accurately.

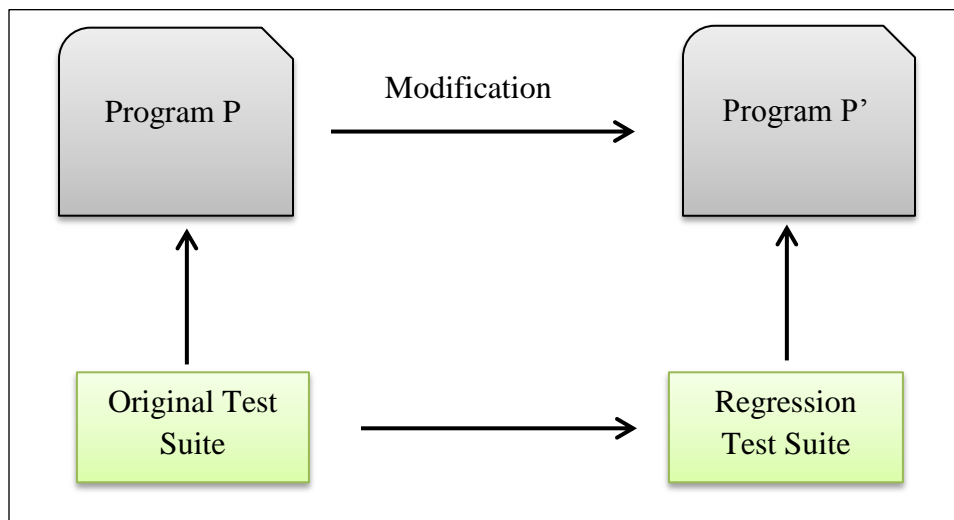


Figure 1-1: Regression Testing basic Concept

Regression testing is vital for real-world software development projects. It is used to test the new changes to computer programs. Regression testing ensures the validity of a software program upon making some changes in the program (Yoo and Harman, 2012). Regression testing might begin in the development phase after identification and correction of errors by reusing the existing test cases to retest the system.

The most important phase of software is the maintenance phase, in which the software which is delivered to customer should be maintained by the software development team (Duggal et al., 2008). Many modifications might also occur at maintenance phase where the system is corrected

Some modules are being added or removed. There are two sorts of maintenance: corrective maintenance and progressive maintenance.

- In progressive maintenance, specifications of software are changed like adding a new module.
- In corrective maintenance, only errors are identified and corrected. Specification of software remains same.

Based on above types of modification/maintenance, Leung and White (1989) have introduced two types of regression testing: Progressive regression testing and Corrective regression testing.

- When software specifications are reformed by adding or removing a module, then Progressive regression testing is performed to test this changed specification of software.
- Corrective regression testing identify and correct the errors in the software. It is applied when software specifications remain unchanged.

The test cases for regression testing are categorized in to five classes:

1. Retestable
2. Reusable
3. Obsolete
4. New-Structural
5. New-Specification

The three classes Retestable, Reusable and Obsolete contain already existing test cases of test suite. The classes New-Structural and New-Specification consists of those test cases which are newly created for regression testing to test the modified program. Those parts which remain unaffected are tested by Reusable test cases. Retestable test cases test only the changed or affected parts of program. Some test cases test the modified parts and some of them test the unmodified parts of program. Also, some obsolete test cases exist which cannot be used anymore because due to modifications, their input/output relation is disturbed and they never again test what they were created to test. The required structural coverage of such test cases is also lost due to modification, although they are structural test cases. New-Structural test cases test the code of changed program and it also increases the structural coverage of code. New-Specification test cases test the specification of modified program (Leung and White, 1989).

As the software is upgraded or enhanced, the number of test cases increases exponentially. This makes regression testing a complicated, inefficient and costly process. Simplest technique of regression testing is to “retest all”. But this technique becomes difficult and costly for such softwares which are large and which have maximum number of test cases. This makes testing a time consuming and expensive process (Yoo and Harman, 2012). Different techniques have been introduced to reduce the cost of regression testing. These techniques including test case selection, test suite minimization, and test case prioritization are shown in Figure 1.2.

- Test case selection selects the correct subset of test cases which are valid for required parts of the software and which traverse the required parts of the software. Remaining test cases are eliminated in this technique (Rothermel and Harrold, 1994).
- Minimization is another technique of regression testing in which test suite is minimized by the removing the redundant test cases. Test cases checking same functionality are removed by this technique. The selected test cases are placed in a set which is called as minimal hitting set (Garey, 1979). To identify a minimal hitting set from test suite is an NP-Complete problem. These problems have exponential time complexity. Different heuristics have been introduced to solve the minimal hitting set problem (Chen et al., 1996; Harrold et al., 1993; Horgan et al., 1992; Offutt et al., 1995).
- The third technique for reducing the cost of regression testing is test case prioritization. Test case prioritization assigns priority to test cases by considering some coverage criteria. This technique is effective than the other two techniques of cost reduction because it does not eliminate any test case from test suite. It assigns priority to test cases in such a way that the test case with highest priority is executed first (Yoo and Harman, 2012). It also provides confidence that if testing process is prematurely halted then all the important test cases have already been executed. Prioritized lists can also be used by the software testers to determine when the test process can be terminated (Leung and White, 1989).

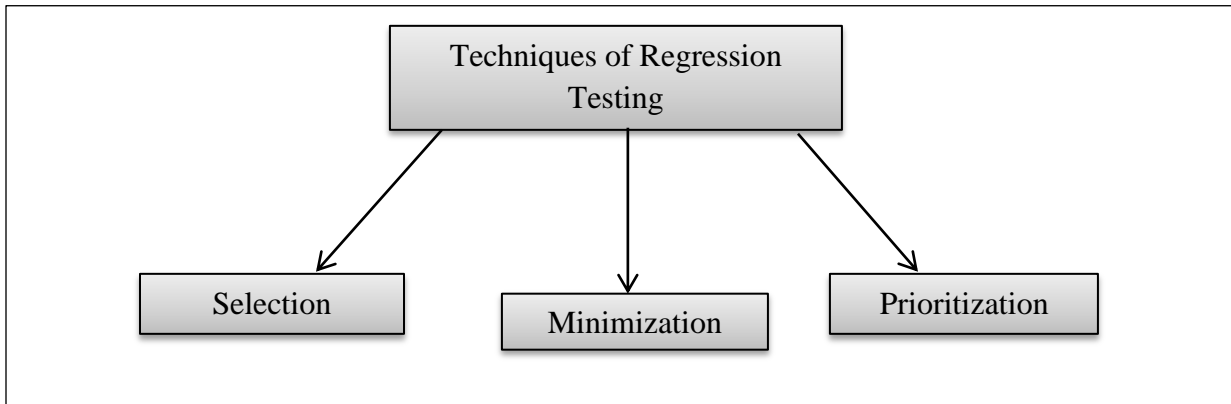


Figure 1-2: Techniques of Regression testing

1.2 Prioritization Approach

Prioritization technique assigns priority to each test case using some criterion. Test case prioritization was first proposed by Wong et al. (1997). Afterwards, many researchers have worked out on test case prioritization problem (Elbaum, 2000; Elbaum, 2002; Korel et al., 2007; Srivastava, 2008; Malishevsky et al., 2006; Elbaum et al., 2004; Li et al., 2007; Rothermel et al., 2001). From the literature, the formal definition of test case prioritization (Rothermel et al., 2001) is given as below:

Definition: The Test Case Prioritization Problem:

Given: T , a test suite; PT , the set of permutations of T ; and a function from PT to the real numbers, $f: PT \rightarrow \mathbb{R}$

Problem: To find $T' \in PT$ such that $(\forall T'' \in PT) (T'' \neq T') [f(T') \geq f(T'')]$.

Prioritization can be coverage based, fault based or distribution based. In this thesis, we are using coverage based prioritization because it is better than other. Fault based prioritization considers the fault exposing potential of a test case. Priority is assigned to each test case on the basis of number of faults revealed by each test case. During prioritization, fault data of each test case is not available and FEP based approaches estimate the value of fault for each test case which is not accurate.

1.2.1 Prioritization Criteria

Prioritization technique assigns priority to test cases by considering some criteria. Test case with maximum coverage of that criterion is assigned highest priority and test case with minimum coverage of criterion will get lowest priority (Fazlalizadeh et al., 2009). These criterion can be the structural entities of source code like statements, branches, function etc. (Rothermel et al., 1999) or can be specification based like interaction between graphical events or customer priority etc. (Henard et al., 2016).

Prioritization can be single criterion based or multiple criteria based.

Single Criterion based: In single criterion based prioritization, only one criterion is used to prioritize the test cases. Criterion can be statement coverage, branch coverage etc.

Multiple Criteria based: In multiple criteria based prioritization, more than one criteria is used to prioritize the test cases. Test cases are prioritized by combining multiple criteria in different ways.

Using different criteria, priority is assigned on the basis of two general strategies named as “Total” and “Additional” proposed by Rothermel et al. (2001). “Total” strategy considers the total number of entities covered by test cases. This can result in repetition of entities if two or more test cases have same coverage data. Also, if testing is halted prematurely, then some entities can be missed. “Additional” strategy is better than “Total” strategy as it not only considers the number of entities covered by each test case but also the additional entities which remains uncovered by previously selected test case. This strategy reduces the repetition of entities and it will also ensure the coverage of each entity by atleast one test case (Rothermel et al. ,2001).

1.2.2 Prioritization Algorithms

Prioritization is of increasing importance and lot of work has been done on it in literature. Many test case prioritization algorithms have been introduced which are prioritizing the test cases by using different criterions. Following are different prioritization algorithms.

A) Test case prioritization based on Genetic Algorithms:

Genetic algorithm:

In Genetic algorithm (Kaur and Goyal, 2011) random population is selected from given set, using fitness function the population is replaced with a new set of population. Fitness function of genetic algorithm is based on total code coverage.

Combined Genetic and Simulated Annealing Algorithm:

This algorithm called as GASA (Maheswari and Mala, 2015) combines the two algorithms: GA (Genetic Algorithm) and SA (Simulated Annealing). Advantage of both algorithms is used by GASA. It uses the quick processing feature of GA and effective solution feature of SA. Initially, solution is produced randomly using GA and afterwards it is refined using SA.

B) Test case prioritization based on Greedy Algorithms:

Greedy Algorithm: Greedy algorithm assigns priority on the basis of total number of entities covered by test cases. Test case covering maximum number of entities is given highest priority. Test case with minimum coverage of entities will get lowest priority (Elbaum, 2002).

Additional Greedy Algorithm: Additional greedy algorithm assigns priority to test cases on the basis of number of entities which remains uncovered in previous selection (Elbaum, 2002).

2-Optimal Algorithm: Another greedy algorithm for prioritization is 2-Optimal algorithm (Lin, 1965). It iteratively selects such test cases which covers the maximum uncovered entities in the previous selection. It is little bit different from additional greedy in sense that two best test cases are selected in each iteration by improving the coverage of additional entities. On gaining complete coverage, it resets the coverage.

Above all algorithms uses some coverage data to prioritize the test cases. Test case is assign priority on the basis of entities it covers. Test case covering maximum entities is given highest priority. Our technique will use the concepts of additional greedy algorithm.

1.3 Problem Statement

The majority of the current prioritization techniques are based on single criterion. In single criterion based prioritization techniques, test case is selected randomly if there is an occurrence of a tie in more than one test case. This reduces code coverage to much extent, and the early fault detection rate is also reduced. Prioritization can be improved by using multiple criteria. The

existing multiple criteria based prioritization techniques are not appropriate because these techniques are using “Total” strategy to prioritize the test cases. These existing techniques are unable to use “Additional” strategy because these techniques combines the coverage information of all criterion. “Additional” strategy cannot be applied on combined coverage of all criterion. Previous studies have shown that Additional strategy is better than Total strategy as Additional strategy update the values of remaining test cases by using the information of previously prioritized test cases (Henard et al., 2016; Saha et al., 2015). Whereas the “Total” strategy does not count the number of additional entities which remains uncovered in the previous selection. It only considers the total number of entities covered by test cases which results in repetition of entities and also some entities can be missed if testing is halted prematurely.

1.4 Research Questions

In this research, we use multiple criteria to generate a priority list of test cases. However, the following factor must be taken into account:

RQ 1:What are the existing and most commonly used test case prioritization techniques?

Literature survey is carried out to analyze the existing prioritization techniques and also to find out the most commonly used techniques.

RQ 2:What are the gaps in existing single and multiple criteria based prioritization techniques?

Through in depth study of literature, we identify these gaps.

RQ 3:Does the proposed multiple criteria based prioritization technique improves fault detection rate when compared with existing prioritization techniques using single criterion or multiple criteria?

Different experiments’ are performed on different subject programs and for comparison of proposed technique with existing techniques, Average Percentage of Fault detection (APFD) metric of all these techniques is calculated for each subject program.

Our research is carried out to answer the above mentioned research questions along with the implementation of Multiple Criteria based prioritization algorithm.

1.5 Research Methodology

First of all, we have done literature review to identify the most relevant and most commonly used white box and black box prioritization techniques. After studying various prioritization techniques, we have reached the conclusion that most of these techniques are single criterion based. If there should arise an occurrence of a tie, these techniques arbitrarily select a test case by overlooking the fact that may be the removed test case has higher coverage than the selected one. The test case with the better ability to find the faults may have been given the lowest priority by these techniques. All the existing multiple criteria based prioritization techniques selects test cases using “Total” strategy, which is not appropriate because it is not considering the additional entities covered by test cases.

2. To overcome the gap in existing techniques, we have proposed another algorithm that will rank the test cases based on multiple criteria, so that fault detection rate can be increased.
3. The implementation of our approach will be performed in following steps:
 - i. In the first phase, we collect all the data including the test suite and coverage detail of two criteria. After collecting the data, proposed prioritization algorithm will be applied to generate the priority list of test cases.
 - ii. We have also implemented some existing prioritization techniques and priority list for these techniques is generated.
 - iii. In the next step, faults are seeded in the program by using mutation operators. Mutants are created.
 - iv. Using these mutants, a mapping is formed between test cases and faults seeded, which shows that which test case detects which fault.
4. For comparison, the fault detection rate will be used as the main parameter. Using above mapping and priority lists; we have calculated the APFD of our proposed and existing techniques for each subject program. For comparison, the fault detection rate will be used as the main parameter.

1.6 Research Contributions

In this research work, we have proposed a new multiple criteria based prioritization technique. This technique is using two coverage criteria, one is considered as primary and other is considered as secondary. Primary criterion assigns priority to each test case and secondary criterion is used to break the tie among test cases. Test case which is covering maximum entities of primary criterion is selected and will be given highest priority. If more than one such test case exists, a tie occurs. This tie is broken by using secondary criterion. Test case which covers maximum additional entities of secondary criterion will be given higher priority.

1.7 Thesis Structure

The rest of thesis is organized as; Chapter 2 surveys the existing work on test case prioritization. Chapter 3 and Chapter 4 discuss the proposed approach and its implementation details. Chapter 5 presents experimental results of three subject programs and comparing it with existing techniques to evaluate the performance of our proposed technique. Finally Chapter 6 finishes up the entire work and furthermore gives some future research directions.

Chapter 2 : Literature Review

Test case prioritization finds an ordering of test cases that provides the maximum benefits to the software testers. The software testers assign priority to each test case in test suite to ensure that the test cases with higher priorities are run earlier in the testing process (Yoo and Harman, 2012). The fault detection rate of test suites is affected by the sequence of the execution of test cases. This detection, in fact, is the measurement of early detection of faults by test suite in the testing process (Elbaum et al. , 2002). Software tester can use the priority list to decide when to stop testing in case of having not enough resources to execute all tests. A prioritized list will also increase the probability that in any case, if testing is halted earlier due to some reason, the most important tests with higher priorities will have been executed, thus increasing the rate of fault detection and enabling debugging in early stages of testing (Elbaum et al., 2002).

Effectiveness of prioritization techniques is evaluated by the rate of fault detection i.e. Average Percentage of Faults Detected (APFD) which is the measure of how early faults are detected in the testing process by the test suite (Rothermel et al., 1999). Following formula is used to calculate the APFD:

$$APFD = 1 - \frac{TF_1 + \dots + TF_m}{nm} + \frac{1}{2n}$$

Where T is the test suite containing n test cases and F is the set of m faults revealed by T. For ordering T', TFi is the order of the first test case that reveals the ith fault Fi.

Priority is assigned on the basis of two general strategies named as “Total” and “Additional” proposed by Rothermel et al. (2001). “Total” strategy considers the total number of entities covered by test cases. “Additional” strategy chooses the test case with high coverage degree and covering entities that are not so far covered by previously choosed test case. Additional strategy is better than total strategy, as “Total” strategy does not count the number of additional entities which remains uncovered in the previous selection. It only considers the number of entities

covered by each test case. If more than one test case is covering same entities then repetition of entities can happen and also some entities can be missed if testing is halted prematurely.

Based on these strategies, prioritization techniques can be single criterion based or multiple criteria based. Single criteria based prioritization techniques uses only one coverage criterion for prioritization of test cases. Multiple criteria based prioritization techniques uses more than one criteria to prioritize the test cases. In the literature, most of the work is based on single criteria based prioritization techniques. Multiple criteria based prioritization techniques are studied less as compared to single criteria based prioritization techniques. Different criteria are used for assigning priority to test cases. These criteria can be black box or white box. White box criteria are the elements of source code including statements, branches, spanning entities or functions for assigning the priority (Rothermel et al., 2001). Black box criteria are specification of the system including the interaction between the events, requirements priority etc. to prioritize test cases in test suite (Henard et al., 2016).

White box and Black box prioritization are two main categories of prioritization techniques.

White box prioritization techniques use the code of the software to define a criterion for prioritizing the test cases. Black box prioritization techniques have no structural information; rather use the specification of the software to prioritize test cases (Hooda and Chhillar, 2015). Highest priority is assigned to test case with low cost according to some criteria.

In this chapter, different existing prioritization techniques will be discussed.

