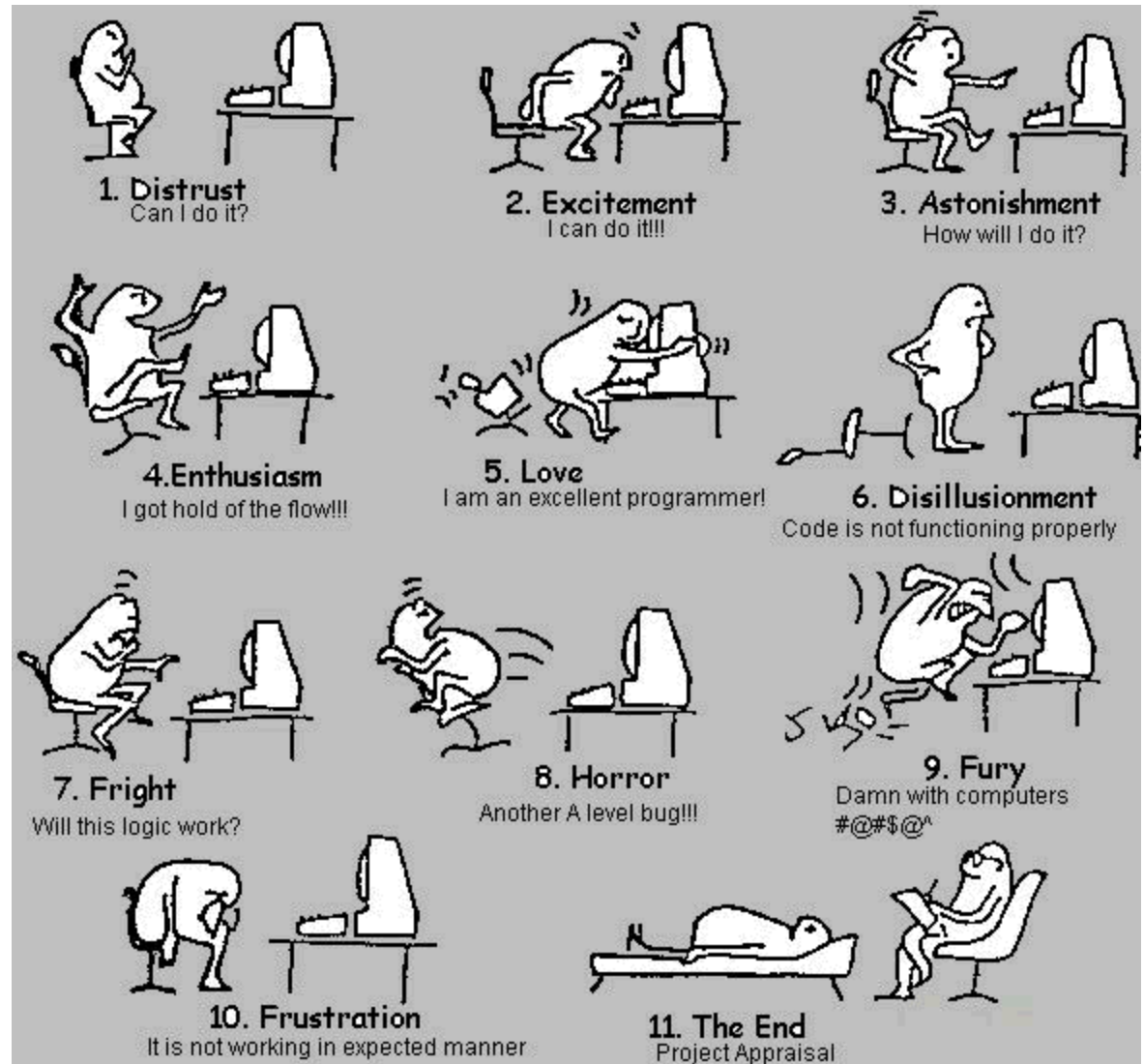


WELCOME TO CMPT 110



- Instructor: Richard Frank – rfrank@sfu.ca
- TA: Kyle Demeule – kdd2@sfu.ca
- CMPT 110 (D100) Programming in Visual Basic
- Class Hours
 - Tuesday: 10:30am-11:20am @ AQ 3005
 - Thursday: 9:30am-11:20am @ C 9000
- Office: TBD
- Office Hours: Tuesday 9:30am – 10:20am
- <http://www.cs.sfu.ca/CourseCentral/110/rfrank/>

CALENDAR OBJECTIVE/DESCRIPTION

- Topics will include
 - User interfaces
 - Objects
 - Event-driven programming
 - Program design
 - File and data management

