Debugging

- Programming is a complex process, and because it is done by human beings, it often leads to errors.
- Programming errors are called bugs and the process of tracking them down and correcting them is called debugging. Coders should debug the smallest of their modules before moving on. This decreases the number of errors thrown up during the testing phase and reduces testing time and effort significantly.

Types of Errors

- Three types of errors:
  1. Syntax Errors
  2. Runtime Errors
  3. Logical Errors/ Semantic Errors

Syntax errors

- Any violation of rules and poor understanding of programming language results in syntax errors. In effect, syntax errors represent *grammar errors* in the use of the programming language. Common examples are:
  - Misspelled variable and function names
  - Missing semicolons
  - Improperly matches parentheses, square brackets, and curly braces
  - Incorrect format in selection and loop statements

Runtime Errors

- Runtime errors occur when a program with no syntax errors asks the computer to do something that the computer is unable to reliably do. These errors are also called exceptions because they usually indicate that something exceptional (and bad) has happened.
- Common examples are:
  - Trying to divide by a variable that contains a value of zero
  - Trying to open a file that doesn't exist

Semantic/Logical Errors

- As the name itself implies, these errors are related to the logic of the program. These errors occur due to incorrect translation of algorithm into the program, poor understanding of the problem and a lack of clarity of hierarchy of operators.
- Common examples are:
  - Multiplying when you should be dividing
  - Adding when you should be subtracting
  - Opening and using data from the wrong file
  - Displaying the wrong message

Documentation

- Program documentation includes hard-copy or electronic manuals that enable users, program developers, and operators to interact successfully with a program.
- The program documentation is a kind of documentation that gives a comprehensive procedural description of a program.
- The program documentation describes what exactly a program does by mentioning about the requirements of the input data and the effect of performing a programming task.

Documentation

- *Internal documentation* is written in a program as *comments*.
- *External documentation* is written in a place where people who need to use the software can read about how to use the software. External documentation can be broken down into *library documentation*, which describes tools that a programmer can use, and *user documentation*, which is intended for users of an application.

Some guidelines for creating the documents

- Documentation should be from the point of view of the reader
- Document should be unambiguous
- There should be no repetition
- Industry standards should be used
- Documents should always be updated
- Any outdated document should be phased out after due recording of the phase out

Advantages of Documentation

- Keeps track of all parts of a software or program
- Maintenance is easier
- Programmers other than the developer can understand all aspects of software
- Improves overall quality of the software
- Assists in user training
- Ensures knowledge de-centralization, cutting costs and effort if people leave the system abruptly

Example Documents

- A software can have many types of documents associated with it. Some of the important ones include −
- **User manual** − It describes instructions and procedures for end users to use the different features of the software.
- **Operational manual** − It lists and describes all the operations being carried out and their inter-dependencies.
- **Design Document** − It gives an overview of the software and describes design elements in detail. It documents details like **data flow diagrams, entity relationship diagrams**, etc.
- **Requirements Document** − It has a list of all the requirements of the system as well as an analysis of viability of the requirements. It can have user cases, reallife scenarios, etc.

Example Documents

- **Technical Documentation** – It is a documentation of actual programming components like algorithms, flowcharts, program codes, functional modules, etc.
- **Testing Document** – It records test plan, test cases, validation plan, verification plan, test results, etc. Testing is one phase of software development that needs intensive documentation.