2.1 Single Criterion based prioritization Techniques:

Different white box and black box prioritization techniques use single criterion to prioritize the test cases. Priority is assigned to each test case on the basis of some criterion. Following are different single criterion based black box and white box prioritization techniques.

2.1.1 White Box Prioritization techniques:

In white box testing,detailed information of internal logic of software product is required. Tester needs complete knowledge of source code for testing (Khan et al. , 2012). In white box prioritization techniques, the test cases are prioritized on the basis of code covered. Different

elements of source code are used by different prioritization techniques. These techniques are given as below.

Coverage based prioritization techniques

Two main white box prioritization strategies were introduced by Rothermel et al. (2001) and Elbaum et al. (2002): “Total” and “Additional”, which include following different prioritization techniques.

i) Total function coverage prioritization: The criterion used in this technique is total number of functions covered. Highest priority is assigned to test case covering maximum number of functions. Test cases with equal number of functions are ordered randomly.

ii) Additional function coverage prioritization: The criterion used in this technique is based on number of uncovered functions. The test case with maximum function coverage is selected and then coverage information of remaining unprioritized test cases is updated on the basis of uncovered functions. Test cases which are covering equal number of functions are ordered randomly. This process continues until all functions are covered by atleast one test case. Remaining test cases are placed randomly in the prioritized list.

iii) Total statement coverage prioritization: The criterion used in this technique is total number of statements covered. Highest priority is assigned to test case covering maximum number of statements of code. Test cases which are covering equal number of statements are ordered randomly.

iv) Additional statement coverage prioritization: This technique ordered the test cases based on number of uncovered statements. The test case with maximum statement coverage is selected and then coverage information of remaining unprioritized test cases is updated on the basis of uncovered statements. Test cases which are covering equal number of statements are ordered randomly. This process continues until all statements are covered by atleast one test case. Remaining test cases are placed randomly in the prioritized list.

v) Total branch coverage prioritization: The criterion used in this technique is total number of branches covered. Highest priority is assigned to test case covering maximum number of

branches. Branch coverage is the possible outcome (True and false) of the condition. The function itself is considered as a branch in case of having no branch in function, branch is said to be covered by each test case which calls that function.

vi) Additional branch coverage prioritization: The criterion used in this technique is based on number of uncovered branches. The test case with maximum branch coverage is selected and then coverage information of remaining unprioritized test cases is updated on the basis of uncovered branches. Test cases which are covering equal number of branches are ordered randomly. This process continues until all branches are covered by atleast one test case. when all branches are covered, then remaining test cases are placed in random order in priority list.

vii) Total FEP based prioritization: In this technique, test cases are assigned prioritizes on the basis of their fault exposing potential. Mutation testing is applied to create mutants of a program and mutant score of a test case for each statement is calculated. Finally, summation is obtained by adding the mutation score of all statements for a specific test case and the test case with maximum mutation score is given highest priority.

viii) Additional FEP based prioritization: Additional FEP based prioritization is same as total FEP with the addition of a new factor called confidence. Confidence $C(s)$ of a statement is the probability that statement s is correct. Value of $C(s)$ is between 0 and 1, inclusive. If we execute a test case on a statement, and the test case does not expose any fault in that statement, then the value of confidence $C(s)$ increases and the new value of confidence will be $C'(s)$ and the additional confidence gained is $Caddi(s)$. $Caddi(t)$ is defined as the additional confidence gained by executing test case t on program P .

Highest priority is assigned to that test case which has maximum $Caddi(t)$ value. After selection of a test case, the value of confidence $C(s)$ for all statements covered by test case t is updated. Also, the $Caddi(s)$ values are updated for remaining statements against remaining test cases. This process continues until all test case are prioritized.

History based prioritization

Kim and Porter (2002) proposed a technique, which is stated as “History-based prioritization”. Historical information is used to select set of test cases for new version of program. Firstly, RTS

techniques are used to produce T' for the test suite T . Afterwards, every test case of T' is given a selection probability. Previous performance is considered to assign selection probability to each test case. These techniques use test histories data which is based on execution history, number of entities covered and fault detection rate. Finally, a test case is selected on the basis of probabilities assigned in previous step. This step is repeated until all test cases are ordered.

Additional Spanning Entities Coverage based Prioritization

Marre' and Bertolino (2003) presented an idea to use spanning set of entities as a criterion for prioritization of test cases. Spanning set of entities is obtained from the subsumption relationship between entities. Subsumption relationship between two entities is defined as, given two entities E and E' , if every complete path which covers E is also covering E' , it means that E subsumes E' , but there exists some entities which are not covered by any other entities. Spanning entities are those entities which subsume other entities without being itself subsumed by other entities. This technique was further divided into Additional Spanning Statements and Additional Spanning Branches. Test cases are ordered on the basis of maximum number of uncovered spanning statements or spanning branches.

2.1.2 Black Box Prioritization techniques

Black box testing does not involve the internal working of software product. It only inspects the architecture of software product without having any access to source code (Khan et al., 2012). Specification based prioritization criteria are defined by black box prioritization techniques without having knowledge of structure of software. Test cases are built around specifications and requirements of software. Following are different prioritization techniques of black box testing from the literature.

Interaction Coverage Based Prioritization

Bryce and Memon (2007) proposed a prioritization technique which is extending the t-way software interaction over sequences of events. Number of event combinations and sequences rises exponentially with the increase in number of events, so management of test suite for event driven systems is difficult. The proposed algorithm greedily selects a test case covering the maximum number of formerly uncovered t-tuples of event interactions between unique windows. Test cases which are covering equal number of event interactions are ordered randomly.

Requirements clustering based prioritization

Software under test may have many requirements with varying priority. All requirements are not significant, while some are of high priority. Arafeen et al. (2013) introduced a new approach which uses requirements information to prioritize the test cases. Their proposed approach extracts useful words from the text using text mining approach. On the basis of these words, relevant requirements are clustered. Test cases are prioritized within these clusters using code complexity. In the next step, clusters are prioritized using code modification information and customer assigned requirements priority. Finally, different selection methods are used to select the test cases from each cluster by visiting each of them.

Prioritization of Requirements for Testing (PORT)

Software quality can be increased by exploiting the testing effort on those requirements which are essential to customer. “Prioritization of requirements for testing” is value based technique proposed by Srikanth et al. (2005). This technique contains four factors values: customer-assigned priority of requirements, developer-perceived implementation complexity, requirements volatility and fault proneness. These factors value are further used to calculate the Prioritization Factor Value (PFV). Prioritization Factor Value (PFV) “is the summation of the product of factor value and the assigned factor weight for each of the factors”. Prioritization factor value reports about the importance of each requirement. PFV of a requirement is essential to calculate the WP (weighted priority) of its associated test cases. Weighted priority is “The PFV contribution of the requirement(s) the test case maps”. Priority is assigned on the basis of weighted priority value of each test case. Test case with high weighted priority value is allotted highest priority and so on.

History based test case prioritization

Black box techniques have very limited information because the source code of a program is not available in black box environment. Qu et al. (2007) introduced a black box technique that prioritizes the test cases on the basis of run-time and test history information. Test suite is initialized from history information. Fault detection relationship of tests cases is formed by using test case relation matrix R, and then a test case covering maximum faults is selected from test suite and it is run. Remaining test cases are arranged using test case relation matrix and run time information.

A Hierarchical System Test Case Prioritization Technique based on Requirements

Kumar et al. (2013) introduced a hierarchical test case prioritization approach based on requirements along with some other factors containing implementation and test case complexity. There are three levels of prioritization used in this technique. In the first level, priority is assigned to each requirement using 12 factors including customer assigned priority, developer assigned priority, requirement volatility, fault proneness, expected fault, implementation complexity, execution frequency, traceability, show stopper requirement, penalty, cost and time. Requirement with higher value of these factors is assigned high priority. Prioritized list of requirements is obtained at this level. In the second level, mapping is drawn between each requirement and its corresponding modules. In case of more than one corresponding modules, the modules are prioritized based on cyclomatic complexity and non dc paths. Module with higher cyclomatic complexity and non dc paths is given the highest priority. In the last level, the test cases are prioritized by means of 4 factors including test impact, test case complexity, requirement coverage and dependency. The resulting test suite is the prioritized test suite.

2.1.3 Analysis and Comparison

Following table provides a comparison of single criterion based prioritization techniques along with their APFD values (Henard et al., 2016).

Table 2-1: Comparison of Single criterion based Prioritization Techniques

Techniques	Category	Information required	APFD
Total Function	White box	Function Coverage	69%
Additional Function	White box	Uncovered Functions	87%
Total Statement	White box	Statement Coverage	67%
Additional Statement	White box	Uncovered Statements	89%
Total Branches	White box	Branches Covered	67%
Additional Branches	White box	Uncovered Branches	90%
Total FEP	White box	Fault exposing Probability	67%

Additional FEP	White box	Confidence factor	90%
Additional Spanning Statements	White box	Spanning entities and their coverage	89%
Additional Spanning Branches	White box	Spanning entities and their additional coverage	90%
History Based	White box	Exec. history, APFD and program entities covered	Not Given
t-wise	Black box	Event sequences	87%
Requirements Clustering based	Black box	Customer assigned Req. priority, code complexity, modification information	Not Given
PORT	Black box	PFV and WP	47%
History Based	Black box	Relation matrix and run time information	Not Given
HSTP	Black box	12 Factors value, cyclomatic complexity, non dc paths, test impact, test case complexity, requirement coverage and dependency	67%

All the existing white box and black box prioritization techniques have been developed to increase the fault detection rate by prioritizing the test cases. But in all these prioritization techniques, test case is selected randomly in case of a tie in more than one test case. It may possibly occur that the test case which is covering more entities is removed, which results in losing desired code coverage and decreasing the fault detection rate to much extent.

Above table shows that Additional Branch Coverage , Additional FEP and Additional Spanning Branches have yielded the highest rate of fault detection among all white box prioritization techniques as well as black box prioritization techniques. These techniques are showing much better results than all other.

This table also shows that White box prioritization techniques are better than Black box prioritization techniques because these are showing higher values of APFD. We have included Black box prioritization techniques in our literature survey for comparison with White box

prioritization techniques to check which one is better. Among the white box prioritization techniques, Additional strategy based techniques are atleast 18% better than the Total strategy based technique because these techniques have high values of APFD.

2.2 Multiple Criteria based prioritization techniques:

Regression testing can be improved by using more than one criteria for prioritization of testing. Following are different multiple criteria based prioritization techniques from the literature.

Ahmed et al. (2012) proposed a multi criteria based prioritization technique which uses Genetic Algorithm. It uses multi-criteria fitness function for ordering of the test cases. This technique uses three criteria; statement coverage, fault severity and control-flow coverage. Coverage data each test case for statements, conditions and multiple conditions is obtained from control flow coverage metrics. Number of faults exposed and their severity is used to assign weight to each metric. This technique shows better result than the previous approaches.

Prakash and Gomathi (2014) carried out an empirical study to prioritize test cases with multiple criteria. Statement coverage, function coverage, path coverage, branch coverage and fault coverage are five coverage criteria used by this technique. Firstly for each test case, each coverage criteria is calculated then weight is assigned to each test case for each coverage criteria. Value of weight is from 1 to 10. After that, overall weight called as test case weight is calculated for each test case by taking average of weight of each coverage criteria. Finally, test case weight value is used to prioritize the test case. Test case with maximum weight value is given highest priority and so on. They have used three standard applications (Hospital Management system, Library Management system, Student Registration system) for proposed test case prioritization method and make a comparison with present prioritization techniques. This technique shows better result than the previous techniques.

Kaur and Mahajan (2014) carried out an empirical study to prioritize test cases by means of more than one criterion. Statement coverage, path coverage, branch coverage and fault coverage are four coverage criteria used in this technique. Firstly for each test case, each coverage criteria is calculated. After that, for each test case, the coverage detail of each criterion is combined in a table. Finally test cases are sorted on the basis of covering maximum number of criterion entities. They have used the bank application for their proposed method. The results of this study show

that proposed method is more effective than the existing method. The main goal of this article is to make comparison of the effectiveness of un-prioritized and ordered test cases using APFD, APSD, APBD and APPD.

Khasragi (2016) gives an overview of regression testing and prioritization technique. A survey is conducted on different multi-objective test case prioritization techniques. These techniques are using different objectives as criterion including statement coverage, code coverage, execution time, test cost, fault severity, client priority etc. Effectiveness of test cases is evaluated using APFD, TSFD, ASFD and APBC metrics. Through an example, working of single criterion and multiple criteria is discussed. Moreover, single criterion and multiple criteria based prioritization techniques are compared.

The decision of this paper shows that multiple criteria based prioritization techniques are better than single criterion based techniques in terms of rate of fault detection.

In above survey paper, 14 different multiple objective based prioritization techniques are discussed. But we are considering only three of them. The reason is that these existing techniques are considering some criterions which are not coverage related. Most of these techniques are using execution time and cost. Our focus in this thesis is effectiveness of test cases not time. That is why we are not considering those existing techniques which considers execution time or cost.

2.2.1 Analysis and Comparison

All existing prioritization techniques are using some criteria to prioritize test cases. Prioritized list of test cases is generated through these techniques. To check the quality, these techniques are evaluated on some faulty programs. Mutation testing is used to inject faults in the programs. These faults are called as mutants. Mutants are the representative of real faults. Hand seeded faults can be hazardous for validity of results (Do and Rothermel, 2005). Using mutation testing, following table illustrates the comparison of multiple criteria based prioritization techniques along with their APFD values (Henard et al., 2016).

Table 2-2: Comparison of Multiple criteria based Prioritization Techniques

Publication Year	Authors	Criteria's Covered	APFD	Issues
2012	Amr Abdel Fatah Ahmed,Dr.Mohamed Shaheen,Dr.Essam Kosba	Control-flow coverage, Statement coverage, Fault severity	86.60%	Total strategy, Requires rate of fault detection for calculating fitness function, Use of non-deterministic algorithm
2014	N.Prakash, K.Gomathi	Code coverage, Branch coverage, Function coverage, Path coverage, Fault coverage	Not Given	Total Strategy, APFD is not calculated
2014	Navleen Kaur, Manish Mahajan	Statement coverage, Fault coverage, Path coverage, Branch coverage	84.87%	Total Strategy, Coverage of criteria are not normalized

Above table shows that technique proposed by Ahmed et al. (2012) has highest rate of fault detection among all other multiple criteria based prioritization techniques. The fault detection rate is also increased by combining more than one criteria. Total Statement coverage and Total Branch coverage have 67% APFD which is increased by combining these criterion with some other criteria. The technique proposed by Ahmed et al. (2012) has 86.60% APFD because it combines the statement coverage with two more criteria. Also, the technique proposed by Kaur et al. (2014) has 84.87% APFD because it combines the statement coverage and branch coverage with two more criteria. This table shows that regardless of Total strategy used by these techniques, multiple criteria based prioritization techniques are generally better than single criterion based prioritization techniques.

The technique proposed by Ahmed et al. (2012) uses genetic algorithm for prioritization. The weakness of this technique is that rate of fault detection (RFT) is required for calculation of fitness function and during prioritization, fault detection data for each test case is not available. The actual number of faults in a software are unknown during prioritization. Only coverage data is available for prioritization. Another weakness is that genetic algorithm is non-deterministic,

and do not guarantee that the prioritize list will be generated in the same way in same amount of time in every execution. Due to these weaknesses, we can not compare our proposed technique with this technique.

The techniques proposed by Prakash et al. (2014) and Kaur et al. (2014) are not appropriate, as these techniques are prioritizing the test cases using Total strategy. The Total strategy does not count the number of additional entities which remain uncovered in the previous selection. It only counts the numbers of entities covered by each test case. There can be more than one test case that covers the same entities. So, there is repetition of the entities covered which is not desirable. Total strategy can also skip some entities in case of halting testing prematurely, for example, if a test case covers only one entity which is not covered by any other test case and lowest priority is assigned to that test case. These two techniques are combining the coverage of all criterion so “Additional” strategy can not be applied on all criterion at the same time.

The coverage of criterions is not normalized in the technique proposed by Kaur et al. (2014). Coverage criterion with maximum entities will automatically get higher priority than other. As in general, a program have maximum statements but few branches and if a test case is covering maximum statements but few branches, it will get high priority and test case covering maximum branches and few statements will get low priority. So, statement coverage gets high priority than branch coverage and prioritization becomes statement based. The technique proposed by Prakash and Gomathi (2014) is better than the technique proposed by Kaur et al. (2014). In this technique, the coverage of criteria is normalized. Such that each criterion will get equal priority. This paper is incomplete in as it is not calculating the Average Percentage of Fault detection (APFD).

We will compare our proposed multiple criteria based prioritization algorithm with the prioritization technique of Prakash and Gomathi (2014), as this technique is better than other. Also, we will make a comparison with the prioritization technique of Kaur et al. (2014).

Chapter 3 : Proposed Approach

In literature survey, we have identified the most relevant and most commonly used white box and black box prioritization techniques. After studying various prioritization techniques, we have reached the conclusion that most of these techniques are single criterion based. In single criterion based prioritization, if two or more than two test cases have same coverage value then test case is selected arbitrarily. These techniques ignore the fact that may be the removed test case has better coverage than the selected one. The test case with the better ability to find the faults may have been given the lowest priority by these techniques. Multiple criteria based prioritization techniques are studied less as compared to single criteria based prioritization techniques. All the existing multiple criteria based prioritization techniques are selecting test cases using “Total” approach, which is not appropriate as it is not selecting test cases on the basis of number of uncovered entities.

To overcome the gap in existing techniques, we have proposed a new technique that will rank the test cases based on multiple criteria using “Additional” strategy. We have developed a new prioritization algorithm which uses two coverage criteria to order the test cases. Test suite and the coverage information of both criteria are inputs of proposed prioritization algorithm. One criterion will be considered as primary and other will be considered as secondary criteria. Initially, the test cases will be prioritized using primary criterion. Secondary criterion will be used to break the tie among test cases when two or more test cases provide equal coverage of entities of first criterion. The output of this algorithm is prioritized list of test cases.