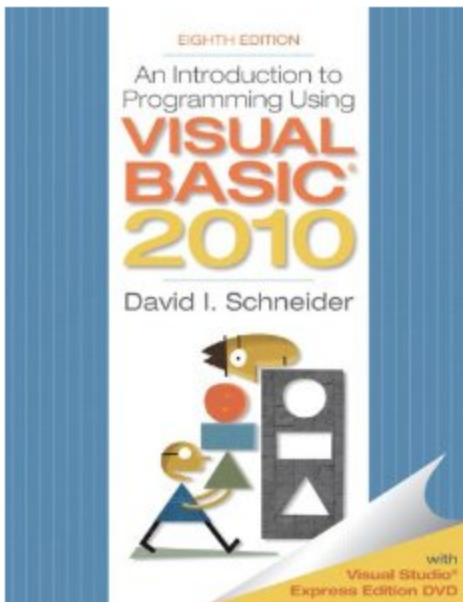
OBJECTIVES

- Introduction to programming in the event-driven paradigm using the Visual Basic language.
- We'll cover
 - Forms
 - Controls
 - Events
 - Menus
 - Objects
 - Subprograms
 - Modular design
 - Decisions and repetition
 - File and data management
 - Special features
- *This is an entry-level course, not a developer's seminar.*

- Assignments 30%
 - 5 assignments
- Midterm exam 30%
 - Week 7 - Oct 16
- Final exam 40%
 - As per SFU date/location


REQUIRED BOOKS

- An Introduction to Programming Using Visual Basic 2010, (w/VS2010 DVD), 8/E, D.I. Schneider , Prentice-Hall, 2010
- Text comes with DVD to install VB at home



Introduction to Programming Using Visual Basic 2010 [Paperback]

[David I. Schneider](#) (Author)

[Be the first to review this item](#)  (0)

List Price: ~~CDN\$ 137.55~~

Price: **CDN\$ 128.77** & this item ships for **FREE with Super Saver Shipping**.

[Details](#)

You Save: **CDN\$ 8.78 (6%)**

In Stock.

Ships from and sold by **Amazon.ca**. Gift-wrap available.

Only 2 left in stock--order soon (more on the way).

Want it delivered Tuesday, September 4? Order it in the next **51 hours and 21 minutes**, and choose **Priority Shipping** at checkout. This is available for most major cities. Please confirm the estimated delivery date when you choose a shipping option for your order.

15 new from **CDN\$ 127.11** **16 used** from **CDN\$ 114.41**

ACADEMIC HONESTY STATEMENT

- Academic honesty plays a key role in our efforts to maintain a high standard of academic excellence and integrity. Students are advised that ALL acts of intellectual dishonesty will be handled in accordance with the SFU Academic Honesty and Student Conduct Policies (<http://www.sfu.ca/policies/Students/index.html>). Students are also encouraged to read the School's policy information page (<http://www.cs.sfu.ca/undergrad/Policies/>).
- Cheaters will be caught → 0.

Week	Chapter	Textbook Reading	Activities
1 Sep 4	Chapter 1: Introduction to Computers and Basics	Section 1 Appendix B	Review "Preparation of Assignments"
2 Sep 11	Chapter 2: Controls and Events	Sections 2.1-2.3 Appendix D	Exercise 2.2 (5, 8, 12, 14, 26, & 29) Exercise 2.3 (11, 13, 15, 33, & 38)
3 Sep 18	Chapter 3: Variables, Input, and Output	Sections 3.1-3.3	Exercise 3.1 (1-22, 39-44) Exercise 3.2 (5, 7, 11, 13, 23, 27-32)
4 Sep 25	Chapter 4: Decisions	Sections 4.1-4.3	Exercise 4.2 (13-20, 22-24, 34, & 37) Exercise 4.3 (9-22, 24, 25, 29, & 31) Assignment 1 due
5 Oct 2	Chapter 5: General Procedures	Sections 5.1-5.5	Exercise 5.1 (28, 29, 31, 35-40) Exercise 5.2 (19, 20, 23, & 24) Exercise 5.3 (11, 12, 15, & 16)
6 Oct 9	Chapter 6: Repetition	Sections 6.1-6.4	Exercise 6.1 (7-10, 21, 28, & 38) Exercise 6.3 (13-18, 24, & 29) Assignment 2 due
7 Oct 16	Midterm exam		

8 Oct 23	Chapter 7: Arrays	Sections 7.1-7.6	Exercise 7.1 (13-18, 21, 29, 30, 33, & 38) Exercise 7.2 (9-14) Exercise 7.3 (9-11) Exercise 7.5 (11-14)
9 Oct 30	Chapter 8: Sequential Files	Sections 8.1-8.3	Exercise 8.1 (13-18) Exercise 8.2 (15, 16, & 18) Assignment 3 due
10 Nov 6	Chapter 9: Additional Controls and Objects	Sections 9.1-9.4	Exercise 9.3 (42) Exercise 9.4 (1-4, 9, & 10)
11 Nov 13	Chapters 10: Database Management	Sections 10.1-10.2	Exercise 10.1 (1-7) Exercise 10.2 (1-12) Assignment 4 due
12 Nov 20	Chapters 11: Object-Oriented Programming	Sections 11.1-11.3	Exercise 11.1 (1-4, 8-14, 21) Exercise 11.2 (1-5) Exercise 11.3 (7-16)
13 Nov 27	Chapter 12: Programming for the Web Review	Section 12	Exercise 12.1 (2, 4, 18) Exercise 12.2 (8, 12) Exercise 12.3 (2, 12) Assignment 5 due
			Final exam

YOUR BACKGROUND?

