operator, changing the value of a constant, changing a data type, etc. A major disadvantage of the mutation-based testing is that it is computationally very expensive, since a large number of possible mutants can be generated.

8.5 DEBUGGING

Once errors are identified, it is necessary to first identify the precise location of the errors and then to fix them. In this section, we will summarize the important approaches that are available to identify the error locations. Each of these approaches has its own advantages and disadvantages and therefore each will be useful in appropriate circumstances.

8.5.1 Debugging Approaches

Brute Force Method

This is the most common method of debugging but also the least efficient of all. In this approach, the program is loaded with print statements to print the intermediate values with the hope that some of the printed values will help to identify the statement in error. This approach becomes more systematic with the use of a symbolic debugger, because values of different variables can be easily checked and break points and watch points can be easily set to test the values of variables effortlessly.

Backtracking

This is also a fairly common approach. In this approach, beginning from the statement at which an error symptom has been observed, the source code is traced backwards until the error is discovered. Unfortunately, as the number of source lines to be traced back increases, the number of potential backward paths also increases and may become unmanageably large thus limiting the use of this approach.

Cause Elimination Method

In this approach, a list of causes which could possibly have contributed to the error symptom is developed and tests are conducted to eliminate each cause. A related technique of identification of the error from the error symptom is the software fault tree analysis.

Program Slicing

This technique is similar to backtracking. However, the search space is reduced by defining slices. A slice of a program for a particular

variable at a particular statement is the set of source lines preceding this statement that can influence the value of that variable.

8.5.2 Debugging Guidelines

The following are some general guidelines for effective debugging:

- Many times, debugging requires a thorough understanding of the program design.
- Debugging may sometimes even require a full redesign of the system. A common mistake that novice programmers often make is attempting not to fix the error but its symptoms.
- We must be beware of the possibility that an error correction may introduce new errors. Therefore after every round of error-fixing, regression testing (see Section 8.9) must be carried out.

8.6 PROGRAM ANALYSIS TOOLS

A program analysis tool usually means an automated tool that takes the source code of a program as input and produces reports regarding several important characteristics of the program, such as the size, complexity, adequacy of commenting, adherence to programming standards, etc. Also, some program analysis tools produce reports regarding the adequacy of the test cases. There are essentially two categories of program analysis tools:

- Static analysis tools
- Dynamic analysis tools