3.1 Proposed Prioritization Algorithm

Context diagram of our proposed solution is shown in figure 3.1.

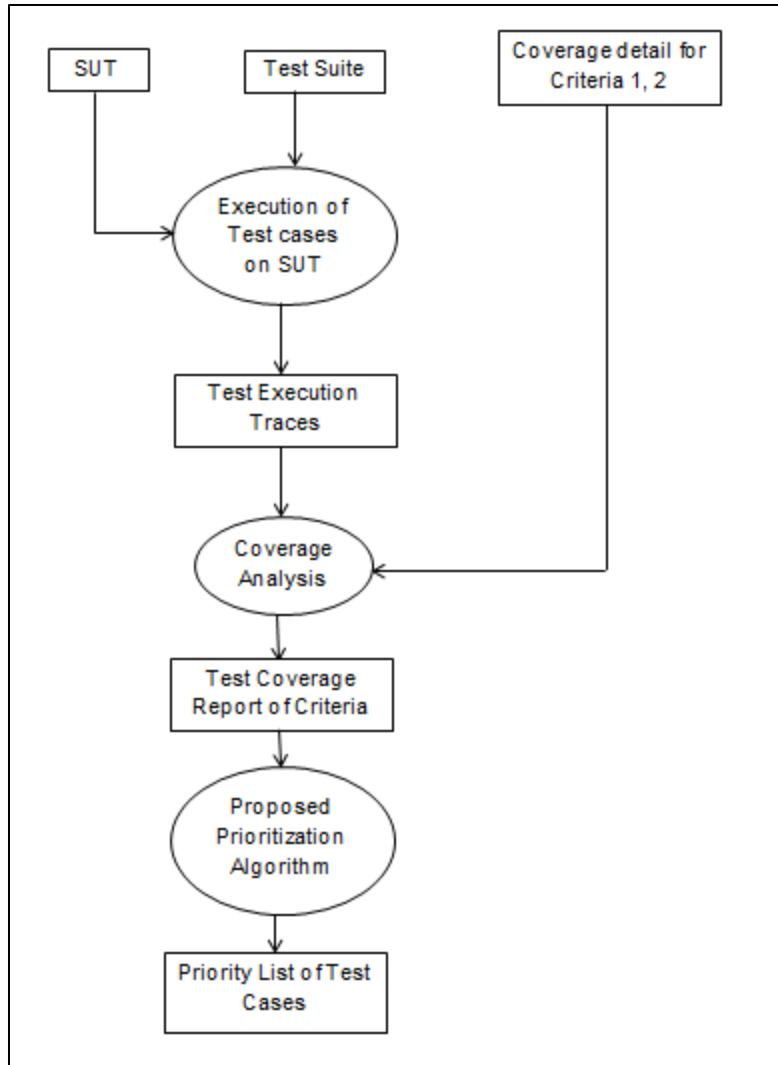


Figure 3-1: Context diagram of proposed Prioritization Algorithm

When a system is developed, it undergoes testing. Test suite which consists of test cases is formed for testing the system. This test suite is run on SUT (System under test) to collect the coverage data. When a specific test case runs on SUT, the amount to which source code of program is executed is the coverage of that SUT. Different criteria are used to measure the coverage of code like Statement coverage, Function coverage, Branch coverage etc. These criteria are basically the conditions which need to be satisfied by the test suite. For our approach, we require the mapping between test cases and coverage criteria. This mapping shows which test case covers which entity of coverage criteria. We use the supposition that we have this mapping. This mapping will be used as input to our prioritization algorithm and it will generate output which is the priority list of test cases.

3.1.1 Multiple Criteria based Prioritization Algorithm

Multiple Criteria based Prioritization Algorithm will use the mapped data of test cases and coverage criteria as input and generate a priority list of test cases as output. Figure 3.2 explains the input, output and procedure of this algorithm.

Our algorithm takes two criteria as input. One is considered as primary and other is considered as secondary. Test cases will be prioritized according to primary criterion. Secondary criterion is applied in case of tie in more than one test case. The inputs of algorithm are: test suite T for program P which consists of test case. $EntitiesCov1$ is the set of entities of first criteria having “ m ” number of entities. $EntitiesCov2$ is the set of entities of second criteria having “ n ” number of entities. $Cov1(t)$ is the set of entities of first criterion covered by each test case. $Cov2(t)$ is the set of entities of second criterion covered by each test case. the output of algorithm is PrT , which is the priority list of test cases. Declarations of algorithm are X and Y which consists of set of test cases.

In step 1, the algorithm will find a test case from test suite T that covers the maximum entities of set $EntitiesCov1$. If more than one test case covers same number of entities, a tie occurs. This tie is broken by using the set $EntitiesCov2$. Test case which covers maximum entities of the set $EntitiesCov2$ is selected from these test cases. In step 2, the selected test case will be added to the priority list PrT and information of the sets $EntitiesCov1$ and $EntitiesCov2$ will be updated to show only the uncovered entities. Test suite T will also be updated. In step 3, the residual coverage $resCov1(t)$ of each test case will be calculated while the test suite T does not get empty. In step 3.2, residual coverage of each test case will be compared to select test case with minimum residual coverage value. If more than one test case in step 3.2 has same residual coverage value, a tie occurs. This tie is broken by calculating the residual coverage $resCov2(t)$ of these test cases using secondary criterion. Test case with minimum residual coverage $resCov2(t)$ is selected from these test cases. In step 3.3, the selected test case will be added to the priority list PrT . Information of the sets $EntitiesCov1$ and $EntitiesCov2$ will be updated to show only the uncovered entities. Test T will also be updated. In step 4, when all entities of both sets are covered, then append any remaining test case in the priority list PrT in random order.

Procedure for prioritizing regression test cases

Input:

1. T : Test Suite for P'
2. $entitiesCov1$: Set of entities of criterion1 in P covered by tests in T .
3. $entitiesCov2$: Set of entities of criterion2 in P covered by tests in T .
4. $cov1(t)$: Set of entities of criterion1 covered by executing P against t .
5. $cov2(t)$: Set of entities of criterion2 covered by executing P against t .

Output:

PrT : A sequence of test cases

Declaration:

X' : Set of Test cases, Y : Set of test cases

Procedure: $PrTest$

Step 1: $X'=T$. Find t in X' such that

If $Cov1(t) > Cov1(u)$ for all u in X' , $u \neq t$. Goto step 2

Else

If $Cov1(t) = Cov1(u)$ for some u in X' , $u \neq t$. Set $Y = \langle t, u \rangle$ find t in Y such that $Cov2(t) \geq Cov2(u)$ for all u in Y , $u \neq t$.

Step 2: Set $PrT = \langle t \rangle$, $X' = X' \setminus \{t\}$, $entitiesCov1 = entitiesCov1 \setminus cov1(t)$, $entitiesCov2 = entitiesCov2 \setminus cov2(t)$, $Y = \emptyset$.

Step 3: Repeat while $X' \neq \emptyset$.

Step 3.1: Compute residual coverage for each test t in X' ; $resCov1(t) = entitiesCov1 \setminus [cov1(t) \cap entitiesCov1]$

Step 3.2: If $resCov1(t) > resCov1(u)$ for all u in X' , $u \neq t$. Goto step 3.3

Else

If $resCov1(t) = resCov1(u)$ for some u in X' , $u \neq t$. Set $Y = \langle t, u \rangle$ find t in Y such that $resCov2(t) \geq resCov2(u)$ for all u in Y , $u \neq t$.

Step 3.3: Set $PrT = append(PrT, t)$, $X' = X' \setminus \{t\}$, $entitiesCov2 = entitiesCov2 \setminus cov2(t)$,

$entitiesCov1 = entitiesCov1 \setminus cov1(t)$.

Step 4: Append to PrT any remaining tests in X' in random order.

Figure 3-2: Multiple Criteria based Prioritization Algorithm

3.1.2 Example of Proposed Prioritization Algorithm

The following example is used to define the concepts of our proposed multiple criteria based prioritization algorithm. Table 3.1 shows the mapping between test cases and coverage data.

$T = \{t1, t2, t3, t4, t5\}$

$EntitiesCov1 = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$

$EntitiesCov2 = \{1, 2, 3, 4\}$

Table 3-1: Example Data

Test case(t)	Entities of Criterion 1	Cov1(t)	Entities of Criterion 2	Cov2(t)
t1	1,2,3,4,5,6	6	1,2	2
t2	4,6,8,9	4	2,4	2
t3	1,4,8,10	4	4	1
t4	1,2,3,4,8,9	6	2,3,4	3
t5	2,4,7,10	4	2,3	2

Step 1: Test case t1 and t4 have same coverage value. So tie occurs between these test cases. Test case t4 will be selected according to coverage of criterion 2 as it covers maximum entities of criterion 2.

Step 2: Test case selected in step 1 will be added to priority list and remaining test suite will be prioritized. EntitiesCov1 and EntitiesCov2 will also be updated to show the remaining uncovered entities.

$PrT = \{t4\}$

$T = \{t1, t2, t3, t5\}$

$EntitiesCov1 = \{5, 6, 7, 10\}$

$EntitiesCov2 = \{1\}$

Step 3: Calculating residual coverage of all remaining test cases while test suite T does not get empty.

The residual coverage of each test case using first criterion will be:

$$\begin{aligned} \text{resCov1}(t1) &= \{5,6,7,10\} \setminus [\{5,6,7,10\} \cap \{1,2,3,4,5,6\}] \\ &= 2 \end{aligned}$$

$$\begin{aligned} \text{resCov1}(t2) &= \{5,6,7,10\} \setminus [\{5,6,7,10\} \cap \{4,8,6,9\}] \\ &= 3 \end{aligned}$$

$$\begin{aligned} \text{resCov1}(t3) &= \{5,6,7,10\} \setminus [\{5,6,7,10\} \cap \{1,4,8,10\}] \\ &= 3 \end{aligned}$$

$$\begin{aligned} \text{resCov1}(t5) &= \{5,6,7,10\} \setminus [\{5,6,7,10\} \cap \{2,4,7,10\}] \\ &= 2 \end{aligned}$$

Test case t1 and test case t5 have same value of residual coverage, so tie occurs between these two test cases. This tie will be broken by calculating the residual coverage of t1 and t5 according to second criterion.

The residual coverage of test case t1 and t5 will be:

$$\begin{aligned} \text{resCov2}(t1) &= \{1\} \setminus [\{1\} \cap \{1,2\}] \\ &= 0 \end{aligned}$$

$$\begin{aligned} \text{resCov2}(t5) &= \{1\} \setminus [\{1\} \cap \{2,3\}] \\ &= 1 \end{aligned}$$

Test case t1 will be selected as it has minimum value of residual coverage. This test case will be added to priority list and test suite T will be updated. EntitiesCov1 and EntitiesCov2 will also be updated to show the remaining uncovered entities.

$$\text{PrT} = \{t4, t1\}$$

$$\text{T} = \{t2, t3, t5\}$$

$$\text{EntitiesCov1} = \{7, 10\}$$

EntitiesCov2= { }

This step is repeated until all test cases will be prioritized and all the entities of both criterions will be covered by atleast one test case.

Step 4:The final priority list will be:

Multiple criteria based priority list (PrT) = {t4, t1, t5, t2, t3}

In this thesis, we will use Branch coverage and statement coverage data for implementation.

Branch coverage is our primary criterion and statement coverage is secondary criterion. We have selected Branch and Statement coverage data because the literature survey which is conducted in chapter 2 depicts that these techniques are showing much better results than other. After implementation, we will compare our algorithm with existing multiple criteria based test case prioritization techniques and also with the strongest single criteria based prioritization approach called branch coverage prioritization approach.

Chapter 4 : Implementation

In this chapter we will discuss the implementation details of our proposed algorithm. We have implemented our algorithm in java language using eclipse software.

4.1 Implementation details

This section describes the classes and the methods used in our code. Code includes two classes. First class is for prioritization of test cases, and the second class handles the tie case between more than one test cases. Figure 4.1 illustrates the Class diagram of our code.

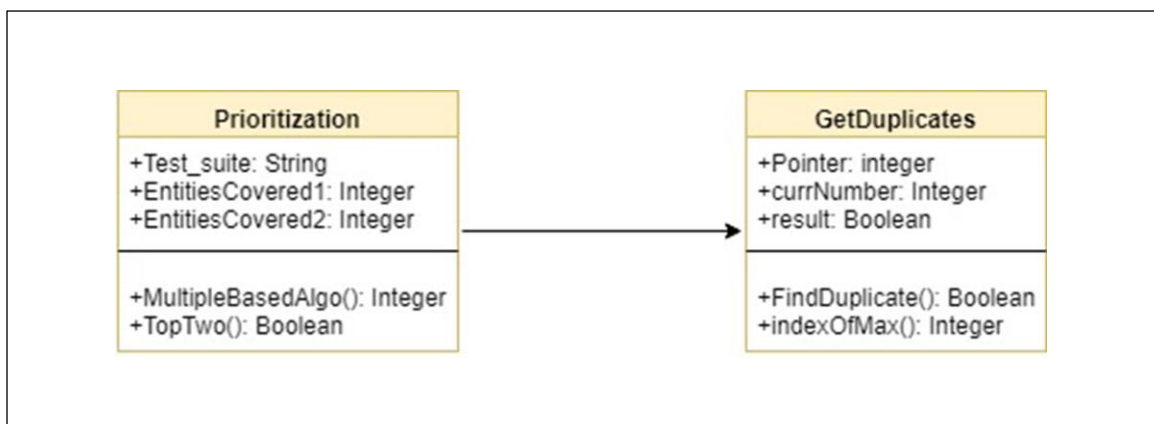


Figure 4-1: Class Diagram of proposed algorithm

- **Prioritization class:** This class takes the entire coverage and test suite data as input. The complete process of prioritization is carried out in this class.
- **GetDuplicates class:** This class handles the duplicate case. It checks the tie between test cases and if more than one test case have same coverage, then this tie is handled by this class.

Three methods are used in in the above defined classes. Following is the detail of these methods.

- **MultipleBasedAlgo ():** This method performs core functionality. The return type of this method is integer type array list. The parameters passed to this method are the array list of entities and coverage of these entities for each test case. This method returns the final priority list of test cases.
- **TopTwo ():**The return type of this method is Boolean. Integer type array list which includes the coverage factor of each test case is passed as parameter to this method.

Coverage factor is the number of entities covered by each test case. If more than one test case has same coverage factor of primary criterion then this function returns true value and then priority is set using secondary criterion.

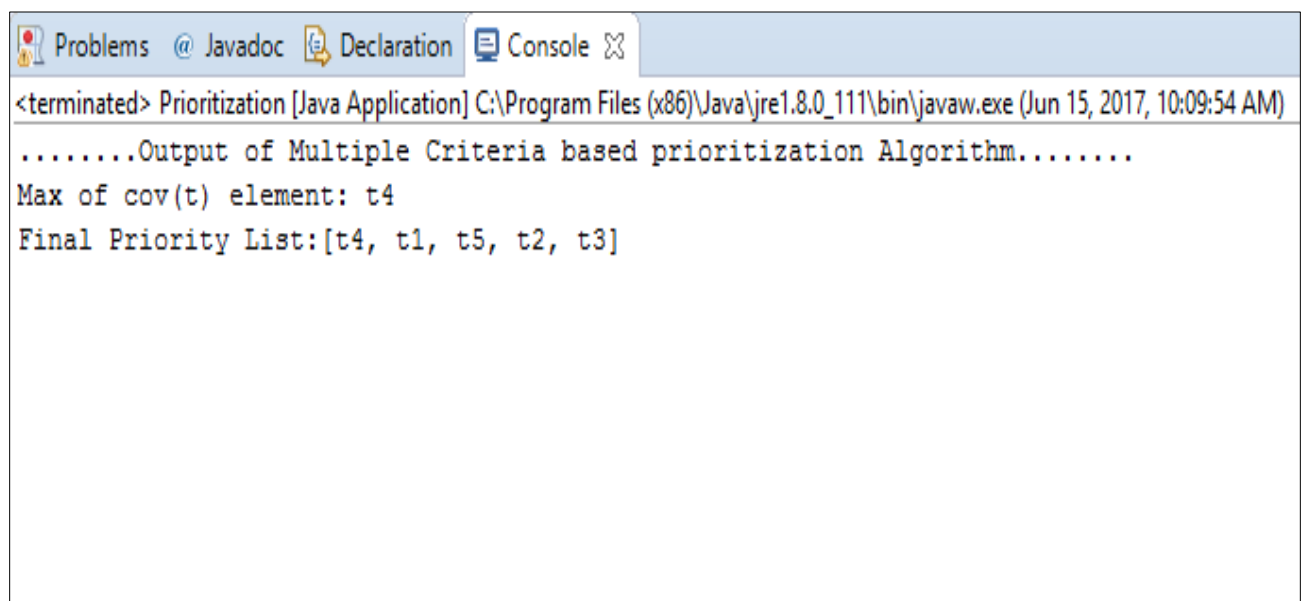
- **FindDuplicate ()**: The return type of this function is Boolean. Integer type array list which includes the Residual coverage of each test case is passed as parameter to this method. Residual coverage indicates the count of currently uncovered entities (Branches, Statements, functions etc.) that will remain uncovered in the previous selection. This function returns true if more than one test case have same value of residual coverage for primary criterion. Then residual coverage of secondary criterion will be used to prioritize the test cases.
- **indexOfMax()**: The return type of this function is integer type array list. It provides the index of those test cases among which tie occurs.

Initially, mapping of test cases to coverage criteria is initialized. The size of each test case for primary criterion and secondary criterion is calculated. **TopTwo ()** checks the size of each test case and if selects the test case which has maximum size according to primary criterion. In case of more than one test cases with same size, the function **indexOfMax()** is called to check the index of tie test cases. To break this tie, size of these test cases in secondary criterion is checked and test case with maximum size is selected using function **MultipleBasedAlgo ()**. Afterwards, residual coverage of remaining test case for primary criterion is calculated to show the additional entities covered by these test cases. **FindDuplicate ()** selects the test case with minimum residual coverage and if more than one test cases have same residual coverage value, then the function **indexOfMax()** is called to check the index of tie test cases. To break this tie, residual coverage of these test cases is calculated using secondary criterion and test case with minimum residual coverage value is selected using function **MultipleBasedAlgo ()**. In case of a tie using secondary criterion, test cases are selected in random order.