- Any programming?
- “Expert” at Windows?
- Excel formulas (if, lookup, ...)?
- Installing programs?



S. ADAMS E-mail: SCOTTADAMS@AOL.COM



6/22 © 1995 United Feature Syndicate, Inc. (NYC)



CHAPTER 1 - INTRO

CHAPTER 1 - AN INTRODUCTION TO COMPUTERS AND PROBLEM SOLVING

- 1.1 An Introduction to Computers
- 1.2 Windows, Folders, and Files
- 1.3 Program Development Cycle
- 1.4 Programming Tools

COMMUNICATING WITH THE COMPUTER

- Machine language
 - low level, hard for humans to understand
- Visual Basic
 - high level, understood by humans, consists of instructions such as Click, If, Do
 - Usable in other applications (Word, Excel...)

COMPUTERS AND COMPLICATED TASKS

- Tasks are broken down into instructions that can be expressed by a computer language
- A *program* is a sequence of instructions
- Programs can be only a few instructions or millions of lines of instructions
- Examples?
 - In real life?
 - In computers?

ALL PROGRAMS HAVE IN COMMON

- Take data and manipulate it to produce a result
- Input – Process – Output
 - Input – from files, the keyboard, or other input device
 - Output – to the monitor, printer, file, or other output device

HARDWARE AND SOFTWARE

- Hardware

- The physical components of a computer
 - Keyboard
 - Disk drive
 - Monitor

- Software

- The instructions that tell the computer what to do

PROGRAMMER AND USER

- Programmer – the person who solves the problem and writes the instructions for the computer
- User – any person who uses the program written by the programmer



PROBLEM SOLVING

- Developing the solution to a problem
- Algorithm – a step by step series of instructions to solve a problem

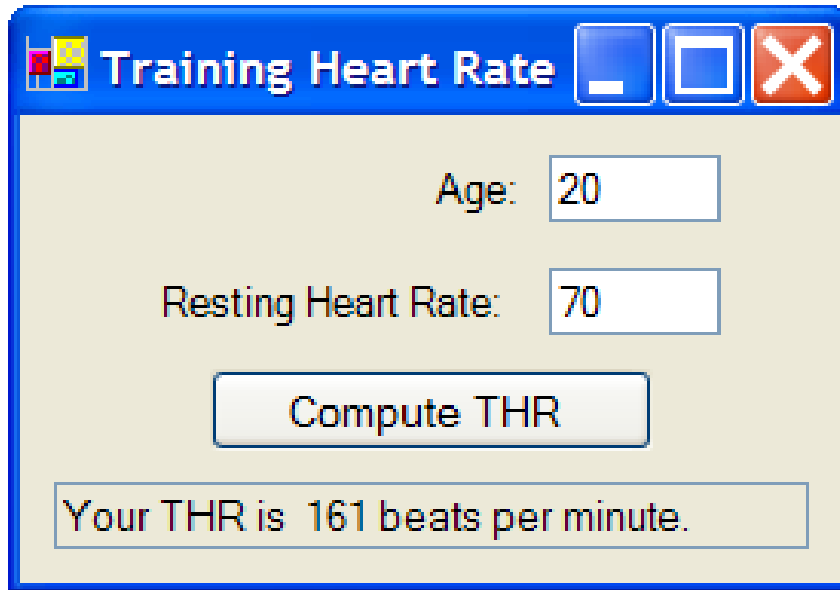
PROBLEM SOLVING

- Problems are solved by carefully reading them to determine what data are given and what outputs are requested
- Then a step-by-step procedure is devised to process the given data and produce the requested output
- This procedure is called an **algorithm**
- Finally, a computer program is written to carry out the algorithm

- BASIC originally developed at Dartmouth in the early 1960s
- Visual Basic created by Microsoft in 1991
- Visual Basic 2010 is similar to original Visual Basic, but more powerful

