PROBLEM SOLVING

Robert Burtch
Surveying Engineering Department
Ferris State University

PROGRAM DEVELOPMENT CYCLE

Performing tasks on computer

1. What is the output?
2. What is the input?
3. Define the process of taking the input to obtain the desired output

PROGRAM DEVELOPMENT CYCLE

- Example: How fast is a car traveling if it goes 50 miles in 2 hours?
  - Output: speed or rate in mph
  - Input: distance and time
  - Process: formula is $\text{rate} = \frac{\text{distance}}{\text{time}}$
  - Graphically:

PROGRAM DEVELOPMENT CYCLE

- Program planning
  1. Analyze – define the problem
  2. Design – plan the solution to the problem
     - algorithm– sequence of precise steps to solve problem
  3. Choose interface – select objects (text boxes, command buttons, etc.)
  4. Code – translate algorithm into programming language
  5. Test and debug
  6. Complete documentation

PROGRAMMING TOOLS

- Flow charts
  - Graphically depict logical steps to carry out a task and show how steps relate to each other
- Pseudocode
  - Uses English-like phrases with some VB terms to outline task
- Hierarchy charts
  - Show how different parts of program relate to each other

FLOW CHART

flow chart also **flow·chart** (fl chärt) *n.*

A schematic representation of a sequence of operations, as in a manufacturing process or computer program. Also called **flow diagram, flow sheet**.

Source: *The American Heritage® Dictionary of the English Language, Fourth Edition*

Flowchart

n : a diagram of the sequence of operations in a computer program or an accounting system [syn: flow chart, flow diagram, flow sheet]

Source: *WordNet ® 2.0, © 2003 Princeton University*

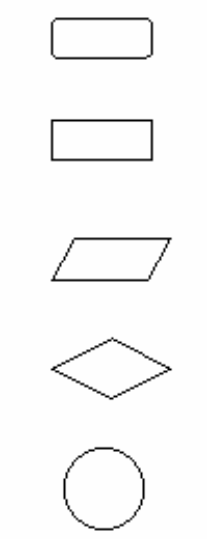
FLOW CHART

Flowchart

n. [techspeak] An archaic form of visual control-flow specification employing arrows and `speech balloons' of various shapes. Hackers never use flowcharts, consider them extremely silly, and associate them with COBOL programmers, card wallopers, and other lower forms of life. This attitude follows from the observations that flowcharts (at least from a hacker's point of view) are no easier to read than code, are less precise, and tend to fall out of sync with the code (so that they either obfuscate it rather than explaining it, or require extra maintenance effort that doesn't improve the code). See also PDL, sense 1.

Source: *Jargon File 4.2.0*

FLOW CHART



Start or end of the program

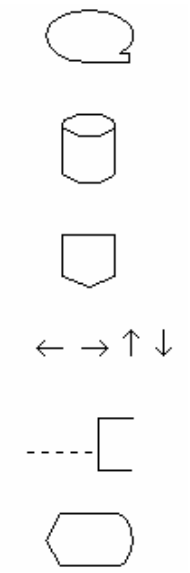
Computational steps or processing function of a program

Input or output operation

Decision making and branching

Connector or joining of two parts of program

FLOW CHART



Magnetic Tape

Magnetic Disk

Off-page connector

Flow line

Annotation

Display

FLOW CHART

- Should flow from top of page to bottom

- Limitations
  1. Complex logic: if program logic is complicated flowchart becomes complex and clumsy.
  2. Alterations and Modifications: flowchart may require re-drawing completely.
  3. Reproduction: flowchart symbols cannot be typed, reproduction becomes a problem.
  4. The essentials of what is done can easily be lost in the technical details of how it is done.

