Programming Languages

- Problem Solving and Algorithm Design
    - Problem Solving
    - Algorithms
    - Pseudocode
    - Top-Down Design Methodology
    - Object Oriented Methodology

Problem Solving

- The act of finding a solution to a perplexing, distressing, vexing or unsettled question.
- How to solve a problem ?
  - You have to understand the problem.
  - After understanding, then make a plan.
  - Carry out the plan.
  - Look back on your work.
- Taken from How to Solve It (1945) by mathematician George Polya.

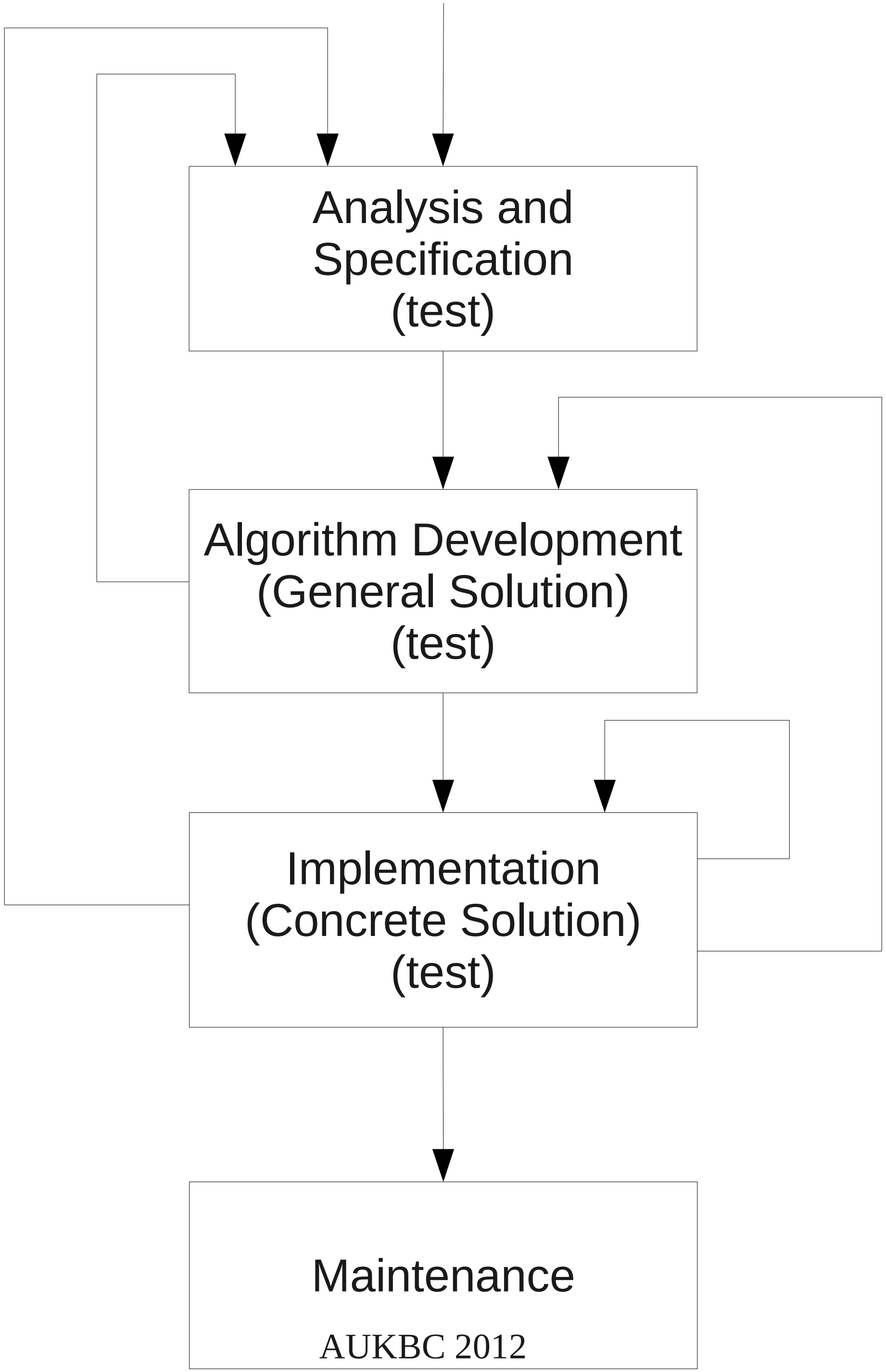
Problem Solving

- The steps outlined are general enough to cover any problem.
- Ask questions regarding the problem.
- See whether the problem can be compared to something you already know.
- Divide and Conquer – break the problem into smaller problems which can be solved more easily than the whole problem.