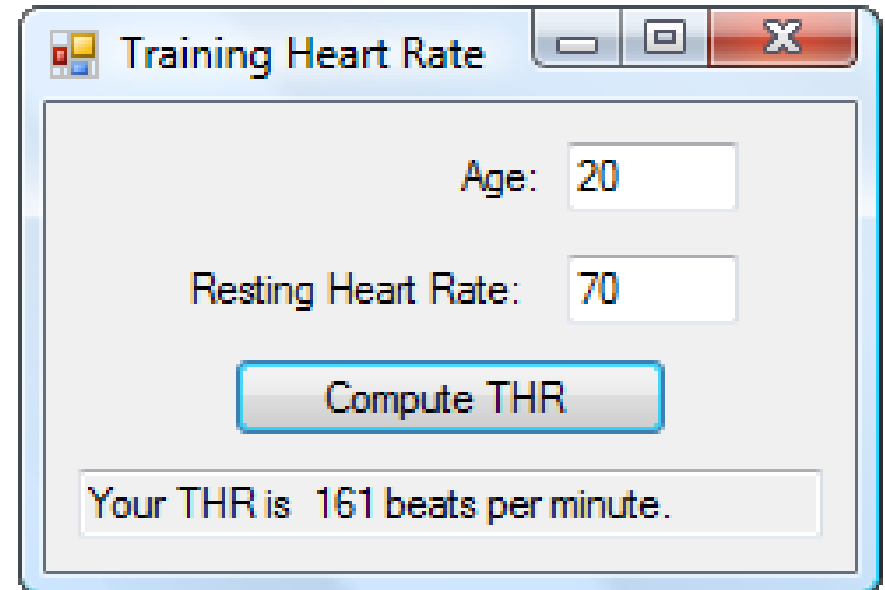
```
Private Sub Timer1_Timer()  
    x = x + 10  
    Dim a, i As Integer  
    Randomize(Timer)  
  
    For i = 0 To 8  
  
        ' To generate random integers 0,1 and 2  
        a = Int(Rnd * 3)  
        Shape1(i).Shape = a  
        If a = 0 Then  
  
            Shape1(i).FillColor = vbRed  
        ElseIf a = 1 Then  
            Shape1(i).FillColor = vbGreen  
        Else  
            Shape1(i).FillColor = vbBlue  
        End If  
    Next i  
  
    'To stop the timer  
    If x > 500 Then  
        Timer1.Enabled = False  
    End If  
End Sub
```

XP VERSUS VISTA



A screenshot of a Windows XP application window titled "Training Heart Rate". The window has a blue title bar with standard minimize, maximize, and close buttons. The main area has a tan background. It contains two input fields: "Age: 20" and "Resting Heart Rate: 70". Below these is a "Compute THR" button. At the bottom, a text box displays "Your THR is 161 beats per minute."

Windows XP



A screenshot of the same "Training Heart Rate" application window in Windows Vista. The window has a light blue title bar with standard minimize, maximize, and close buttons. The main area has a light gray background. It contains two input fields: "Age: 20" and "Resting Heart Rate: 70". Below these is a "Compute THR" button. At the bottom, a text box displays "Your THR is 161 beats per minute."

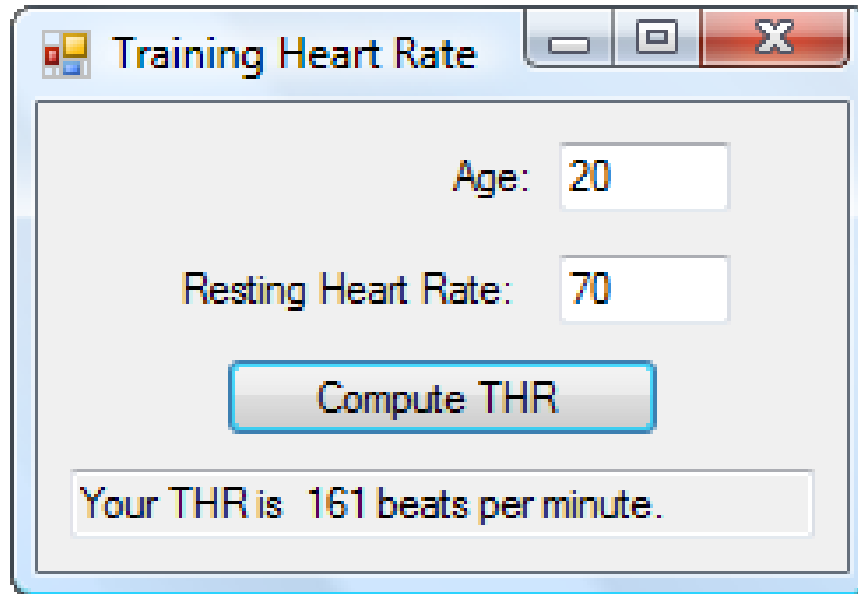
Windows Vista

1.2 WINDOWS, FOLDERS, AND FILES

- Windows and Its Little Windows
- Mouse Actions
- Files and Folders

WINDOWS AND ITS LITTLE WINDOWS

- Difference between *Windows* and *windows*.
- Title bar indicates if window is active.



- **Clicking** (single-clicking) means pressing and releasing the left mouse button once.
- **Double-clicking** means clicking the left mouse button twice in quick succession
 - **Note:** An important Windows convention is that clicking selects an object so you can give Windows or the document further directions about it, but double-clicking tells Windows to perform a default operation.

MOUSE ACTIONS

- **Pointing** means moving your mouse across your desk until the mouse pointer is over the desired object on the screen
- **Hovering** means to linger the mouse at a particular place and wait for a message or menu to appear
- **Dragging** usually moves a Windows object. If you see a sentence that begins “Drag the . . .”, you need to click on the object and hold