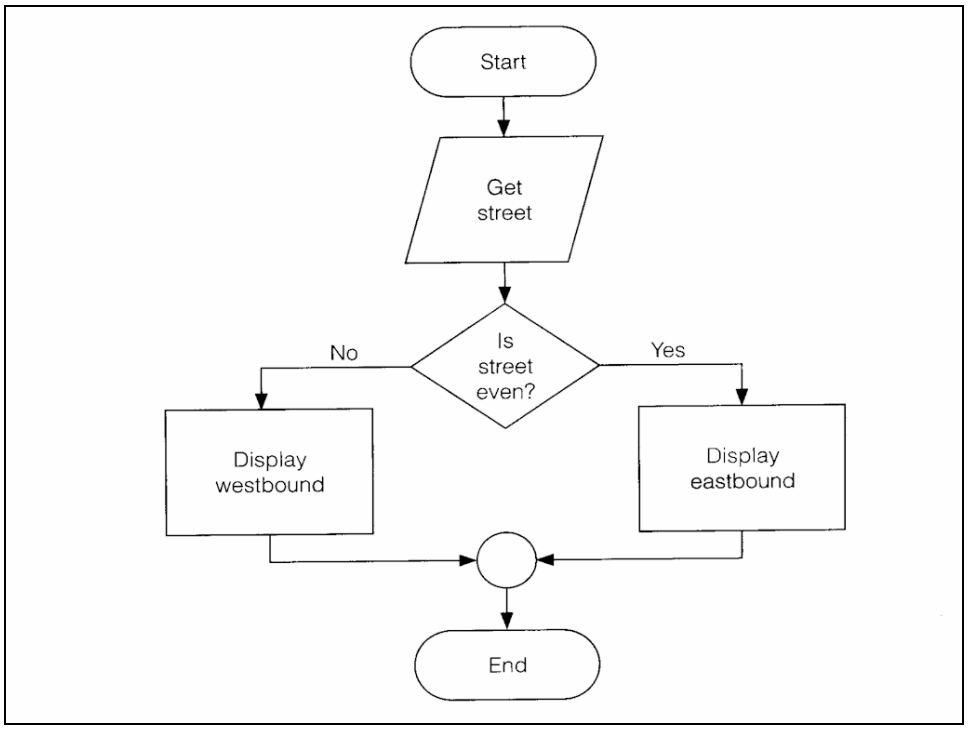
FLOW CHART

- Advantages
  1. Communication: better way of communicating the logic to all concerned.
  2. Effective analysis: problem can be analyzed more effectively
  3. Proper documentation: serve as a good program documentation
  4. Efficient Coding: act as a guide or blueprint during the systems analysis and program development phase.
  5. Proper Debugging: helps in debugging process.
  6. Efficient Program Maintenance: maintenance of operating program becomes easier with the help of flowchart

EXAMPLE

- Problem: Given a street number of a one-way street in New York, decide the deirection of the street, either eastbound or westbound

- Simple rule to tell direction – even-numbered streets run eastbound
- Input: street number
- Processing: see if street number divisible by 2
- Output: “Eastbound” or “Westbound”



PSEUDOCODE

- Symbols in flowchart replaced by English-like statements outlining process
- Allows programmer to focus on steps required to solve problem
- When completed – translated into VB
- Advantages
  - Compact – probably will not extend for pages
  - Plan looks like code to be written

EXAMPLE

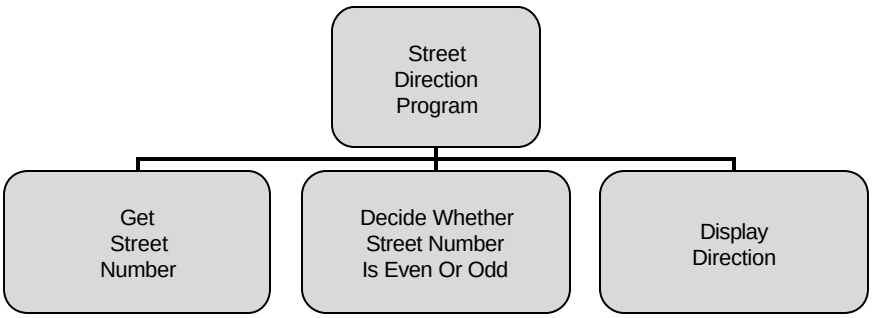
Program: Determine the direction of a numbered NYC street

```
Get Street
If Street is even Then
      Display Eastbound
   Else
      Display Westbound
   End If
```

HIERARCHY CHARTS

- Show overall program structure
- Show organization of program without specific processing steps
- Describe what each part/module does & how they relate
  - Details missing
- Advantage
  - Break down major parts – divide-and-conquer method

EXAMPLE


Street
Direction
Program
Get
Street
Number
Decide Whether
Street Number
Is Even Or Odd
Display
Direction