.4.2 Usage Details

Our algorithm takes the input of test suite and mapping of test cases to coverage criteria from a file. The extension of file can be .txt or .data. This file includes the coverage of two criterion. One criterion is set as primary and other as secondary. Using this information, our algorithm

which is discussed in detail in previous chapter will assign priorities to each test case. The algorithm will select the test case which is covering maximum entities of primary criterion. If there exists more than one such test case, then secondary criterion will be used to select that test case which covers maximum entities of secondary criterion. Selected test case will be given highest priority. Afterwards, the coverage information will be updated to indicate entities which remain uncovered. The algorithm will then find the residual coverage of all test cases according to primary criterion. In case if more than one test case with same residual coverage exists, then residual coverage is calculated for each test case using secondary criterion and test case with maximum value of residual coverage will be selected and assigned priority. Our algorithm generates multiple criteria based priority list. This output is displayed on console. Following figure shows the output of an example program on console.



```
<terminated> Prioritization [Java Application] C:\Program Files (x86)\Java\jre1.8.0_111\bin\javaw.exe (Jun 15, 2017, 10:09:54 AM)
.....Output of Multiple Criteria based prioritization Algorithm.....
Max of cov(t) element: t4
Final Priority List:[t4, t1, t5, t2, t3]
```

Figure 4-2: Output of example program on console

Chapter 5 : Results and Discussion

This chapter includes the results of experiments' done on different subject programs and also these results are discussed in detail. We have collected the coverage data of statements and branches for three different subject programs. Coverage data of statements and branches is given in Appendix B. The test suite for each program is created by Worst case boundary value analysis. Using these coverage data, we have generated the priority lists for each subject program using the algorithm discussed in chapter 3. This algorithm generates a priority list based on multiple criteria. We have compared our technique with the two existing multiple criteria based prioritization techniques proposed by Prakash et al. (2014) and Kaur et al. (2014), also with strongest single criteria based prioritization technique called Additional Branch Coverage. For comparison, we have also implemented the existing techniques and generated a priority list for these techniques. Average Percentage of fault detection (APFD) metric is used to compare these techniques.

To test our technique, we have used three subject programs. Faults are seeded in the subject programs by using mutation operator AORB (Arithmetic Operator Replacement Binary), as the mutation faults are the representative of real faults. Hand seeded faults can be problematic for validity of results (Do et al., 2005). Afterwards, mutants of each subject program are created. Using these mutants, a mapping is formed between test cases and faults seeded, which shows that which test case detects which fault. This mapped data is given in Appendix B. Using this mapping and priority lists; we have calculated the Average Percentage of Fault Detection (APFD) of each technique for each subject program. APFD values are used to perform a comparison between these techniques.

5.1 Subject programs

In our experiments, we have used three subject programs: Triangle problem, Commission problem and Date problem for testing.

Triangle program (Jorgensen, 2010) takes three inputs which are the sides of a triangle. Using these three inputs, the program checks whether it is scalene, isosceles, and equilateral or not a

triangle. If these three values are valid and equal, then triangle is equilateral. If these values are valid and two of them are equal, then triangle is isosceles. If these values are valid and unequal, then triangle is scalene. And if any one of the values is invalid, then it is not a triangle.

Next Date program (Jorgensen, 2010) takes three inputs: day, month, year. Using these values, it gives the date of next day as output. The date which is given as input to program is treated as current date. This date is checked according to Gregorian calendar that whether the given date is valid. It also considers the rules of leap year.

Commission program (Jorgensen, 2010) is based on three inputs: locks, barrels and stocks. These values are used to calculate the total sales. It calculates the total commission according to different values of sales.

Source codes of these programs are given in Appendix A. The features of these programs are given below in table 5-1.

Table 5-1: Subject Programs summary

Program	Lines of Code	No. of inputs	No. of decisions	No. of mutants using AORB	No. of Statements	No. of branches
Triangle Problem	42	3	13	48	35	26
Commission problem	40	3	2	68	29	4
Date problem	72	3	20	40	35	40

5.2 Priority Lists

Subject programs are used to generate the priority lists. We have generated priority lists for our Proposed multiple criteria based prioritization technique: Branch-Statement Coverage, existing single criteria based prioritization technique: Branch Coverage and existing multiple criteria based prioritization techniques: Prakash and Gomathi (2014) and Kaur et al. (2014). These

prioritization lists for Triangle problem, Commission problem, Next Date problem are given in Table 5.2, Table 5.3, and Table 5.4.

Table 5-2: Priority Lists for Triangle Problem

Branch-Statement coverage based priority list	Branch coverage based priority list	N.Prakash and K.Gomathi,2014	N. Kaur et al. ,2014
{t56, t11, t26, t51, t1, t32, t71, t101, t2, t3, t4, t5, t6, t7, t8, t9, t10, t12, t13, t14, t15, t16, t17, t18, t19, t20, t21,t22, t23, t24, t25, t27, t28, t29, t30, t31, t33, t34, t35, t36, t37, t38, t39, t40, t41, t42, t43, t44, t45, t46, t47, t48, t49, t50, t52, t53, t54, t55, t57, t58, t59, t60, t61, t62, t63, t64, t65, t66, t67, t68, t69, t70, t72, t73, t74, t75, t76, t77, t78, t79, t80, t81, t82, t83, t84, t85, t86, t87, t88, t89, t90, t91, t92, t93, t94, t95, t96, t97, t98, t99, t100, t102, t103, t104, t105, t106, t107, t108, t109, t110, t111, t112, t113, t114, t115, t116, t117, t118, t119, t120, t121, t122, t123, t124, t125}	{t6, t11, t26, t51, t1, t32, t71, t101, t2, t3, t4, t5, t7, t8, t9, t10, t12, t13, t14, t15, t16, t17, t18, t19, t20, t21, t22, t23, t24, t25, t27, t28, t29, t30, t31, t33, t34, t35, t36, t37, t38, t39, t40, t41, t42, t43, t44, t45, t46, t47, t48, t49, t50, t52, t53, t54, t55, t56, t57, t58, t59, t60, t61, t62, t63, t64, t65, t66, t67, t68, t69, t70, t72, t73, t74, t75, t76, t77, t78, t79, t80, t81, t82, t83, t84, t85, t86, t87, t88, t89, t90, t91, t92, t93, t94, t95, t96, t97, t98, t99, t100, t102, t103, t104, t105, t106, t107, t108, t109, t110, t111, t112, t113, t114, t115, t116, t117, t118, t119, t120, t121, t122, t123, t124, t125}	{t56, t87, t6, t24, t31, t37, t62, t74, t81, t99, t106, t112, t124, t12, t49, t113, t51, t61, t66, t71, t77, t82, t92, t97, t103, t108, t118,t123, t11, t16, t17, t18, t21, t22, t23, t32, t36, t38, t41, t42, t43, t44, t46, t47, t48, t50, t52, t63, t67, t68, t69, t70, t72, t73, t75, t76, t78, t83, t91, t93, t94, t95, t96, t98, t100, t102, t107, t109, t114, t115, t116, t117, t119, t120, t121, t122, t125, t26, t27, t28, t29, t30, t57, t58, t59, t60,t86, t88, t89, t90, t101, t1, t2, t3, t4, t5, t7, t8, t9, t10, t13, t14, t15, t19,t20, t25, t33, t34, t35, t39, t40, t45, t53, t54, t55, t64, t65, t79, t80, t84, t85, t104, t105, t110}	{t56, t87, t6, t24, t31, t37, t62, t74, t81, t99, t106, t112, 124, t12, t49, t113, t61, t66, t71, t77, t82, t92,t103, t108, t118, t123, t11, t16, t17, t18, t21, t22, t23, t32, t36, t38, t41, t42, t43, t44, t46, t47, t48, t50, t51, t52, t63, t67, t68, t69, t70, t72, t73, t75, t76, t78, t83, t91, t93, t94, t95, t96, t97, t98, t100, t102, t107, t109, t111, t114, t115, t116, 117, t119, t120, t121, t122, t125, t26, t27, t28, t29, t30, t57, t58, t59, t60, t86, t88, t89,t90, t101, t1, t2, t3, t4, t5, t7, t8, t9, t10, t13, t14, t15, t19, t20, t25, t33, t34, t35, t39, t40, t45, t53, t54, t55, t64, t65, t79, t80, t84, t85, t104, t105, t110 }

Table 5-3: Priority Lists for Commission program

Branch-Statement coverage based priority list	Branch coverage based priority list	N.Prakash and K.Gomathi,2014	N. Kaur et al. ,2014
{t1, t4, t2, t3, t5, t6, t7, t8, t9, t10, t11, t12, t13, t14, t15, t16, t17, t18, t19, t20, t21, t22, t23, t24, t25, t26, t27, t28, t29, t30, t31, t32, t33, t34, t35, t36, t37, t38, t39, t40, t41, t42, t43, t44, t45, t46, t47, t48, t49, t50, t51, t52, t53, t54, t55, t56, t57, t58, t59, t60, t61, t62, t63, t64, t65, t66, t67, t68, t69, t70, t71, t72, t73, t74, t75, t76, t77,t78, t79, t80, t81, t82, t83, t84, t85, t86, t87, t88, t89, t90, t91, t92, t93, t94, t95, t96, t97, t98, t99, t100, t101, t102, t103, t104, t105, t106, t107, t108, t109, t110, t111, t112, t113, t114, t115, t116, t117, t118, t119, t120, t121, t122, t123, t124, t125}	{t1,t5, t2, t4, t3, t6, t7, t8, t9, t10, t11, t12, t13, t14, t15, t16, t17, t18, t19, t20, t21, t22, t23, t24, t25, t26, t27, t28, t29, t30, t31, t32, t33, t34, t35, t36, t37, t38, t39, t40, t41, t42, t43, t44, t45, t46, t47, t48, t49, t50, t51, t52, t53, t54, t55, t56, t57, t58, t59, t60, t61, t62, t63, t64, t65, t66, t67, t68, t69, t70, t71, t72, t73, t74, t75, t76, t77, t78, t79, t80, t81, t82, t83, t84, t85, t86, t87, t88, t89, t90, t91, t92, t93, t94, t95, t96, t97, t98, t99, t100, t101, t102, t103, t104, t105, t106, t107, t108, t109, t110, t111, t112, t113, t114, t115, t116, t117, t118, t119, t120, t121, t122,	{ t1, t2, t3, t6, t7, t8, t11, t12, t26, t27, t28, t31, t32, t33, t36, t4, t5,t9, t10, t13, t14, t15, t16, t17, t18, t19, t20, t21, t22, t23 t24, t25,t29, t30, t34, t35, t37, t38, t39, t40, t41, t42, t43, t44, t45, t46, t47, t48, t49, t50, t51, t52, t53, t54, t55, t56, t57, t58, t59, t60, t61, t62, t63, t64, t65, t66, t67, t68, t69, t70, t71, t72, t73, t74, t75, t76, t77, t78, t79, t80, t81, t82, t83, t84, t85, t86, t87, t88, t89, t90, t91, t92, t93, t94, t95, t96, t97, t98, t99, t100, t101,t102, t103, t104,t105, t106, t107, t108, t109, t110, t111, t112, t113, t114, t115, t116, t117, t118, t119, t120, t121, t122, t123, t124, t125, t120, t121, t122, t123, t124, t125}	{t1, t2, t3, t6, t7, t8, t11, t12, t26, t27, t28, t31, t32, t33, t36, t4, t5, t9, t10, t13, t14, t15, t16, t17, t18, t19, t20,t21, t22, t23, t24, t25, t29, t30, t34, t35, t37, t38, t39, t40, t41, t42, t43, t44, t45, t46, t47, t48, t49, t50, t51, t52, t53, t54, t55, t56, t57,t58, t59, t60, t61, t62, t63, t64, t65, t66, t67, t68, t69, t70, t71, t72, t73, t74, t75, t76, t77, t78, t79, t80, t81, t82, t83, t84, t85, t86, t87, t88, t89,t90, t91, t92, t93, t94, t95, t96, t97, t98, t99, t100, t101, t102, t103, t104, t105, t106, t107, t108, t109, t110, t111, t112, t113, t114, t115, t116, t117, t118, t119, t120, t121, t122, t123, t124, t125 }

Table 5-4: Priority Lists for Next Date program

Branch-Statement coverage based priority list	Branch coverage based priority list	N.Prakash and K.Gomathi,2014	N. Kaur et al. ,2014
{t107, t1, t11, t21, t81, t6, t16, t82, t106, t2, t3, t4, t85, t5, t7, t8, t9, t10, t12, t13, t14, t15, t17, t18, t19, t20, t22, t23, t24, t25, t26, t27, t28, t29, t30, t31, t32, t33, t34, t35, t36, t37, t38, t39, t40, t41, t42, t43,t44, t45, t46, t47, t48, t49, t50, t51, t52, t53, t54, t55, t56, t57, t58, t59, t60, t61, t62, t63, t64, t65, t66, t67, t68, t69, t70, t71, t72, t73, t74, t75, t76, t77, t78, t79, t80, t83, t84, t86, t87, t88, t89, t90, t91,t92,t93, t94, t95, t96, t97, t98, t99, t100,t101, t102, t103, t104, t105,t108, t109, t110, t111, t112,t113, t114, t115, t116, t117, t118, t119, t120,t121, t122, t123, t124, t125}	{t106, t1, t11, t21, t81, t6, t16, t82, t2, t3, t4, t5, t7, t8, t9, t10, t12, t13, t14, t15, t17, t18, t19, t20, t22, t23, t24, t25, t26, t27, t28, t29, t30, t31, t32, t33, t34, t35, t36, t37, t38, t39, t40, t41, t42, t43, t44, t45, t46, t47, t48, t49, t50, t51, t52, t53, t54, t55, t56, t57, t58, t59, t60, t61, t62, t63, t64, t65, t66, t67, t68, t69, t70, t71, t72, t73, t74, t75, t76, t77, t78, t79, t80, t83, t84, t85, t86, t87, t88, t89, t90, t91, t92, t93, t94, t95, t96, t97, t98, t99, t100, t101,t102, t103, t104, t105,t107, t108, t109, t110,t111, t112, t113, t114,t115, t116, t117, t118,t119, t120, t121, t122, t123, t124, t125}	{t107, t110, t106, t108, t109, t81, t82, t83, t85, t84, t7, t10, t32, t35, t57, t60, t6, t8, t9, t31, t33, t34, t56, t58, t59,t22, t25, t47, t50, t72, t75, t97, t122, t125, t21, t23, t24, t46, t73, t74, t96, t98, t99, t17, t20, t42, t45, t67, t70, t92, t95, t117, t120, t16, t18, t19, t41, t43, t44, t66, t68, t69, t91, t93, t94, t116, t118, t119, t112, t115, t37, t40, t62, t65, t87, t90, t112, t115, t11, t13, t14, t36, t38, t39, t61, t63, t64, t86, t88, t89, t111, t113, t114, t5, t27, t30, t52, t55, t77, t102, t105, t1, t2, t3, t4, t26, t28, t29, t51, t53, t54,t76, t78,t79, t80, t101, t103, t104}	{t107, t110, t106, t108, t109, t81, t82, t83, t85, t84, t7, t10, t32, t35, t57, t60, t6, t8, t9, t31, t33, t34, t56, t58, t59, t22, t25, t47, t50, t72, t75, t97, t122, t125, t21, t23, t24, t46, t48, t49, t71, t73, t74, t96, t98, t100, t121, t123, t124, t17, t20, t42, t45, t67, t70, t92, t95, t99, t117, t120, t16, t18, t19, t41, t43, t44, t66, t68, t69, t91, t93, t94, t116, t118, t119, t12, t15, t37, t38, t39, t62, t65, t87, t90, t112, t115, t11, t13, t14, t36, t40, t61, t63, t64, t86, t88, t89, t111, t113, t114, t2, t5, t27, t30, t52, t55, t77, t102, t105, t1, t3, t4, t26, t28, t29, t51, t53, t54, t76, t78, t79, t80, t101, t103, t104 }

5.3 Comparison

The effectiveness of these prioritization techniques is evaluated by the rate of fault detection i.e. Average Percentage of Faults Detected (APFD) which is the measure of how early faults are detected in the testing process by the test suite (Rothermel et al., 1999). Following formula is used to calculate the APFD:

$$APFD = 1 - \frac{TF_1 + \dots \dots TF_m}{nm} + \frac{1}{2n}$$

Where T is the test suite containing n test cases and F is the set of m faults revealed by T. For ordering T', TFi is the order of the first test case that reveals the ith fault Fi. Faults which are not detected by any test case are not considered for APFD calculation. These faults are ignored.

To calculate the APFD metric, faults are seeded in the subject programs by using mutation operator AORB (Arithmetic Operator Replacement Binary). AORB operator replace basic binary arithmetic operators with other binary arithmetic operators.