FILES AND FOLDERS

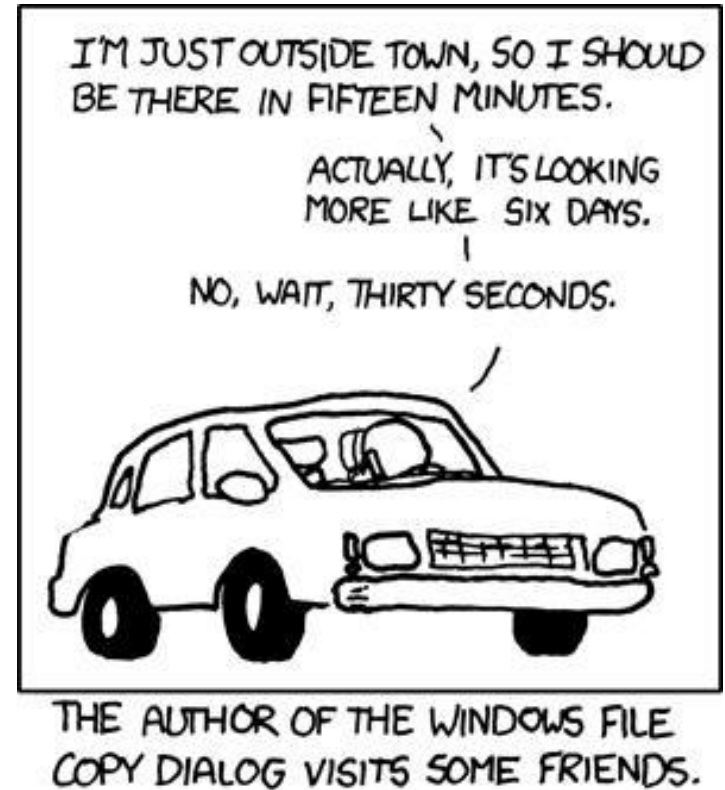
- File: holds programs or data. Its name usually consists of letters, digits, and spaces.
- Folder: contains files and other folders (called subfolders).

KEY TERMS IN USING FOLDERS AND FILES

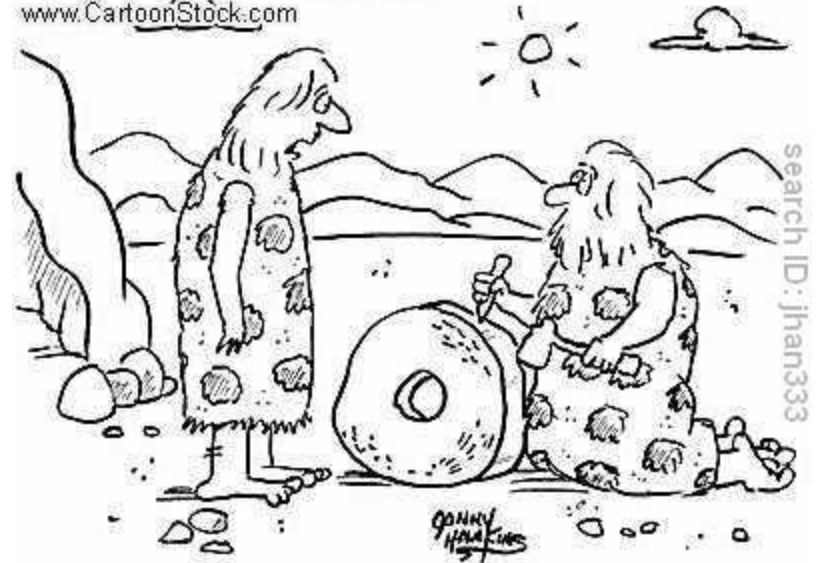
Term	Example
Disk	Hard disk, CD
File name	PAYROLL
Extension	.TXT
Filename	PAYROLL.TXT
Path	TextFiles\PAYROLL.TXT
Filespec	C:\TextFiles\PAYROLL.TXT

WINDOWS EXPLORER

- Used to view, organize and manage folders and files.
- Manage: copy, move, delete



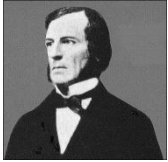
© Original Artist
Reproduction rights obtainable from
www.CartoonStock.com



"Nice invention - how do you boot it up?"

BIOGRAPHICAL HISTORY OF COMPUTING

29



- George Boole – devised Boolean algebra



- Charles Babbage – created "analytical engine"



- Augusta Ada Byron – first computer programmer



- Herman Hollerith – founder of company that would become IBM



- **Alan Turing** – deciphered German code in WWII; father of artificial intelligence



- **John V. Atanasoff** – inventor of first electronic digital special purpose computer



- **Howard Aiken** – built large scale digital computer, Mark I



- **Grace M. Hopper** – originated term "debugging"; pioneered development and use of COBOL



- **John Mauchley and J. Presper Eckert** – built first large scale general purpose computer, ENIAC

1940S CONTINUED



- **John von Neumann** – developed stored program concept



- **Maurice V. Wilkes** – built EDSAC, first computer to use stored program concept



- **John Bardeen, Walter Brattain, and William Shockley** – developed transistor that replaced vacuum tubes



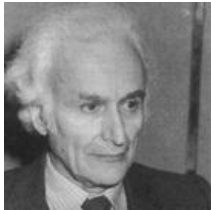
- **John Backus** – created Fortran; early user of interpreters and compilers



- **Reynold B. Johnson** – invented the disk drive
- **Donald L. Shell** – developed efficient sorting algorithm



