5. Program Design Techniques

Lesson 1:
An overview of program design

5.1. Program design

5.1.1. Importance of professional programming

- In most cases software is used in large and complex systems
- Software is usually maintained by people other than the original author
- Software needs to last for many years and must be continually enhanced and amended to meet the requirements of changing environments

5.1. Program design

5.1.2. Tools for Program Design

- A good design tool will lead to programs which are :
  - Elegant
  - Accurate
  - Easy to amend and enhance over a long period of time
  - Unambiguous

5.1. Program design

5.1.3. Evolution of software design

- We will take a look at four stages in the evolution of software design
  - Programming as an Art Form
  - Modular Programming
  - Structured Programming
  - Object-Oriented Programming (OOP)

5.1. Program design

5.1.3. Evolution of software design (cont...)

- Programming as an Art Form
  - Some forty years ago , the main preoccupation was with the correctness of the program.
  - The design tool that emerged from this period was the flowchart.
    - These are a diagrammatic representation of the flow of logic within a program or within an individual process.
  - Main concern was with logic flow.

5.1. Program design

5.1.3. Evolution of software design (cont...)

- Modular Programming
  - In early 60s, program designers stressed the importance of modular programming
  - This was a process of top-down decomposition
  - Main concern was with the structure of the program as well as the logic flow

5.1. Program design

5.1.3. Evolution of software design (cont...)

- Structured Programming
  - In 1966 Bohm and Jacopini demonstrated that any program is a combination of sequence, selection and iteration

- Object-Oriented Programming (OOP)
  - Emerging major programming paradigm
  - Main concern is an Object which is a particular instance of a class which consist of data and methods

5.1. Program design

5.1.4. Tools for algorithm specification

- Algorithms Specification consists of specifying accurately and clearly what exactly the algorithm, should do

- We will take a look at three tools for algorithm specification
  - Flowcharts
  - Nassi - Shneiderman (NS) Diagrams
  - Pseudocode

5.1. Program design

5.1.4. Tools for algorithm specification (cont…)

- Flowcharts
  - Are a diagrammatic representation of a process
  - Use arrows to indicate the flow of logic

- Nassi - Shneiderman (NS) Diagrams
  - Represent the flow of logic
  - Removes of necessity to use arrows

5.1. Program design

5.1.4. Tools for algorithm specification (cont…)

- Pseudocode
  - Formulates the solution to a problem, independent of the syntax of a programming language
  - It is essential to be able to describe a problem solution to other programmers who may not understand the programming language normally used by developer of that solution
  - The solution to a problem must be able to be expressed with the precision of a programming language but without concentrating too much on the finer points of its syntax.

5.1. Program design

5.1.4. Tools for algorithm specification (cont…)

Flowcharts	**NS Diagrams**	**Pseudocode**
Easy to follow	Easy to follow	Easier to translate into a program
Good for simple procedures	Good for simple procedures	Easier to computerise
Poor illustration of program structure	Absence of arrows ensure strict adherence to structured programming concepts	Universally accepted as a basis of algorithm specification
No provision for data definition and scoping	No body really uses them as design tools	Harder to understand and poor for designing structure