As an example, consider the triangle problem:

Faults detected:

F= {5, 6, 7, 8, 13, 14, 15, 16, 17, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 38, 39, 40, 41, 42, 43, 44, 46, 47, 48}

Priority List for Branch-Statement Coverage:

T= { t56, t11, t26, t51, t1, t32, t71, t101, t2, t3, t4, t5, t6, t7, t8, t9, t10, t12, t13, t14, t15, t16, t17, t18, t19, t20, t21, t22, t23, t24, t25, t27, t28, t29, t30, t31, t33, t34, t35, t36, t37, t38, t39, t40, t41, t42, t43, t44, t45, t46, t47, t48, t49, t50, t52, t53, t54, t55, t57, t58, t59, t60, t61, t62, t63, t64, t65, t66, t67, t68, t69, t70, t72, t73, t74, t75, t76, t77, t78, t79, t80, t81, t82, t83, t84, t85, t86, t87, t88, t89, t90, t91, t92, t93, t94, t95, t96, t97, t98, t99, t100, t102, t103, t104, t105, t106, t107, t108, t109, t110, t111, t112, t113, t114, t115, t116, t117, t118, t119, t120, t121, t122, t123, t124, t125}

Number of Test cases:

n = 125

Number of Faults:

$$m = 36$$

$$APFD = 1 - \frac{3 + 3 + 3 + 3 + 4 + 4 + 4 + 4 + 1 + 1 + 1 + 1 + 1 + 1 + \dots + 4 + 1 + 1 + 1}{125(36)} + \frac{1}{2(125)}$$

$$APFD = 1 - \frac{244}{4500} + \frac{1}{250}$$

$$APFD = 94.1\%$$

This is the APFD value for Triangle program calculated by using above formula. Similarly, APFDs of Next date program and Commission Program are calculated same as above. APFDs are calculated for proposed technique and the existing techniques to make a comparison between these techniques. The mapping of test cases to faults detected is given in Appendix B.

The APFDs of all techniques for each subject program are given in Table 5.5.

Table 5-5: APFDs of Subject Programs

Subject Program	No. of Test Cases	No. of faults Seeded	No. of faults Detected	Branch-Statement	Branch Coverage	N. Prakash and K. Gomathi, 2014	N. Kaur et al., 2014
Triangle Problem	125	48	36	94.1%	92.51%	71.60%	72.11%
Next Date Problem	125	40	24	92%	86%	73.80%	74.13%
Commission Problem	125	68	58	99.00%	97.35%	95.60%	95.69%

Following figure explains the Graphical representation of APFDs of Subject Programs.

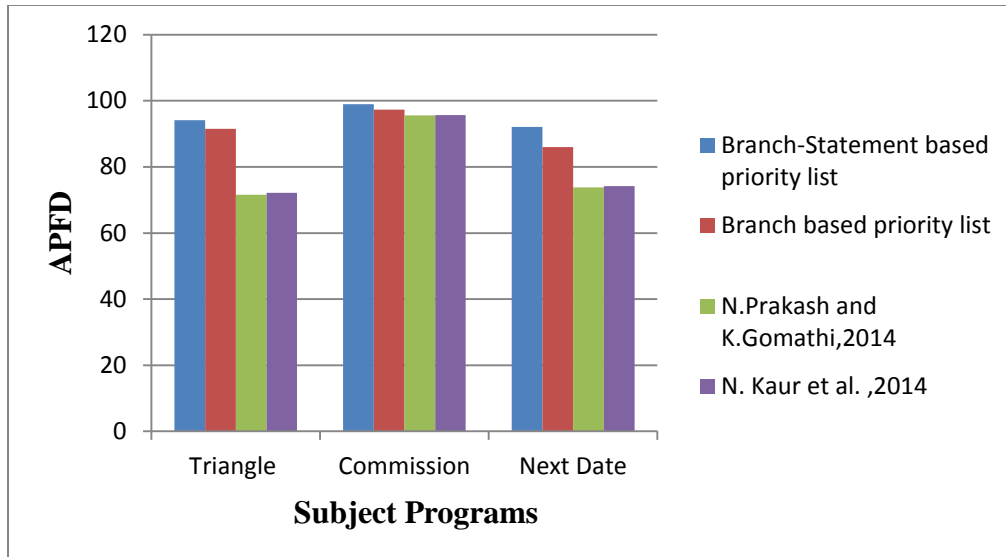


Figure 5-1: Graphical Representation of APFD

From above graph, it is concluded that for all three subject programs, the APFD value of our proposed multiple criteria based prioritization technique (Branch-Statement coverage) is higher than the existing Single criteria based prioritization technique (Branch coverage). The reason is that our technique uses two criterions to prioritize the test cases. Second criterion is used to reduce the random selection of test cases, so that the APFD is better than single criteria based prioritization technique.

The existing multiple criteria based prioritization techniques proposed by Prakash and Gomathi (2014) and Kaur et al (2014) have very low APFD for triangle and Next date program. The APFD values for Commission program are almost same, because the priority lists of both techniques are almost same. The reason is that both of these techniques are prioritizing the test cases using “Total” strategy. Test cases are assigned priority on the basis of total number of entities, not considering the uncovered entities factor.

Therefore, it is concluded that the APFD values of our proposed multiple criteria based prioritization technique (Branch-Statement coverage) are relatively better than other three existing techniques. The reason is that it is based on two coverage criteria and it prioritizes the test cases using “Additional” strategy. Test cases are assigned priority on the basis of maximum coverage and covering entities that are not so far covered by previously choosed test case. One

coverage criteria is used to prioritize test cases and other is used to remove the random selection of test cases. So, random selection of test cases is reduced to much extent.

Following table shows the fault detection of test cases for Triangle program.

Table 5-6: Fault detection of test cases for Traingle program

Test Cases	Branch-Statement Coverage	Branch Covergae	N. Prakash and K. Gomathi, 2014	N. Kaur et al. , 2014
1	9	7	9	9
2	9	7	9	9
3	16	14	9	9
4	23	21	9	9
5	23	21	9	9
6	31	29	9	9
7	32	30	9	9
8	33	31	9	9
9	33	31	9	9
10	33	31	9	9
11	33	31	9	9
12	33	31	9	9
13	33	31	9	9
14	33	31	9	9
15	33	31	9	9
16	33	31	16	16
17	33	31	16	16
18	33	31	16	17
19	33	31	17	17
20	33	31	17	17
21	33	31	17	17
22	33	31	17	17
23	33	31	17	17
24	33	31	17	17
25	33	31	17	17
26	33	31	17	17
27	33	31	17	17
28	33	31	17	17
29	33	31	17	17
30	33	31	17	17
31	33	31	17	17
32	33	31	17	17

Test Cases	Branch-Statement Coverage	Branch Covergae	N. Prakash and K. Gomathi, 2014	N. Kaur et al. , 2014
33	33	31	17	17
34	33	31	17	25
35	33	31	17	26
36	33	32	25	26
37	34	32	26	26
38	34	32	26	27
39	34	33	26	27
40	35	33	27	27
41	35	33	27	27
42	35	33	27	27
43	35	33	27	27
44	35	33	27	27
45	35	34	27	27
46	36	34	27	27
47	36	34	27	27
48	36	34	27	27
49	36	34	27	27
50	36	34	27	27
51	36	34	27	27
52	36	34	27	27
53	36	34	27	27
54	36	34	27	27
55	36	34	27	27
56	36	34	27	27
57	36	34	27	27
58	36	36	27	27
59	36	36	27	27
60	36	36	27	27
61	36	36	27	27
62	36	36	27	27
63	36	36	27	27
64	36	36	27	27
65	36	36	27	27
66	36	36	27	27
67	36	36	27	27
68	36	36	27	27
69	36	36	27	27
70	36	36	27	27
71	36	36	27	27
72	36	36	27	27

Test Cases	Branch-Statement Coverage	Branch Covergae	N. Prakash and K. Gomathi, 2014	N. Kaur et al. , 2014
73	36	36	27	27
74	36	36	27	27
75	36	36	27	27
76	36	36	27	27
77	36	36	27	27
78	36	36	27	27
79	36	36	34	34
80	36	36	34	34
81	36	36	34	34
82	36	36	34	34
83	36	36	34	34
84	36	36	34	34
85	36	36	34	34
86	36	36	34	34
87	36	36	34	34
88	36	36	34	34
89	36	36	34	34
90	36	36	34	34
91	36	36	34	34
92	36	36	35	35
93	36	36	35	35
94	36	36	35	35
95	36	36	35	35
96	36	36	35	35
97	36	36	35	35
98	36	36	35	35
99	36	36	35	35
100	36	36	35	35
101	36	36	35	35
102	36	36	35	35
103	36	36	35	35
104	36	36	35	35
105	36	36	35	35
106	36	36	35	35
107	36	36	35	35
108	36	36	36	36
109	36	36	36	36
110	36	36	36	36

Figure 5.2 shows the graphical representation of fault detection of test cases for Triangle program.

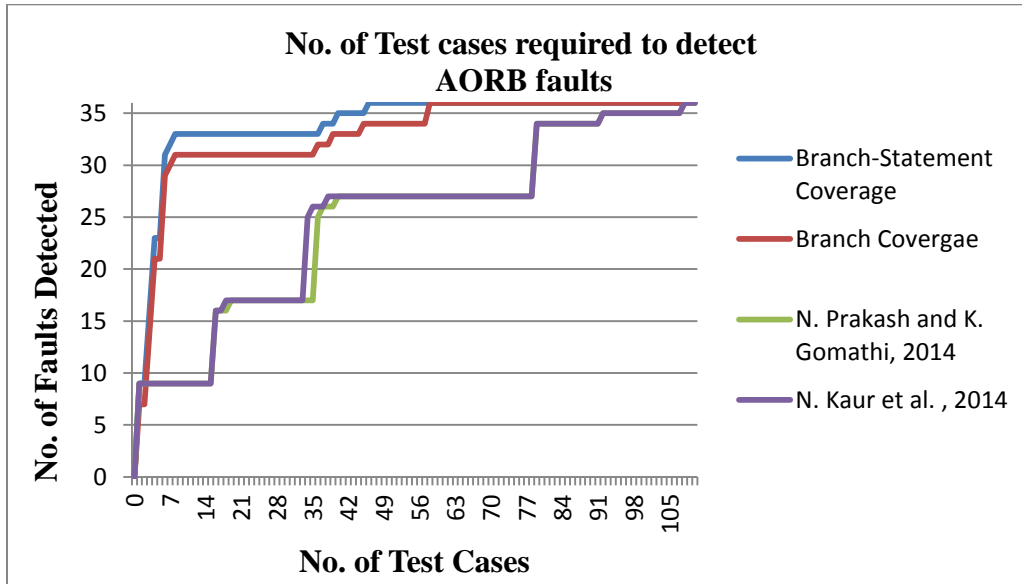


Figure 5-2: Graphical representation of Fault detection of test cases for Triangle program

From above graph, it is concluded that among all the techniques, Branch-Statement coverage requires minimum number of test cases to detect all the faults for Triangle program. APFD value of Branch-statement coverage is 1.59% higher than the APFD of Branch coverage. This minor difference is due to test case t56. In Branch-Statement coverage priority list, t56 is at 1th position whereas in Branch Coverage priority list, t56 is at 58th position and is detecting two faults. APFD value of Branch-statement coverage is 22.5% higher than APFD of Prakash et al. (2014) and 21.99% higher than Kaur et al. (2014). This difference is due to redundancy and low priority assigned to test cases by these two techniques. For example, the test case t26 detects 7 faults, t51 detects 7 faults and t32 detects 8 faults. Branch-Statement coverage assigned highest priority to these test cases. Test cases t26, t51 are redundant and t32 is given the lowest priority in the priority list of Prakash et al. (2014). Similarly, test case t51 is redundant and t26, t32 is given the lowest priority in the priority list of Kaur et al. (2014).

Following table shows the fault detection of test cases for Commission program.

Table 5-7: Fault detection of test cases for Commission program

Test Cases	Branch-Statement Coverage	Branch Covergae	N. Prakash and K. Gomathi, 2014	N. Kaur et al. , 2014
1	43	43	43	43
2	58	43	43	43
3	58	43	43	43
4	58	58	43	43
5	58	58	43	43
6	58	58	43	43
7	58	58	43	43
8	58	58	43	43
9	58	58	43	43
10	58	58	43	43
11	58	58	43	43
12	58	58	43	43
13	58	58	43	43
14	58	58	43	43
15	58	58	43	43
16	58	58	58	58
17	58	58	58	58
18	58	58	58	58
19	58	58	58	58
20	58	58	58	58
21	58	58	58	58
22	58	58	58	58
23	58	58	58	58
24	58	58	58	58
25	58	58	58	58

Figure 5.3 shows the graphical representation of fault detection of test cases for Commission program.

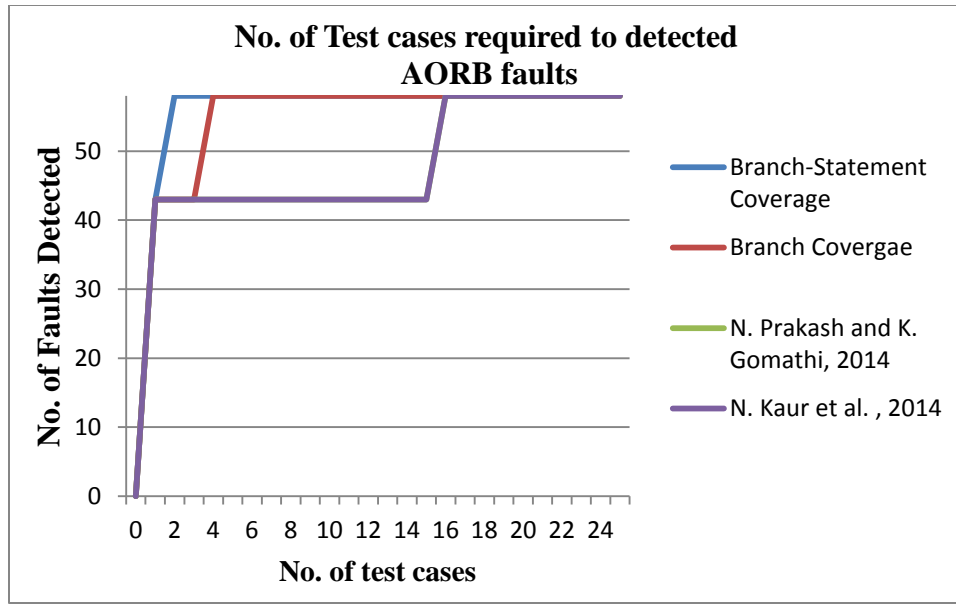


Figure 5-3: Graphical representation of Fault detection of test cases for Commission program

From above graph, it is determined that for Commission program, Branch-Statement coverage requires minimum number of test cases to detect all the faults as compared to other techniques. The APFD of Branch-Statement Coverage is 1.65% higher than APFD of Branch Coverage. Commission program has very few branches and minimum test cases (only two) are required to detect all faults. This difference in APFD is due to test case t4, which is at 2nd position in Branch-Statement Coverage priority list and at 4th position in Branch priority list. Similarly, the APFD of Branch-Statement coverage is 3.4% higher than APFD of both Prakash et al. (2014) and Kaur et al. (2014). Difference in APFD is due to low priority assigned to test case by these two techniques.

Following table shows the fault detection of test cases for Next Date program.

Table 5-8: Fault detection of test cases for Next Date program

Test Cases	Branch-Statement Coverage	Branch Covergae	N. Prakash and K. Gomathi, 2014	N. Kaur et al. , 2014
1	4	0	4	4
2	8	4	8	8
3	12	8	8	8
4	16	12	8	8
5	16	12	8	8
6	20	16	8	8

Test Cases	Branch-Statement Coverage	Branch Covergae	N. Prakash and K. Gomathi, 2014	N. Kaur et al. , 2014
7	20	16	8	8
8	20	20	8	8
9	20	20	8	8
10	20	20	8	8
11	20	20	8	12
12	20	20	12	12
13	24	20	12	12
14	24	20	12	12
15	24	20	12	12
16	24	20	12	12
17	24	20	12	12
18	24	20	12	12
19	24	20	12	12
20	24	20	12	12
21	24	20	12	12
22	24	20	12	12
23	24	20	12	12
24	24	20	12	12
25	24	20	12	12
26	24	20	16	16
27	24	20	16	16
28	24	20	16	16
29	24	20	16	16
30	24	20	16	16
31	24	20	16	16
32	24	20	16	16
33	24	20	16	16
34	24	20	16	16
35	24	20	16	16
36	24	20	16	16
37	24	20	16	16
38	24	20	16	16
39	24	20	16	16
40	24	20	16	16
41	24	20	16	16
42	24	20	16	16
43	24	20	16	16
44	24	20	16	16
45	24	20	16	16
46	24	20	16	16

Test Cases	Branch-Statement Coverage	Branch Covergae	N. Prakash and K. Gomathi, 2014	N. Kaur et al. , 2014
47	24	20	16	16
48	24	20	16	16
49	24	20	16	16
50	24	20	16	20
51	24	20	20	20
52	24	20	20	20
53	24	20	20	20
54	24	20	20	20
55	24	20	20	20
56	24	20	20	20
57	24	20	20	20
58	24	20	20	20
59	24	20	20	20
60	24	20	20	20
61	24	20	20	20
62	24	20	20	20
63	24	20	20	20
64	24	20	20	20
65	24	20	20	20
66	24	20	20	20
67	24	20	20	20
68	24	20	20	20
69	24	20	20	20
70	24	20	20	20
71	24	20	20	20
72	24	20	20	20
73	24	20	20	20
74	24	20	20	20
75	24	20	20	20
76	24	20	20	20
77	24	20	20	20
78	24	20	20	20
79	24	20	20	20
80	24	20	20	20
81	24	20	20	20
82	24	20	20	20
83	24	20	20	20
84	24	20	20	20
85	24	20	20	20
86	24	24	20	20

Test Cases	Branch-Statement Coverage	Branch Covergae	N. Prakash and K. Gomathi, 2014	N. Kaur et al. , 2014
87	24	24	20	20
88	24	24	20	20
89	24	24	20	20
90	24	24	20	20
91	24	24	20	20
92	24	24	20	20
93	24	24	20	20
94	24	24	20	20
95	24	24	20	20
96	24	24	20	20
97	24	24	20	20
98	24	24	20	20
99	24	24	20	20
100	24	24	20	20
101	24	24	24	24
102	24	24	24	24
103	24	24	24	24
104	24	24	24	24
105	24	24	24	24

Figure 5.4 shows the graphical representation of fault detection of test cases for Next Date program.