- **John G. Kemeny and Thomas E. Kurtz** – invented BASIC



- **Corrado Bohm and Guiseppe Jacopini** – proved that any program can be written with only 3 structures: sequence, decision, and loops

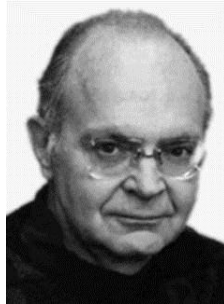


- **Edsger W. Dijkstra** – stimulated move to structured programming by declaring "GOTO" harmful

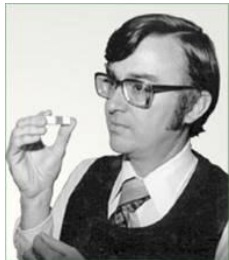
1960S CONTINUED



- **Harlan B. Mills** – advocated use of structured programming



- **Donald E. Knuth** – wrote definitive work on algorithms.

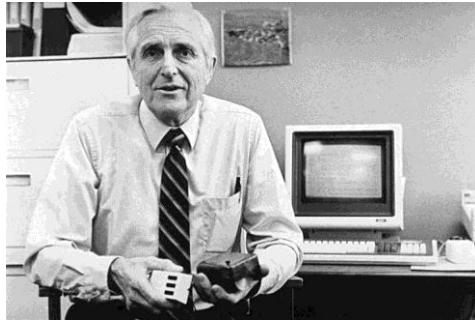


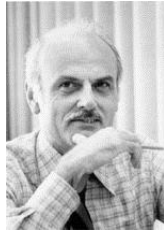
- **Ted Hoff, Stan Mazer, Robert Noyce, and Frederico Faggin** – developed first microprocessor



1960S CONTINUED

- **Douglas Engelbart** – invented computer mouse





- **Ted Codd** - software architect; laid the groundwork for relational databases



- **Paul Allen and Bill Gates** - cofounders of Microsoft Corporation



- **Stephen Wozniak and Stephen Jobs** - cofounders of Apple Computer Inc.



- **Dan Bricklin and Dan Fylstra** - wrote VisiCalc, the first electronic spreadsheet program

1970S CONTINUED



- **Dennis Ritchie** - creator of the C programming language.



- **Ken Thompson** - created the Unix operating system



- **Alan Kay** – developer of Smalltalk, a pure object-oriented language



- **Don Chamberlain** - created a database programming language, later known as SQL,



- **Phillip “Don” Estridge** - at IBM directly responsible for the success of the personal computer.



- **Mitchell D. Kapor** - cofounder of Lotus Corporation



- **Tom Button** - group product manager for applications programmability at Microsoft;
 - headed the team that developed QuickBasic, QBasic, and Visual Basic.

1980S CONTINUED



- **Alan Cooper** - considered the father of Visual Basic.



- **Tim Berners-Lee** - father of the World Wide Web.



- **Charles Simonyi** - father of Word.



- **Bjarne Stroustrup** - creator of the C++ programming language.



- **Richard M. Stallman** - founded Free Software Foundation



- **Marc Andreessen** - inventor of the Web browser.



- **James Gosling** – creator of Java.



- **Linus Torvalds** - developed the popular Linux operating system.



- **Sergey M. Brin and Lawrence E. Page** – founders of Google



- **Mark Zuckerberg** – founder of Facebook.



- **Steve Chen, Chad Hurley, and Jawed Karim** – founders of YouTube.



1.3 PROGRAM DEVELOPMENT CYCLE

- Performing a Task on the Computer
- Program Planning



- A computer program may also be called:
 - Project
 - Application
 - Solution

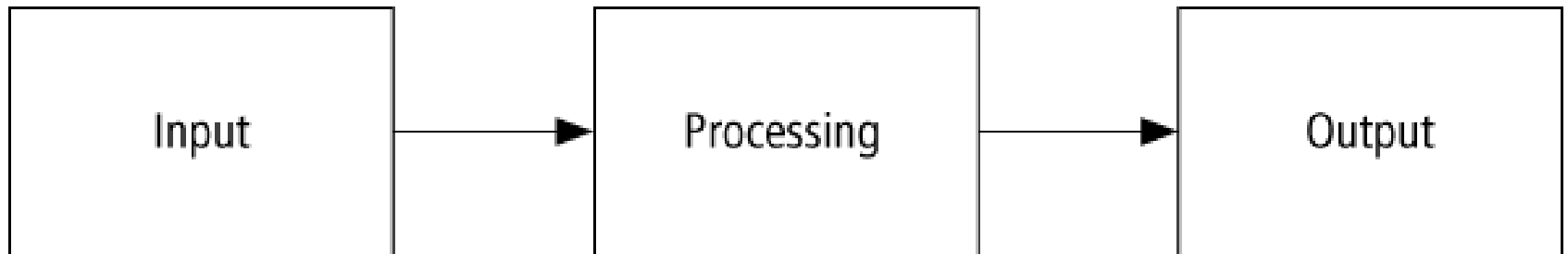
PROGRAM DEVELOPMENT CYCLE

- Software refers to a collection of instructions for the computer
- The computer only knows how to do what the programmer tells it to do
- Therefore, the programmer has to know how to solve problems
 - Take big problem, break it down
 - Break it down further
 - Repeat until you get to very fundamental steps

