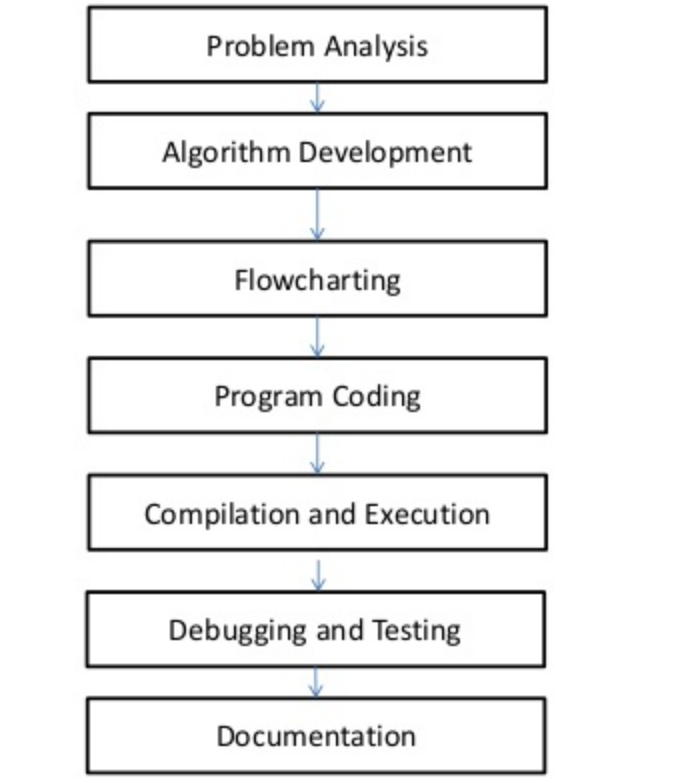
Errors, debugging and Documentation

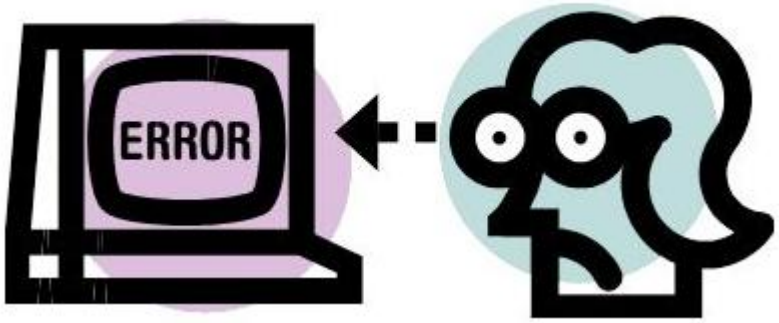
Dr. Priyakshi Mahanta
Assistant Professor
CCSA, Dibrugarh University



Steps in Problem soving

Program Errors

- Compiler errors (syntax errors)
- Runtime errors
- Logic errors



Compiler Errors

- *Syntax error*
  - Detected by the compiler
  - A program with compilation errors cannot be run
- *Syntax error(example)*
  - Forgetting a semicolon

syntax-error.c: In function 'main':syntax-error.c:7:5: error: expected ';' before 'return' **return** 0;

  - Leaving out a closing bracket }
  - Redeclaring a variable

Compiler Errors

- Hints to help find/fix compiler errors:
  - Compiler errors are cumulative
  - Read the error messages issued by the compiler!
  - Realize that the error messages from the compiler are often not very helpful

Rule of thumb

- When an interpreter/compiler gives you a line number, it doesn't necessarily mean that line is incorrect
- If a compiler spits out multiple errors/warnings, start by trying to resolve *only* the first one.
- Treat warnings as seriously as compiler errors.

Logic Errors

- **Logic error**: program runs but results are not correct
- Logic errors can be caused by:
  - incorrect algorithms

- Very common logic errors are:
  - using == instead of the *equals* method
  - infinite loops
  - misunderstanding of operator precedence
  - starting or ending at the wrong index of an array
    - If index is invalid, you would get an exception
  - Misplaced parentheses

Runtime Errors

- ***Runtime error***: program runs but gets an *exception* error message
  - Program may be terminated
- Runtime errors can be caused by
  - Program bugs
  - Bad or unexpected input
  - Hardware or software problems in the computer system

Runtime Errors

- Very common runtime errors are:
  - null reference (NullPointerException)
  - array index out of bounds (ArrayIndexOutOfBoundsException)
  - Running out of memory
  - Floatingpointexception
  - Segmentation fault

Debugging Techniques

- Reactive techniques
  - *print debugging*
- Preemptive techniques
  - *Assertions*
  - *Logging*
- **Debugger**

Documentation

- Textual information within a program
- Helps convey the meaning of the program to human readers who need to understand, modify, or debug the program.

Forms of documentation

- Inherent documentation
- Inline documentation

(In Java, everything from a // until the end of a line is considered a comment)

- Header comments
- External documentation
- User manual

Thank You

- References

1)https://www.cse.wustl.edu/~cytron/101Pages/f08/Notes/Debugging/debugging.html

2)https://www.csd.uwo.ca/Courses/CS1027b/notes

3) https://en.wikipedia.org/wiki/Debugging