Algorithms

- In computing the plan to solve a problem is called an algorithm.
- Algorithm – Unambiguous instructions for solving a problem or subproblem in a finite amount of time using a finite amount of data.
- All conditions necessary for solving the problem must be made explicitly.
- The Computer Problem-Solving Process can be now formulated.

Problem Solving Phases


Analysis and Specification (test)
Algorithm Development (General Solution) (test)
Implementation (Concrete Solution) (test)
Maintenance

Algorithms

- Most of the time we experience algorithms in the context of following them.
- Developing algorithms is usually done first using pseudocode – a mixture of English and indenting to make the steps of the solution explicit.
- The algorithm is then converted into a plan using a design process.

Pseudocode

- Pseudocode can be written using some common constructs.
  - Variables – Represent named places into which values are stored and from which values are retrieved.
  - Assignment – Storing a value into a variable.
  - Input / Output – Input : reading in a value, probably from the keyboard. Output : Displaying the contents of a variable, probably on the screen.

Pseudocode

- Repetition (iteration, looping) – Repeat one or more statements as long as a condition is true.
- Selection if-then – If a condition is true, execute the indented statements; if a condition is not true, skip the indented statements.
- Selection if-then-else – If a condition is true, execute the indented statements; if a condition is not true, execute the indented statements below else.

Pseudocode

```
Write “Number of pairs ?”
Read numberOfPairs
Set numberRead to 0
While (numberRead < numberOfPairs)
    Write “Enter 2 values, separated by blank; press return”
    Read number1
    Read number2
    If (number1 < number2)
        Print number1 + “ “ + number2
    Else
        Print number2 + “ “ + number1
    Increment numberRead
```

Top-Down Design Methodology

- A technique for developing a program in which the problem is divided into more easily handled sub problems, the solutions of which create a solution to the overall problem.
- Module – A self-contained collection of steps that solves a problem or a sub problem.
- Abstract step – An algorithmic step for which some details remain unspecified.
- Concrete step – A step for which the details are fully specified.

Top-Down Design Methodology

- Consists of four major steps.
  - Analyze the problem.
  - Write the Main Module – In english or pseudocode without much detail. Give names to lower level modules that solves certain tasks.
  - Write the remaining modules – Write the lower level modules using multiple levels if required.
  - Re-sequence and Revise as necessary.
- Also known as functional decomposition.

Top-Down Design Methodology

- The goal of computer problem solving is to create the right process to solve the problem
- The process must be tested or verified.
- Problems discovered earlier are easier to address.
- Desk Checking – Tracing the execution of a design on paper.
- Walk-through – A manual simulation of a design, performed by a team.
- Inspection – A non designer reads the design line by line while others point out mistakes.

Object-Oriented Methodology

- A problem-solving methodology that produces solutions to problems in terms of self-contained entities called objects, which are composed of both data and operations that manipulate the data.
- Object – An entity or thing that is relevant in the context of the problem.
- Object class (class) – A description of a group of objects with similar properties and behaviors.

Object-Oriented Methodology

- Fields – Named items in a class; can be data or sub programs.
- Method – A named algorithm that defines one aspect of the behavior of a class.
- Object Oriented Design (OOD) – has gained popularity in recent years.

Object Oriented Methodology

- Consists of four stages:
  - Brainstorming – First pass at determining the classes in the problem.
  - Filtering – Second pass over the classes identified at brainstorming to see if any can be combined or if any are missing.
  - Scenarios – Stage in which behavior of each class is determined.
    - Behaviors are called responsibilities of the class.
  - Responsibility algorithms – Stage in which the algorithms are written for the responsibilities for each of the classes.