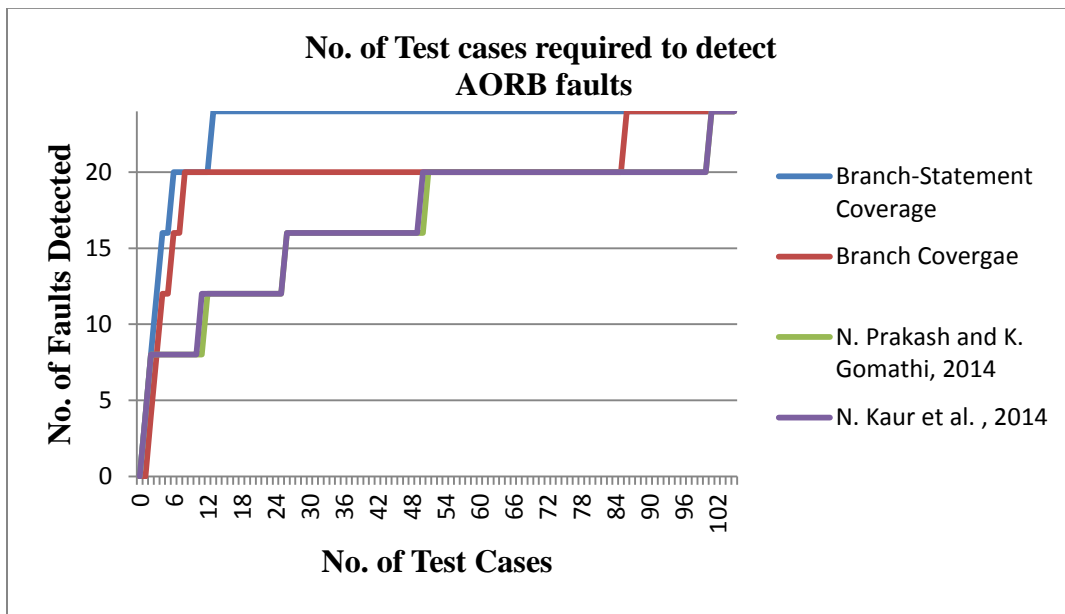


Figure 5-4: Graphical representation of Fault detection of test cases for Next Date program

From above graph, it is concluded that for Next Date program, Branch-Statement coverage requires minimum number of test cases to detect all the faults as compared to other techniques. The APFD of Branch-Statement coverage is 6% higher than the APFD of Branch coverage. This difference is due to test case t85 which detects 4 faults out of 24. In Branch-Statement coverage priority list, t85 is at 13th position and at 86th position in Branch coverage priority list. APFD value of Branch-statement coverage is 18.2% higher than APFD of Prakash et al. (2014) and 17.87% higher than Kaur et al. (2014). This difference is due to redundancy and low priority assigned to test cases by these two techniques. For example, test case t21 detects 4 faults out of 24. Branch-Statement coverage assigned highest priority (4th position) to this test case. Low priority is assigned to this test case (35th position) by both the priority lists of Prakash et al. (2014) and Kaur et al. (2014).

From above detailed discussion, it is concluded that the proposed multiple criteria based prioritization technique (Branch-Statement coverage) is performing better than the existing single criteria based prioritization technique and existing multiple criteria based prioritization technique. These techniques have many such test cases which are redundant or have assigned low priority but are detecting maximum faults. These techniques are selecting test cases randomly in case of a tie, also the existing multiple criteria based techniques are using “Total” strategy to prioritize the test cases. Our proposed technique is using “Additional” strategy and it is also minimizing randomness to much extent.

Chapter 6 : Conclusion and Future Work

Regression testing is important because in maintenance phase, software continuously undergoes changes and requirements are also changed which can result in addition of some new faults. Different cost reduction techniques are used to reduce the cost of regression testing. The most effective technique is test case prioritization as it does not eliminate any test case. Different criteria are used to assign priority to each test case on the basis of maximum coverage of those criteria. Those criteria can be the structural entities of source code like statements, branches, function etc. (Rothermel et al., 2001) or can be specification based like interaction between graphical events or customer priority etc. (Henard et al., 2016). Literature has different prioritization techniques based on single criterion and multiple criteria. Following are the answers of our research questions depicted in chapter 1, which we have identified after doing literature review and experimentation.

RQ 1: What are the existing and most commonly used test case prioritization techniques?

From the literature survey we have identified that test case prioritization techniques are separated into two broad categories: white box and black box. White box prioritization techniques use the source code information for prioritizing the test cases whereas, Black box prioritization techniques use the specification of software to prioritize the test cases. Test case prioritization uses “Total” and “Additional” as two different strategies to prioritize the test cases. “Total” strategy considers the total number of entities covered by each test case whereas, “Additional” strategy considers the additional entities which remain uncovered in the previous selection. “Additional” strategy is better than “Total” strategy because “Total” strategy does not consider the additional entities which remain uncovered in previous selection and thus repetition or missing of entities can happen. Based on these strategies, prioritization techniques can be single criterion or multiple criteria based. Multiple criteria based prioritization techniques are better than single criterion based prioritization techniques because these techniques are using more than one criteria to prioritize the test cases and hence increasing the rate of fault detection. In the literature, most of the work is based on single criteria based prioritization techniques.

RQ 2: What are the gaps in existing single and multiple criteria based prioritization techniques?

In existing single criterion based prioritization techniques, test case selection is random in case of occurrence of a tie between two or more test cases when two or more test cases provide equal coverage of entities of first criterion. This reduces the coverage of criterion and the fault detection rate is also decreased. Multiple criteria based prioritization techniques are studied less as compared to single criterion based prioritization techniques. Existing multiple criteria based prioritization techniques are not suitable because of some issues. These techniques use “Total” strategy to prioritize the test cases. “Total” strategy considers the total number of entities covered by each test case, rather than considering the entities which remains uncovered in the previous selection. Also, some techniques use randomized algorithms like Genetic Algorithm (GA). These algorithms are non-deterministic and do not guarantee to generate results in same way in every execution.

RQ 3: Does the proposed multiple criteria based prioritization technique improves fault detection rate when compared with existing prioritization techniques using single criterion or multiple criteria?

We have proposed a new multiple criteria based prioritization technique which prioritizes the test cases using “Additional” strategy. Our technique uses two coverage criteria to prioritize the test cases. One criterion is considered as primary and other is considered as secondary. Test cases are prioritized using primary criterion whereas secondary criterion is applied in case of occurrence of a tie between two or more test cases. This algorithm generates a priority list of test cases. We have performed experiments on three different subject programs to calculate the APFD of this priority list. For comparison, we have implemented the existing strongest single criteria based prioritization technique: Additional Branch Coverage and two existing multiple criteria based techniques proposed by Prakash et al. (2014) and Kaur et al. (2014). Priority lists of these techniques are also obtained. These priority lists are used to calculate the APFD of each technique for each subject program. Test case detecting maximum number of faults is given low priority by these existing techniques. Our technique overcomes this limitation by giving highest priority to test case which is detecting maximum number of faults. The final results of APFD for each subject program shows that our multiple criteria based prioritization technique performs

1.59% to 22.5% better than the existing single and multiple criteria based prioritization techniques.

6.1 Future Work

In future, we plan to extend our algorithm by using more than two criteria. This will increase the APFD to much extent. Also, now we have used AORB mutation operator for evaluation of our technique and comparison with existing techniques. In future, we will use some other mutation operator for this purpose. Using different operators, different types of faults can be identified. In this thesis, we have used three case studies: Triangle Program, Commission Program and Next Date Program. In future, we plan to use some larger and realistic case studies for comparison and evaluation.

References

- Ahmed, A.A., Shaheen, M. and Kosba, E. (2012). Software testing suite prioritization using multi-criteria fitness function. In Computer Theory and Applications (ICCTA), 22nd International Conference on IEEE, pp. 160-166.
- Arafeen, M.J. and Do, H. (2013). Test case prioritization using requirements-based clustering. In Software Testing, Verification and Validation (ICST), IEEE Sixth International Conference, pp. 312-321.
- Baresi, L., Pezze, M. (2006). An introduction to software testing, Electronic Notes in Theoretical Computer Science, vol.148, pp. 89-111.
- Bryce, R. and Colbourn, C. (2006). Prioritized interaction testing for pair-wise coverage with seeding and constraints. Information and Software Technology, 48(10), pp.960-970.
- Bryce, R.C. and Memon, A.M. (2007). Test suite prioritization by interaction coverage. In Workshop on Domain specific approaches to software test automation: in conjunction with the 6th ESEC/FSE joint meeting in ACM , pp. 1-7.
- Burnstein, I. (2006). Practical software testing: a process-oriented approach. Springer Science & Business Media.
- Chauhan, KR. and Singh, I. (2014). Latest Research and Development on Software Testing Techniques and Tools. International Journal of Current Engineering and Technology, 4(4).
- Chen, T.Y., Lau, M.F. (1996). Dividing strategies for the optimization of a test suite. Information Processing Letters, 60(3), pp.135– 141.
- Do, H., Rothermel, G. (2005). A controlled experiment assessing test case prioritization techniques via mutation faults. Proceedings of the 21st IEEE International Conference on Software Maintenance (ICSM). IEEE Computer Society Press, pp. 411–420.
- Duggal, G. and Suri, B. (2008). Understanding regression testing techniques. In Proceedings of 2nd National Conference on Challenges and Opportunities in Information Technology.

Elbaum, S., Malishevsky, A.G. and Rothermel, G. (2000).Prioritizing test cases for regression testing, 25(5), pp. 102-112.ACM.

Elbaum, S., Malishevsky, A.G. and Rothermel, G. (2002). Test case prioritization: A family of empirical studies. IEEE transactions on software engineering, 28(2), pp.159-182.

Elbaum, S., Rothermel, G., Kanduri, S. and Malishevsky, A.G. (2004).Selecting a cost-effective test case prioritization technique.Software Quality Journal, 12(3), pp.185-210.

Fazlalizadeh, Y., Khalilian, A., Azgomi, M., Parsa, S. (2009).Prioritizing test cases for resource constraint environments using historical test case performance data. 2nd IEEE International Conference on Computer Science and Information Technology, pp. 190–195.

Feldt, R., Poulding, S., Clark, D. and Yoo, S. (2016). Test set diameter: Quantifying the diversity of sets of test cases. 2016 IEEE International Conference on Software Testing, Verification and Validation (ICST), pp. 223-233.

Garey, M.R. (1979). Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H.Freeman and company: New York, 44(2), pp.340.

Harrold, M.J., Gupta, R., Soffa, M.L. (1993). A methodology for controlling the size of a test suite. ACM Transactions on Software Engineering and Methodology, 2(3), pp. 270–285.

Henard, C., Papadakis, M., Harman, M., Jia, Y. and Traon, Y.L. (2016).Comparing White-box and Black-box Test Prioritization.In Proceedings of the 38th International Conference on Software Engineering (ICSE).ACM.

Hooda, A., Pal, S. and Kumar, S. (2015). Regression Testing: A Complete Overview. International Journal of Advanced Research in Computer Science and Software Engineering, 5(5).

Hooda, I. and Chhillar, R.S. (2015). Software Test Process, Testing Types and Techniques. International Journal of Computer Applications, 111(13), pp.10-14.

Horgan, J., London, S. (1992). ATAC: A data flow coverage testing tool for c. Proceedings of the Symposium on Assessment of Quality Software Development Tools, IEEE Computer Society Press, pp. 2–10.

Jeffrey, D. and Gupta, N. (2008). Experiments with test case prioritization using relevant slices. *Journal of Systems and Software*, 81(2), pp.196-221.

Jorgensen, P.C. (2010). *Software Testing: A Craftsman's Approach*. 2nd Edition. Boca Raton, FL: CRC Press. ISBN: 13: 978-1-4665-6069-7.

Kaur, A. and Goyal, S. (2011). A genetic algorithm for regression test case prioritization using code coverage. *International journal on computer science and engineering*, 3(5), pp.1839-1847.

Kaur, n. and Mahajan, m. (2014). Regression testing with multiple criteria based test case prioritization. *International Journal for Research in Applied Science and Engineering Technology*, 2(V).

Khan, M.E. and Khan, F. (2012). A comparative study of white box, black box and grey box testing techniques. *International Journal of Advanced Computer Sciences and Applications*, 3(6), pp.12-1.

Khasragi, AB. (2016). Comparison between Multi objective Test Case Prioritization Techniques. 2th International conference in new research on Electrical Engineering & Computer Science.

Kim, J.M. and Porter, A. (2002). A history-based test prioritization technique for regression testing in resource constrained environments. In *Proceedings of the 24th international conference on software engineering*, pp. 119-129. ACM.

Korel, B., Koutsogiannakis, G. and Tahat, L.H. (2007), July. Model-based test prioritization heuristic methods and their evaluation. In *Proceedings of the 3rd international workshop on Advances in model-based testing*, pp. 34-43. ACM.

Kumar, H., Pal, V. and Chauhan, N. (2013). A hierarchical system test case prioritization technique based on requirements. *13th Annual International Software Testing Conference*, pp.4–5.

Leung, H.K. and White, L. (1989). Insights into regression testing (software testing). In Software Maintenance, 1989., Proceedings., Conference on IEEE, pp. 60-69.

Li, Z., Harman, M., Hierons, R. (2007). Search Algorithms for Regression Test Case Prioritization. Proceedings of IEEE Transactions on Software Engineering, Vol. 33, pp. 225-237.

Lin, S. (1965). Computer solutions of the traveling salesman problem. The Bell system technical journal, 44(10), pp.2245-2269.

Maheswari, R.U. and Mala, D.J. (2015). Combined genetic and simulated annealing approach for test case prioritization. Indian Journal of Science and Technology, 8(35).

Malishevsky, A.G., Ruthruff, J.R., Rothermel, G. and Elbaum, S. (2006). Cost-cognizant test case prioritization. Department of Computer Science and Engineering, University of Nebraska-Lincoln, Technical Report.

Marré, M. and Bertolino, A. (2003). Using spanning sets for coverage testing. IEEE Transactions on Software Engineering, 29(11), pp.974-984.

Myers, G.J. (1979). IBM Systems Research Institute, Lecturer in Computer Science, Polytechnic Institute of New York, The Art of Software Testing, by John Wiley & Sons.

Offutt, J., Pan, J., Voas, J. (1995). Procedures for reducing the size of coverage-based test sets. Proceedings of the 12th International Conference on Testing Computer Software, ACM Press, pp.111–123.

Prakash, N. and Gomathi, K. (2014). Improving Test Efficiency through Multiple Criteria Coverage Based Test Case Prioritization. International Journal of Scientific & Engineering Research, 5(4).

Quadri, S. and Farooq, S. (2010). Software Testing: Goals, Principles and Limitations. International Journal of Computer Applications, 6(9), pp.7-10.

Qu, B., Nie, C., Xu, B. and Zhang, X. (2007). Test case prioritization for black box testing. In Computer Software and Applications Conference. COMPSAC 2007. 31st Annual International, Vol. 1, pp. 465-474.

Rothermel, G, Harrold MJ. (1994). A framework for evaluating regression test selection techniques. Proceedings of the 16th International Conference on Software Engineering (ICSE), IEEE Computer Society Press, pp. 201–210.

Rothermel, G. Untch, R.H., Chu, C., Harrold, M.J. (1999). Test case prioritization: An empirical study. Proceedings of International Conference on Software Maintenance (ICSM), IEEE Computer Society Press, pp.179–188.

Rothermel, G., Untch, R.H., Chu, C. and Harrold, M.J. (2001).Prioritizing test cases for regression testing.IEEE Transactions on software engineering, 27(10), pp.929-948.

Saha, R.K., Zhang, L., Khurshid, S. and Perry, D.E. (2015). An information retrieval approach for regression test prioritization based on program changes.In Software Engineering (ICSE), 2015 IEEE/ACM 37th IEEE International Conference, Vol. 1, pp. 268-279.IEEE.

Saxena, R. and Singh, M. (2014).Gray Box Testing: Proactive Methodology for the Future Design of Test Cases to Reduce Overall System Cost. Journal of Basic and Applied Engineering Research, Volume 1, pp. 62-66.

Singh, S., Singh, G. and Singh, S. (2010). Software Testing.International Journal of Advanced Research in Computer Science, 1(3).

Srikanth, H., Williams, L. and Osborne, J. (2005).Towards the Prioritization of system test cases.North Carolina State University TR-44.

Srivastava, P.R. (2008). TEST CASE PRIORITIZATION. Journal of Theoretical & Applied Information Technology, 4(3).

Wong, W. E., Horgan, J. R., London, S., Agrawal, H. (1997). A study of effective regression testing in practice, Proceedings of the 8th International Symposium on Software Reliability Engineering (ISSRE), IEEE Computer Society, pp. 264–275.

Yoo, S. and Harman, M. (2012). Regression testing minimization, selection and prioritization: a survey. Software Testing, Verification and Reliability, 22(2), pp.67-120.