PERFORMING A TASK ON THE COMPUTER

- Determine Output
- Identify Input
- Determine process necessary to turn given Input into desired Output

PICTORIAL REPRESENTATION OF THE PROBLEM SOLVING PROCESS



PROBLEM-SOLVING: APPROACH LIKE ALGEBRA CLASS

- How fast is a car traveling if it goes 50 miles in 2 hours?
- **Output:** a number giving the speed in miles per hour
- **Input:** the distance and time the car has traveled
- **Process:** $\text{speed} = \text{distance} / \text{time}$

PROGRAM PLANNING

- A recipe is a good example of a plan
- Ingredients and amounts are determined by what you want to bake
- Ingredients are *input*
- The way you combine them is the *processing*
- What is baked is the *output*

PROGRAM PLANNING TIPS

- Always have a plan before trying to write a program
- The more complicated the problem, the more complex the plan must be
- Planning and testing before coding saves time coding



Copyright 2003 Randy Glasbergen. www.glasbergen.com

PROGRAM DEVELOPMENT CYCLE

1. *Analyze*: Define the problem.
2. *Design*: Plan the solution to the problem.
3. *Choose the interface*: Select the objects (text boxes, buttons, etc.).
4. *Code*: Translate the algorithm into a programming language.
5. *Test and debug*: Locate and remove any errors in the program.
6. *Complete the documentation*: Organize all the materials that describe the program.

- Why is documentation important?

1.4 PROGRAMMING TOOLS

- Flowcharts
- Pseudocode
- Hierarchy Chart
- Direction of Numbered NYC Streets Algorithm
- Class Average Algorithm

PROGRAMMING TOOLS

- Three tools are used to convert algorithms into computer programs:
 - **Flowchart** - Graphically depicts the logical steps to carry out a task and shows how the steps relate to each other.
 - **Pseudocode** - Uses English-like phrases with some Visual Basic terms to outline the program.
 - **Hierarchy chart** - Shows how the different parts of a program relate to each other.

PROBLEM SOLVING EXAMPLE

- How many stamps do you use when mailing a letter?
- One rule of thumb is to use one stamp for every five sheets of paper or fraction thereof.

1. INPUT: Request the number of sheets of paper;
call it Sheets
2. PROCESSING: Divide Sheets by 5
3. PROCESSING: Round the quotient up to the next
highest whole number; call it Stamps
4. OUTPUT: Reply with the number Stamps

FLOWCHARTS





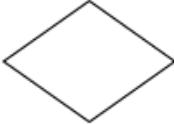
- Graphically depict the logical steps to carry out a task and show how the steps relate to each other.



"It may be a model, Captain, but it's highly illogical."

www.FieldstoneAlliance.org

FLOWCHART SYMBOLS

Symbol	Name	Meaning
	<i>Flowline</i>	Used to connect symbols and indicate the flow of logic.
	<i>Terminal</i>	Used to represent the beginning (Start) or the end (End) of a task.
	<i>Input/Output</i>	Used for input and output operations, such as reading and displaying. The data to be read or displayed are described inside.
	<i>Processing</i>	Used for arithmetic and data-manipulation operations. The instructions are listed inside the symbol.
	<i>Decision</i>	Used for any logic or comparison operations. Unlike the input/output and processing symbols, which have one entry and one exit flowline, the decision symbol has one entry and two exit paths. The path chosen depends on whether the answer to a question is "yes" or "no."

FLOWCHART SYMBOLS CONTINUED



Connector

Used to join different flowlines.



Offpage Connector

Used to indicate that the flowchart continues to a second page.



Predefined Process

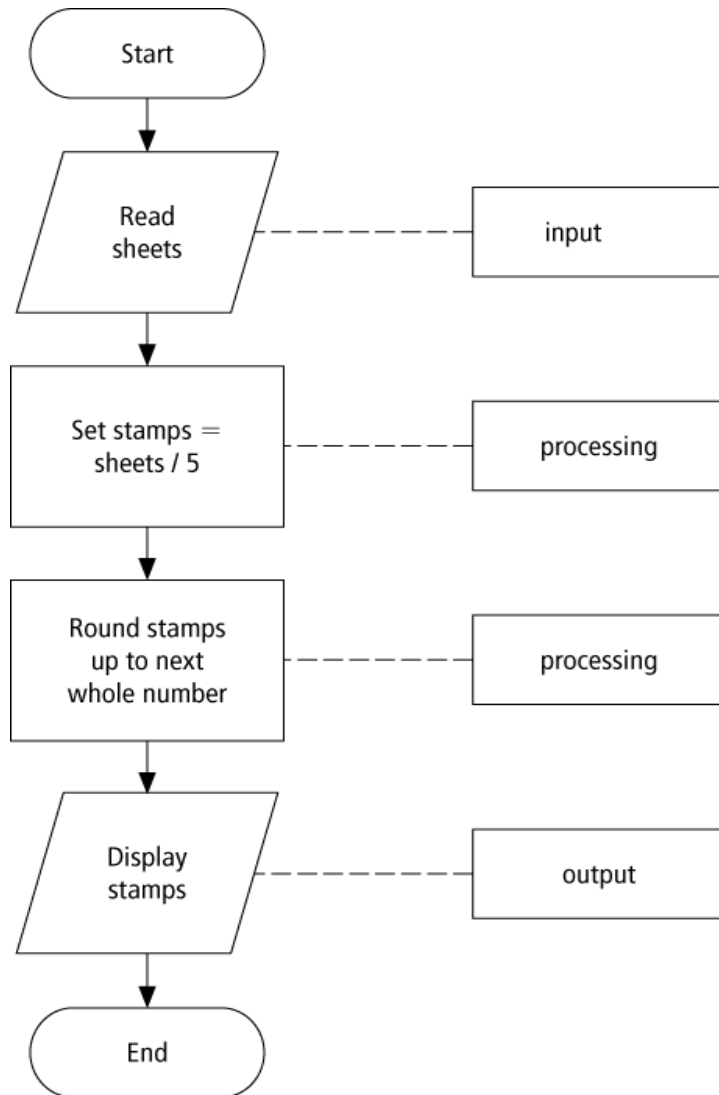
Used to represent a group of statements that perform one processing task.