Appendix A: Source Code of Subject Programs

Source code of Triangle Program:

```
public class TriangleProbelm {
    String newline = System.getProperty("line.separator");
    public void triangle(inta,intb,int c){
        int match=0,d,e;
        if(a==b)
            match = match - match + 1;
        if(a==c)
            match= match - match + 2;
        if(b==c)
            match= match - match + 3;
            d=a+b;
            e=b+c;
        if(match==0){
            if(d<=c)
                System.out.print("NOT A TRIANGLE" +newline);
            else if(e<=a)
                System.out.print("NOT A TRIANGLE" +newline);
            else if(a+c<=b)
                System.out.print("NOT A TRIANGLE" +newline);
            else
                System.out.println("Scalane" +newline);
        }
        else if(match==1){
            if(a+c<=b)
                System.out.println("NOT A TRIANGLE" +newline);}
            else
                System.out.println("Isoscles" +newline);
    }
}
```



```

    }
else if (match==2){
if(a+c<=b)
System.out.print("NOT A TRIANGLE" +newline);
else
System.out.print("Isoscles" +newline);
    }
else if(match==3){
if(b+c<=a)
System.out.print("NOT A TRIANGLE" +newline);
else
System.out.print("Isoscles" +newline);
    }
else
    System.out.println("Equilateral" +newline);
}
}

```

Source Code of Commission Program:

```

public class Commission {
public void commission(int locks, int stocks, int barrels){
inttotalLocks,totalStocks,totalBarrels;
doublelockPrice, stockPrice, barrelPrice;
    doublelockSales, stockSales, barrelSales, sales, commission;
lockPrice = 45.0;
stockPrice = 30.0;
barrelPrice = 25.0;
totalLocks = 0;
totalStocks = 0;
totalBarrels = 0;
totalLocks = totalLocks + locks;

```

```

totalStocks = totalStocks + stocks;
totalBarrels = totalBarrels + barrels;
System.out.println("Locks sold:" +totalLocks);
System.out.println("Stocks sold:" +totalStocks);
System.out.println("Barrels sold:" +totalBarrels);
lockSales = lockPrice * totalLocks;
stockSales = stockPrice * totalStocks;
barrelSales = barrelPrice * totalBarrels;
sales = lockSales + stockSales + barrelSales;
System.out.println("Total Sales:" +sales);
if(sales > 1800.0)
    {
        commission = 0.10 * 1000.0;
        commission = commission + 0.15 * 800.0;
        commission = commission + 0.20 * (sales - 1800.0);
    }
else
    {
        if (sales > 1000.0)
            {
                commission = 0.10 * 1000.0;
                commission = commission + 0.15*(sales - 1000.0);
            }
        else
            {
                commission = 0.10 * sales;
            }
    }
System.out.println("Commission:" +commission);
}

```

```

public static void main(String[] args )
{
    Commission object=new Commission();
object.commission(1, 1, 1);
}
}

```

Source Code of nextDate Program:

```

public class nextdate {
    public void nextday(intday,intmonth,int year){
        inttomorrowDay = day,tomorrowMonth=month,tomorrowYear=year;
        booleanleapyear;
        leapyear= ((year % 4 == 0) && (year % 100 != 0 || year %400==0));
        switch(month)
        {
            case 1: case 3: case 5: case 7: case 8: case 10:
                if (day < 31)
                {
                    tomorrowDay = day + 1;
                }
                else
                {
                    tomorrowDay = 1;
                    tomorrowMonth = month + 1;
                }
                break;
            case 4: case 6: case 9: case 11:
                if (day < 30)
                {
                    tomorrowDay = day + 1;
                }

```

```

else
    {
tomorrowDay = 1;
tomorrowMonth = month + 1;
    }
break;
case 12:
    if(day < 31)
        {
tomorrowDay = day + 1;
        }
    else
        {
tomorrowDay = 1;
tomorrowMonth = 1;
tomorrowYear = year + 1;
        }
    break;
case 2:
    if (day < 28)
        {
tomorrowDay = day + 1;
        }
    else if( day == 28)
        {
            if (leapyear)
                tomorrowDay = 29;
            else
                {
tomorrowDay = 1;

```

```

        tomorrowMonth = 3;
    }
}
else
{
    if (day == 29)
    {
        if(leapyear)
        {
            tomorrowDay = 1;
            tomorrowMonth = 3;
        }
        else
            System.out.println("February cannot have days" +day);
    }
}
break;
}

    System.out.printf("Date is:" +tomorrowDay + "-" +tomorrowMonth + "-"
+tomorrowYear);
}
public static void main(String[] args )
{
nextdate object=new nextdate();
object.nextday(1,1,1);
}
}

```

Appendix B: Test Data for Subject Programs

Test data for Triangle Program:

Test case	Statements Covered	Cov(t) for Statements	Branches covered	Cov(t) for branches	Errors detected
t1	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	
t2	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	
t3	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	
t4	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	
t5	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	
t6	1,2,4,6,7,8,9,10,22,28, 34,35,38,39	14	2,4,5,8,16,20,23, 26	8	21,22,23,24,46,4 7,48
t7	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	
t8	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	
t9	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	
t10	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	
t11	1,2,4,6,8,9,10,14,17,18 ,19	12	2,4,6,7,10,12,13	7	
t12	1,2,4,6,7,8,9,10,22,28, 34,38,39	13	2,4,5,8,16,20,23, 26	8	21,22,23,24,46,4 7,48
t13	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	
t14	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	
t15	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	
t16	1,2,4,6,8,9,10,11,14,17 ,20,21	12	2,4,6,7,10,12,13	7	
t17	1,2,4,6,8,9,10,11,14,17 ,20,21	12	2,4,6,7,10,12,13	7	
t18	1,2,4,6,8,9,10,11,14,17 ,18,19	12	2,4,6,7,10,12,13	7	
t19	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	
t20	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	
t21	1,2,4,6,8,9,10,11,14,17 ,18,19	12	2,4,6,7,10,12,13	7	
t22	1,2,4,6,8,9,10,11,14,17 ,18,19	12	2,4,6,7,10,12,13	7	
t23	1,2,4,6,8,9,10,11,14,17 ,18,19	12	2,4,6,7,10,12,13	7	
t24	1,2,4,6,7,8,9,10,22,28, 34,35,38,39	14	2,4,5,8,16,20,23, 26	8	21,22,23,24,46,4 7,48

t25	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	
t26	1,2,3,4,6,8,9,10,22,23, 24,25	12	1,4,6,8,15,18	6	5,6,7,8,38,39,40
t27	1,2,3,4,6,8,9,10,22,23, 26,27	12	1,4,6,8,15,18	6	5,6,7,8,38,39,40
t28	1,2,3,4,6,8,9,10,22,23, 26,27	12	1,4,6,8,15,18	6	5,6,7,8,38,39,40
t29	1,2,3,4,6,8,9,10,22,23, 26,27	12	1,4,6,8,15,18	6	5,6,7,8,38,39,40
t30	1,2,3,4,6,8,9,10,22,23, 26,27	12	1,4,6,8,15,18	6	5,6,7,8,38,39,40
t31	1,2,4,6,7,8,9,10,22,28, 34,35,38,39	14	2,4,5,8,16,20,23, 26	8	21,22,23,24,46,4 7,48
t32	1,2,4,6,8,9,10,11,14,17 ,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,32,3 4,35,36
t33	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	25
t34	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	
t35	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	
t36	1,2,4,6,8,9,10,11,14,17 ,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,3 2,34,35,36
t37	1,2,4,6,7,8,9,10,22,28, 34,35,38,39	14	2,4,5,8,16,20,23, 26	8	21,22,23,24,46,4 7,48
t38	1,2,4,6,8,9,10,11,14,17 ,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,32, 34,35,36
t39	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	25
t40	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	
t41	1,2,4,6,8,9,10,11,14,17 ,18,19	12	2,4,6,7,10,12,13	7	
t42	1,2,4,6,8,9,10,11,14,17 ,18,19	12	2,4,6,7,10,12,13	7	33
t43	1,2,4,6,8,9,10,11,14,17 ,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,3 2,34,35,36
t44	1,2,4,6,8,9,10,11,14,17 ,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,32, 34,35,36
t45	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	25
t46	1,2,4,6,8,9,10,11,14,17 ,18,19	12	2,4,6,7,10,12,13	7	
t47	1,2,4,6,8,9,10,11,14,17 ,18,19	12	2,4,6,7,10,12,13	7	33
t48	1,2,4,6,8,9,10,11,14,17 ,18,19	12	2,4,6,7,10,12,13	7	33
t49	1,2,4,6,7,8,9,10,22,28, 34,38,39	13	2,4,5,8,16,20,23, 26	8	21,22,23,24,46,4 7,48
t50	1,2,4,6,8,9,10,11,14,17 ,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,32, 34,35,36

t51	1,2,4,5,6,8,9,10,22,28, 29,32,33	13	2,3,6,8,16,19,22	7	13,14,15,16,42,4 3, 44
t52	1,2,4,6,8,9,10,11,14,17 ,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,3 2, 34, 36
t53	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	25
t54	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	
t55	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	
t56	1,2,3,4,5,6,7,8,9,10,22, 28,34,38,39	15	1,3,5,8,16,20,23, 26	8	17,20,21,22,23,2 4,46,47,48
t57	1,2,3,4,6,8,9,10,22,23, 26,27	12	1,4,6,8,15,18	6	5,6,7,8,38,39,40
t58	1,2,3,4,6,8,9,10,22,23, 26,27	12	1,4,6,8,15,18	6	5,6,7,8,38,39,40
t59	1,2,3,4,6,8,9,10,22,23, 26,27	12	1,4,6,8,15,18	6	5,6,7,8,38,39,40
t60	1,2,3,4,6,8,9,10,22,23, 26,27	12	1,4,6,8,15,18	6	5,6,7,8,38,39,40
t61	1,2,4,5,6,8,9,10,22,28, 29,32,33	13	2,3,6,8,16,19,22	7	13,14,15,16,42,4 3,44
t62	1,2,4,6,7,8,9,10,22,28, 34,35,38,39	14	2,4,5,8,16,20,23, 26	8	21,22,23,24,46,4 7,48
t63	1,2,4,6,8,9,10,11,14,17 ,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,32,3 4,35,36
t64	1,1,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	25
t65	1,1,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	25
t66	1,2,4,5,6,8,9,10,22,28, 29,30,31	13	2,3,6,8,16,19,22	7	16,41
t67	1,2,4,6,8,9,10,11,14,17 ,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,3 2,34,35,36
t68	1,2,4,6,8,9,10,11,14,17 ,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,3 2,34,35,36
t69	1,2,4,6,8,9,10,11,14,17 ,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,3 2,34,35,36
t70	1,2,4,6,8,9,10,11,14,17 ,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,3 2,34,35,36
t71	1,2,4,5,6,8,9,10,22,28, 29,30,31	13	2,3,6,8,16,19,21	7	16,41
t72	1,2,4,6,8,9,10,11,14,17 ,18,19	12	2,4,6,7,10,12,13	7	
t73	1,2,4,6,8,9,10,11,14,17 ,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,3 2,34,35,36
t74	1,2,4,6,7,8,9,10,22,28, 34,35,38,39	14	2,4,5,8,16,20,23, 26	8	21,22,23,24,46,4 7,48
t75	1,2,4,6,8,9,10,11,14,17 ,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,3 2,34,35,36

t76	1,2,4,6,8,9,10,11,14,17,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,32,34,35,36
t77	1,2,4,5,6,8,9,10,22,28,29,32,33	13	2,3,6,8,16,19,22	7	13,14,15,16,42,43,44
t78	1,2,4,6,8,9,10,11,14,17,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,32,34,35,36
t79	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	25
t80	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	
t81	1,2,4,6,7,8,9,10,22,28,34,35,38,39	14	2,4,5,8,16,20,23,26	8	21,22,23,24,46,47,48
t82	1,2,4,5,6,8,9,10,22,28,29,32,33	13	2,3,6,8,16,19,22	7	13,14,15,16,42,43,44
t83	1,2,4,6,8,9,10,11,14,17,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,32,34,35,36
t84	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	25
t85	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	25
t86	1,2,3,4,6,8,9,10,22,23,26,27	12	1,4,6,8,15,18	6	5,6,7,8,38,39,40
t87	1,2,3,4,5,6,7,8,9,10,22,28,34,38,39	15	1,3,5,8,16,20,23,26	8	17,20,21,22,23,24,46,48,49
t88	1,2,3,4,6,8,9,10,22,23,26,27	12	1,4,6,8,15,18	6	5,6,7,8,38,39,40
t89	1,2,3,4,6,8,9,10,22,23,26,29	12	1,4,6,8,15,18	6	5,6,7,8,38,39,40
t90	1,2,3,4,6,8,9,10,22,23,26,27	12	1,4,6,8,15,18	6	5,6,7,8,38,39,40
t91	1,2,4,6,8,9,10,11,14,17,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,32,34,35,36
t92	1,2,4,5,6,8,9,10,22,28,29,32,33	13	2,3,6,8,16,19,22	7	13,14,15,16,42,43,44
t93	1,2,4,6,8,9,10,11,14,17,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,32,34,35,36
t94	1,2,4,6,8,9,10,11,14,17,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,32,34,35,36
t95	1,2,4,6,8,9,10,11,14,17,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,32,34,35,36
t96	1,2,4,6,8,9,10,11,14,17,18,19	12	2,4,6,7,10,12,13	7	33
t97	1,2,4,5,6,8,9,10,22,28,29,32,33	13	2,3,6,8,16,19,22	7	13,14,15,16,42,43,44
t98	1,2,4,6,8,9,10,11,14,17,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,32,34,35,36
t99	1,2,4,6,7,8,9,10,22,28,34,35,38,39	14	2,4,5,8,16,20,23,26	8	21,22,23,24,46,47,48
t100	1,2,4,6,8,9,10,11,14,17,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,32,34,35,36

t101	1,2,4,6,8,9,10,11,14,15,16	11	2,4,6,7,10,11	6	29,30
t102	1,2,4,6,8,9,10,11,14,17,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,32,34,35,36
t103	1,2,4,5,6,8,9,10,22,28,29,32,33	13	2,3,6,8,16,19,22	7	13,14,15,16,42,43,44
t104	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	25
t105	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	25
t106	1,2,4,6,7,8,9,10,22,28,34,35,38,39	14	2,4,5,8,16,20,23,26	8	21,22,23,24,46,47,48
t107	1,2,4,6,8,9,10,11,14,17,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,32,34,35,36
t108	1,2,4,5,6,8,9,10,22,28,29,32,33	13	2,3,6,8,16,19,22	7	13,14,15,16,42,43,44
t109	1,2,4,6,8,9,10,11,14,17,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,32,34,35,36
t110	1,2,4,6,8,9,10,11,12,13	10	2,4,6,7,9	5	25
t111	1,2,4,6,8,9,10,11,14,17,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,32,34,35,36
t112	1,2,4,6,7,8,9,10,22,28,34,35,38,39	14	2,4,5,8,16,20,23,26	8	21,22,23,24,46,47,48
t113	1,2,4,5,6,7,8,9,10,22,28,29,32,33	14	2,3,6,8,16,19,22	7	13,14,15,16,42,43,44
t114	1,2,4,6,8,9,10,11,14,17,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,32,34,35,36
t115	1,2,4,6,8,9,10,11,14,17,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,32,34,35,36
t116	1,2,4,6,8,9,10,11,14,17,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,32,34,35,36
t117	1,2,4,6,8,9,10,11,14,17,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,32,34,35,36
t118	1,2,4,5,6,8,9,10,22,28,29,32,33	13	2,3,6,8,16,19,22	7	13,14,15,16,42,43,44
t119	1,2,4,6,8,9,10,11,14,17,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,32,34,35,36
t120	1,2,4,6,8,9,10,11,14,17,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,32,34,35,36
t121	1,2,4,6,8,9,10,11,14,17,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,32,34,35,36
t122	1,2,4,6,8,9,10,11,14,17,20,21	12	2,4,6,7,10,12,14	7	26,27,28,30,31,32,34,35,36
t123	1,2,4,5,6,8,9,10,22,28,29,32,32	13	2,3,6,8,16,19,22	7	13,14,15,16,42,43,44
t124	1,2,4,6,7,8,9,10,22,28,34,35,38,39	14	2,4,5,8,16,20,23,26	8	21,22,23,24,46,47,48
t125	1,2,4,6,8,9,10,11,14,17	12	2,4,6,7,10,12,14	7	26,27,28,30,31,32

	,20,21				2,34,35,36
--	--------	--	--	--	------------

Test data for Commission Program:

Test case	Statements Covered	Cov(t) for Statements	Branches covered	Cov(t) for branches	Errors detected
t1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,26,27,28,30	26	2,3	2	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,57,58,59,60,61,62,63
t2	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,26,27,28,30	26	2,3	2	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,57,58,59,60,61,62,63
t3	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,26,27,28,30	26	2,3	2	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,57,58,59,60,61,62,63
t4	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t5	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,57,58,59,60,61,62,63
t6	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,26,27,28,30	26	2,3	2	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,57,58,59,60,61,62,63
t7	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,26,27,28,30	26	2,3	2	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,57,58,59,60,61,62,63
t8	1,2,3,4,5,6,7,8,9,10,11,	26	2,3	2	1,2,3,4,5,6,7,8,9,10,11,12,

	12,13,14,15,16,17,18,19,20,21,22,26,27,28,30				13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,57,58,59,60,61,62,63
t9	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,26,27,28,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t10	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,26,27,28,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t11	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,26,27,28,30	26	2,3	2	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,57,58,59,60,61,62,63
t12	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,26,27,28,30	26	2,3	2	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,57,58,59,60,61,62,63
t13	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t14	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t15	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t16	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51

	9,20,21,22,23,24,25,30				1,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t17	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t18	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t19	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t20	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t21	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t22	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t23	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51

t24	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,37,3 8,39,40,41,42,43,44,45,46, 47,48,49,50,51
t25	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,37,3 8,39,40,41,42,43,44,45,46, 47,48,49,50,51
t26	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,26,27,28,30	26	2,3	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,57,5 8,59,60,61,62,63
t27	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,26,27,28,30	26	2,3	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,57,5 8,59,60,61,62,63
t28	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,26,27,28,30	26	2,3	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,57,5 8,59,60,61,62,63
t29	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,37,3 8,39,40,41,42,43,44,45,46, 47,48,49,50,51
t30	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,37,3 8,39,40,41,42,43,44,45,46, 47,48,49,50,51
t31	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	2,3	2	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,57,5 8,59,60,61,62,63
t32	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1	26	2,3	2	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2

	9,20,21,22,23,24,25,30				1,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,57,58,59,60,61,62,63
t33	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	2,3	2	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,57,58,59,60,61,62,63
t34	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t35	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t36	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	2,3	2	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,57,58,59,60,61,62,63
t37	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t38	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t39	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t40	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,

					30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t41	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t42	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t43	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t44	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t45	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t46	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t47	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t48	1,2,3,4,5,6,7,8,9,10,11,	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,

	12,13,14,15,16,17,18,19,20,21,22,23,24,25,30				13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t49	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t50	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t51	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t52	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t53	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t54	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t55	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51

					47,48,49,50,51
t56	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,37,3 8,39,40,41,42,43,44,45,46, 47,48,49,50,51
t57	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,37,3 8,39,40,41,42,43,44,45,46, 47,48,49,50,51
t58	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,37,3 8,39,40,41,42,43,44,45,46, 47,48,49,50,51
t59	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,37,3 8,39,40,41,42,43,44,45,46, 47,48,49,50,51
t60	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,37,3 8,39,40,41,42,43,44,45,46, 47,48,49,50,51
t61	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,37,3 8,39,40,41,42,43,44,45,46, 47,48,49,50,51
t62	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,37,3 8,39,40,41,42,43,44,45,46, 47,48,49,50,51
t63	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29,

					30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t64	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t65	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t66	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t67	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t68	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t69	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t70	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t71	1,2,3,4,5,6,7,8,9,10,11,	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,

	12,13,14,15,16,17,18,19,20,21,22,23,24,25,30				13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t72	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t73	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t74	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t75	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t76	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t77	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t78	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51

					47,48,49,50,51
t79	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,37,3 8,39,40,41,42,43,44,45,46, 47,48,49,50,51
t80	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,37,3 8,39,40,41,42,43,44,45,46, 47,48,49,50,51
t81	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,37,3 8,39,40,41,42,43,44,45,46, 47,48,49,50,51
t82	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,37,3 8,39,40,41,42,43,44,45,46, 47,48,49,50,51
t83	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,37,3 8,39,40,41,42,43,44,45,46, 47,48,49,50,51
t84	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,37,3 8,39,40,41,42,43,44,45,46, 47,48,49,50,51
t85	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,37,3 8,39,40,41,42,43,44,45,46, 47,48,49,50,51
t86	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29,

					30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t87	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t88	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t89	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t90	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t91	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t92	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t93	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t94	1,2,3,4,5,6,7,8,9,10,11,	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,

	12,13,14,15,16,17,18,19,20,21,22,23,24,25,30				13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t95	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t96	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t97	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t98	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t99	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t100	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t101	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51

					47,48,49,50,51
t102	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,37,3 8,39,40,41,42,43,44,45,46, 47,48,49,50,51
t103	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,37,3 8,39,40,41,42,43,44,45,46, 47,48,49,50,51
t104	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,37,3 8,39,40,41,42,43,44,45,46, 47,48,49,50,51
t105	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,37,3 8,39,40,41,42,43,44,45,46, 47,48,49,50,51
t106	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,37,3 8,39,40,41,42,43,44,45,46, 47,48,49,50,51
t107	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,37,3 8,39,40,41,42,43,44,45,46, 47,48,49,50,51
t108	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29, 30,31,32,33,34,35,36,37,3 8,39,40,41,42,43,44,45,46, 47,48,49,50,51
t109	1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18,1 9,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20,2 1,22,23,24,25,26,27,28,29,

					30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t110	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t111	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t112	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t113	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t114	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t115	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t116	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t117	1,2,3,4,5,6,7,8,9,10,11,	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,

	12,13,14,15,16,17,18,19,20,21,22,23,24,25,30				13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t118	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t119	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t120	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t121	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t122	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t123	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51
t124	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51

					47,48,49,50,51
t125	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30	26	1	1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51

Test data for Next Date Program:

Test case	Statements Covered	Cov(t) for Statements	Branches covered	Cov(t) for branches	Errors detected
t1	1,2,3,5,6,12,13,43	8	1,13	2	13,14,15,16
t2	1,2,3,5,6,12,13,43	9	1,13	2	13,14,15,16
t3	1,2,3,5,6,12,13,43	8	1,13	2	13,14,15,16
t4	1,2,3,5,6,12,13,43	8	1,13	2	13,14,15,16
t5	1,2,3,5,6,12,13,43	9	1,13	2	13,14,15,16
t6	1,2,3,5,6,7,8,9,10,11,16,17,18,19,24,30,31,32,43	19	2,4,6,8,10,12,16,18,20,22,26,29,31	13	37,38,39,40
t7	1,2,3,5,6,7,8,9,10,11,16,17,18,19,24,30,31,32,43	20	2,4,6,8,10,12,16,18,20,22,26,29,31	13	37,38,39,40
t8	1,2,3,5,6,7,8,9,10,11,16,17,18,19,24,30,31,32,43	19	2,4,6,8,10,12,16,18,20,22,26,29,31	13	37,38,39,40
t9	1,2,3,5,6,7,8,9,10,11,16,17,18,19,24,30,31,32,43	19	2,4,6,8,10,12,16,18,20,22,26,29,31	13	37,38,39,40
t10	1,2,3,5,6,7,8,9,10,11,16,17,18,19,24,30,31,32,43	20	2,4,6,8,10,12,16,18,20,22,26,29,31	13	37,38,39,40
t11	1,2,3,5,6,7,8,9,10,11,16,17,20,21,43	15	2,4,6,8,10,12,16,17,23	9	21,22,23,24
t12	1,2,3,5,6,7,8,9,10,11,16,17,20,21,43	16	2,4,6,8,10,12,16,17,23	9	21,22,23,24
t13	1,2,3,5,6,7,8,9,10,11,16,17,20,21,43	15	2,4,6,8,10,12,16,17,23	9	21,22,23,24
t14	1,2,3,5,6,7,8,9,10,11,16,17,20,21,43	15	2,4,6,8,10,12,16,17,23	9	21,22,23,24
t15	1,2,3,5,6,7,8,9,10,11,16,17,20,21,43	16	2,4,6,8,10,12,16,17,23	9	21,22,23,24
t16	1,2,3,5,6,7,8,9,10,11,16,17,20,21,43	17	2,4,6,8,10,12,16,18,20,21,23	11	21,22,23,24
t17	1,2,3,5,6,7,8,9,10,11,16,17,20,21,43	18	2,4,6,8,10,12,16,18,20,21,23	11	21,22,23,24
t18	1,2,3,5,6,7,8,9,10,11,16,17,20,21,43	17	2,4,6,8,10,12,16,18,20,21,23	11	21,22,23,24

t19	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	17	2,4,6,8,10,12,16,18,20,2 1,23	11	21,22,23,24
t20	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	18	2,4,6,8,10,12,16,18,20,2 1,23	11	21,22,23,24
t21	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,25,26,43	18	2,4,6,8,10,12,16,18,20,2 2,25,27	12	29,30,31,32
t22	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,25,26,43	19	2,4,6,8,10,12,16,18,20,2 2,25,27	12	29,30,31,32
t23	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,25,26,43	18	2,4,6,8,10,12,16,18,20,2 2,25,27	12	29,30,31,32
t24	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,25,26,43	18	2,4,6,8,10,12,16,18,20,2 2,25,27	12	29,30,31,32
t25	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,25,26,43	19	2,4,6,8,10,12,16,18,20,2 2,25,27	12	29,30,31,32
t26	1,2,3,5,6,12,13,43	8	1,13	2	13,14,15,16
t27	1,2,3,5,6,12,13,43	9	1,13	2	13,14,15,16
t28	1,2,3,5,6,12,13,43	8	1,13	2	13,14,15,16
t29	1,2,3,5,6,12,13,43	8	1,13	2	13,14,15,16
t30	1,2,3,5,6,12,13,43	9	1,13	2	13,14,15,16
t31	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,30,31,32,43	19	2,4,6,8,10,12,16,18,20,2 2,26,29,31	13	37,38,39,40
t32	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,30,31,32,43	20	2,4,6,8,10,12,16,18,20,2 2,26,29,31	13	37,38,39,40
t33	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,30,31,32,43	19	2,4,6,8,10,12,16,18,20,2 2,26,29,31	13	37,38,39,40
t34	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,30,31,32,43	19	2,4,6,8,10,12,16,18,20,2 2,26,29,31	13	37,38,39,40
t35	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,30,31,32,43	20	2,4,6,8,10,12,16,18,20,2 2,26,29,31	13	37,38,39,40
t36	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	15	2,4,6,8,10,12,16,17,23	9	21,22,23,24
t37	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	16	2,4,6,8,10,12,16,17,23	9	21,22,23,24
t38	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	15	2,4,6,8,10,12,16,17,23	9	21,22,23,24
t39	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	15	2,4,6,8,10,12,16,17,23	9	21,22,23,24
t40	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	16	2,4,6,8,10,12,16,17,23	9	21,22,23,24
t41	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	17	2,4,6,8,10,12,16,18,20,2 1,23	11	21,22,23,24
t42	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	18	2,4,6,8,10,12,16,18,20,2 1,23	11	21,22,23,24
t43	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	17	2,4,6,8,10,12,16,18,20,2 1,23	11	21,22,23,24

t44	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	17	2,4,6,8,10,12,16,18,20,2 1,23	11	21,22,23,24
t45	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	18	2,4,6,8,10,12,16,18,20,2 1,23	11	21,22,23,24
t46	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,25,26,43	18	2,4,6,8,10,12,16,18,20,2 2,25,27	12	29,30,31,32
t47	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,25,26,43	19	2,4,6,8,10,12,16,18,20,2 2,25,27	12	29,30,31,32
t48	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,25,26,43	18	2,4,6,8,10,12,16,18,20,2 2,25,27	12	29,30,31,32
t49	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,25,26,43	18	2,4,6,8,10,12,16,18,20,2 2,25,27	12	29,30,31,32
t50	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,25,26,43	19	2,4,6,8,10,12,16,18,20,2 2,25,27	12	29,30,31,32
t51	1,2,3,5,6,12,13,43	8	1,13	2	13,14,15,16
t52	1,2,3,5,6,12,13,43	9	1,13	2	13,14,15,16
t53	1,2,3,5,6,12,13,43	8	1,13	2	13,14,15,16
t54	1,2,3,5,6,12,13,43	8	1,13	2	13,14,15,16
t55	1,2,3,5,6,12,13,43	9	1,13	2	13,14,15,16
t56	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,30,31,32,43	19	2,4,6,8,10,12,16,18,20,2 2,26,29,31	13	37,38,39,40
t57	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,30,31,32,43	20	2,4,6,8,10,12,16,18,20,2 2,26,29,31	13	37,38,39,40
t58	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,30,31,32,43	19	2,4,6,8,10,12,16,18,20,2 2,26,29,31	13	37,38,39,40
t59	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,30,31,32,43	19	2,4,6,8,10,12,16,18,20,2 2,26,29,31	13	37,38,39,40
t60	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,30,31,32,43	20	2,4,6,8,10,12,16,18,20,2 2,26,29,31	13	37,38,39,40
t61	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	15	2,4,6,8,10,12,16,17,23	9	21,22,23,24
t62	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	16	2,4,6,8,10,12,16,17,23	9	21,22,23,24
t63	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	15	2,4,6,8,10,12,16,17,23	9	21,22,23,24
t64	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	15	2,4,6,8,10,12,16,17,23	9	21,22,23,24
t65	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	16	2,4,6,8,10,12,16,17,23	9	21,22,23,24
t66	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	17	2,4,6,8,10,12,16,18,20,2 1,23	11	21,22,23,24
t67	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	18	2,4,6,8,10,12,16,18,20,2 1,23	11	21,22,23,24
t68	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	17	2,4,6,8,10,12,16,18,20,2 1,23	11	21,22,23,24

t69	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	17	2,4,6,8,10,12,16,18,20,2 1,23	11	21,22,23,24
t70	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	18	2,4,6,8,10,12,16,18,20,2 1,23	11	21,22,23,24
t71	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,25,26,43	18	2,4,6,8,10,12,16,18,20,2 2,25,27	12	29,30,31,32
t72	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,25,26,43	19	2,4,6,8,10,12,16,18,20,2 2,25,27	12	29,30,31,32
t73	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,25,26,43	18	2,4,6,8,10,12,16,18,20,2 2,25,27	12	29,30,31,32
t74	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,25,26,43	18	2,4,6,8,10,12,16,18,20,2 2,25,27	12	29,30,31,32
t75	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,25,26,43	19	2,4,6,8,10,12,16,18,20,2 2,25,27	12	29,30,31,32
t76	1,2,3,5,6,12,13,43	8	1,13	2	13,14,15,16
t77	1,2,3,5,6,12,13,43	9	1,13	2	13,14,15,16
t78	1,2,3,5,6,12,13,43	8	1,13	2	13,14,15,16
t79	1,2,3,5,6,12,13,43	8	1,13	2	13,14,15,16
t80	1,2,3,5,6,12,13,43	8	1,13	2	13,14,15,16
t81	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,30,31,33,34, 36,37,43	22	22,4,6,8,10,12,16,18,20, 22,26,29,32,33,36	15	
t82	1,2,3,4,5,6,7,8,9,10,11,1 6,17,18,19,24,30,31,33,3 4,35,43	22	2,4,6,8,10,12,16,18,20,2 2,26,29,32,33,35	15	1,2,3,4
t83	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,30,31,33,34, 36,37,43	22	2,4,6,8,10,12,16,18,20,2 2,26,29,32,33,36	15	
t84	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,30,31,34,36, 37,43	21	2,4,6,8,10,12,16,18,20,2 2,26,29,32,33,36	15	
t85	1,2,3,4,5,6,7,8,9,10,11,1 6,17,18,19,24,30,31,33,3 4,35,43	22	2,4,6,8,10,12,16,18,20,2 2,26,29,32,33,3	15	1,2,3,4,9,10,11, 12
t86	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	15	2,4,6,8,10,12,16,17,23	9	21,22,23,24
t87	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	16	2,4,6,8,10,12,16,17,23	9	21,22,23,24
t88	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	15	2,4,6,8,10,12,16,17,23	9	21,22,23,24
t89	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	15	2,4,6,8,10,12,16,17,23	9	21,22,23,24
t90	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	16	2,4,6,8,10,12,16,17,23	9	21,22,23,24
t91	1,2,3,5,6,7,8,9,10,11,16,	17	2,4,6,8,10,12,16,18,20,2	11	21,22,23,24

	17,20,21,43		1,23		
t92	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	18	2,4,6,8,10,12,16,18,20,2 1,23	11	21,22,23,24
t93	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	17	2,4,6,8,10,12,16,18,20,2 1,23	11	21,22,23,24
t94	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	17	2,4,6,8,10,12,16,18,20,2 1,23	11	21,22,23,24
t95	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	18	2,4,6,8,10,12,16,18,20,2 1,23	11	21,22,23,24
t96	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,25,26,43	18	2,4,6,8,10,12,16,18,20,2 2,25,27	12	29,30,31,32
t97	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,25,26,43	19	2,4,6,8,10,12,16,18,20,2 2,25,27	12	29,30,31,32
t98	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,25,26,43	18	2,4,6,8,10,12,16,18,20,2 2,25,27	12	29,30,31,32
t99	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,25,26,43	17	2,4,6,8,10,12,16,18,20,2 2,25,27	12	29,30,31,32
t100	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,25,26,43	18	2,4,6,8,10,12,16,18,20,2 2,25,27	12	29,30,31,32
t101	1,2,3,5,6,12,13,43	8	1,13	2	13,14,15,16
t102	1,2,3,5,6,12,13,43	9	1,13	2	13,14,15,16
t103	1,2,3,5,6,12,13,43	8	1,13	2	13,14,15,16
t104	1,2,3,5,6,12,13,43	8	1,13	2	13,14,15,16
t105	1,2,3,5,6,12,13,43	9	1,13	2	13,14,15,16
t106	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,30,31,33,38, 39,42,43	22	2,4,6,8,10,12,16,18,20,2 2,26,29,32,34,37,40	16	
t107	1,2,3,4,5,6,7,8,9,10,11,1 6,17,18,19,24,30,31,33,3 8,39,40,41,43	24	2,4,6,8,10,12,16,18,20,2 2,26,29,32,34,37,40	16	1,2,3,4
t108	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,30,31,33,38, 39,42,43	22	2,4,6,8,10,12,16,18,20,2 2,26,29,32,34,37,40	16	
t109	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,30,31,33,38, 39,42,43	22	2,4,6,8,10,12,16,18,20,2 2,26,29,32,34,37,40	16	
t110	1,2,3,4,5,6,7,8,9,10,11,1 6,17,18,19,24,30,31,33,3 8,39,40,41,43	24	2,4,6,8,10,12,16,18,20,2 2,26,29,32,34,37,39	16	1,2,3,4,9,10,11, 12
t111	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	15	2,4,6,8,10,12,16,17,23	9	21,22,23,24
t112	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	16	2,4,6,8,10,12,16,17,23	9	21,22,23,24
t113	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	15	2,4,6,8,10,12,16,17,23	9	21,22,23,24

t114	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	15	2,4,6,8,10,12,16,17,23	9	21,22,23,24
t115	1,2,3,5,6,7,8,9,10,11,16, 17,20,21,43	16	2,4,6,8,10,12,16,17,23	9	21,22,23,24
t116	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,20,21,43	17	2,4,6,8,10,12,16,18,20,2 1,23	11	21,22,23,24
t117	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,20,21,43	18	2,4,6,8,10,12,16,18,20,2 1,23	11	21,22,23,24
t118	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,20,21,43	17	2,4,6,8,10,12,16,18,20,2 1,23	11	21,22,23,24
t119	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,20,21,43	17	2,4,6,8,10,12,16,18,20,2 1,23	11	21,22,23,24
t120	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,20,21,43	18	2,4,6,8,10,12,16,18,20,2 1,23	11	21,22,23,24
t121	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,25,26,43	18	2,4,6,8,10,12,16,18,20,2 2,25,27	12	29,30,31,32
t122	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,25,26,43	19	2,4,6,8,10,12,16,18,20,2 2,25,27	12	29,30,31,32
t123	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,25,26,43	18	2,4,6,8,10,12,16,18,20,2 2,25,27	12	29,30,31,32
t124	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,25,26,43	18	2,4,6,8,10,12,16,18,20,2 2,25,27	12	29,30,31,32
t125	1,2,3,5,6,7,8,9,10,11,16, 17,18,19,24,25,26,43	19	2,4,6,8,10,12,16,18,20,2 2,25,27	12	29,30,31,32