Annotation

Used to provide additional information about another flowchart symbol.

FLOWCHART EXAMPLE



- Uses English-like phrases with some Visual Basic terms to outline the task.

PSEUDOCODE EXAMPLE

- Determine the proper number of stamps for a letter
 - Read Sheets (*input*)
 - Set the number of stamps to $\text{Sheets} / 5$ (*processing*)
 - Round the number of stamps up to the next whole number (*processing*)
 - Display the number of stamps (*output*)

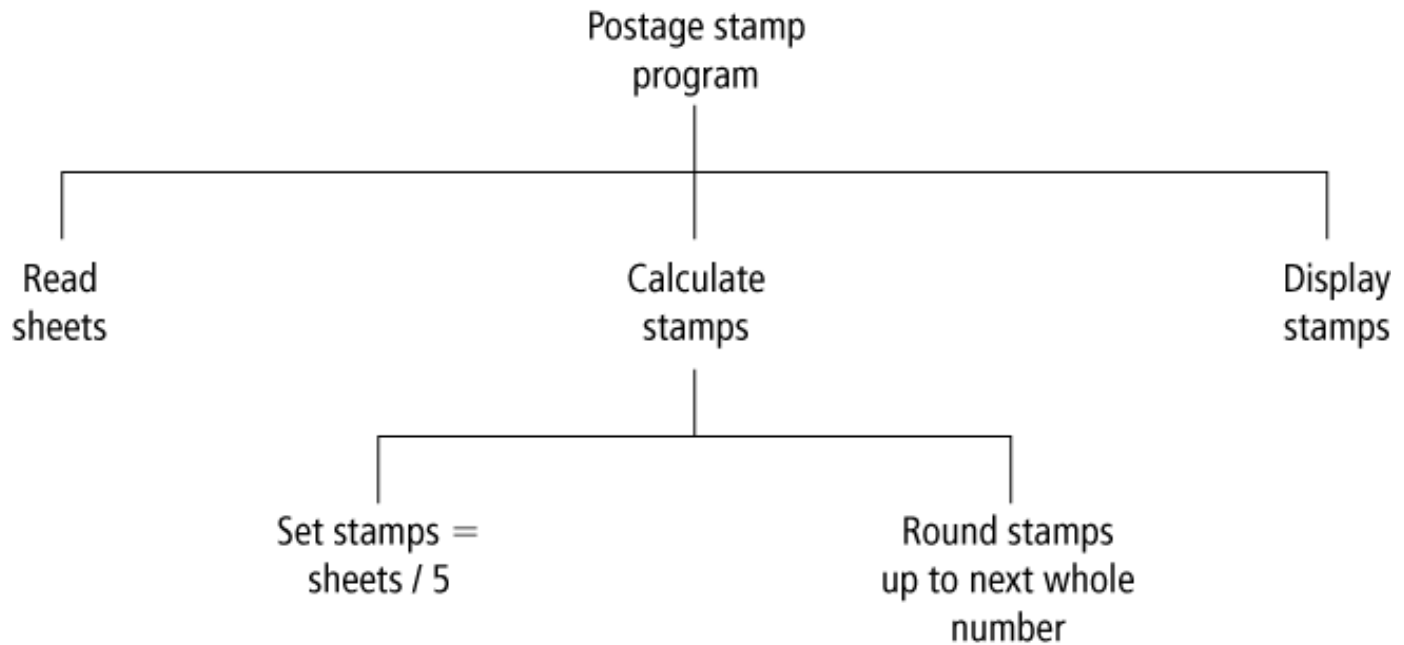
HIERARCHY CHARTS

- Show how the different parts of a program relate to each other

Hierarchy charts may also be called

- structure charts
- HIPO (Hierarchy plus Input-Process-Output) charts
- top-down charts
- VTOC (Visual Table of Contents) charts

HIERARCHY CHARTS EXAMPLE



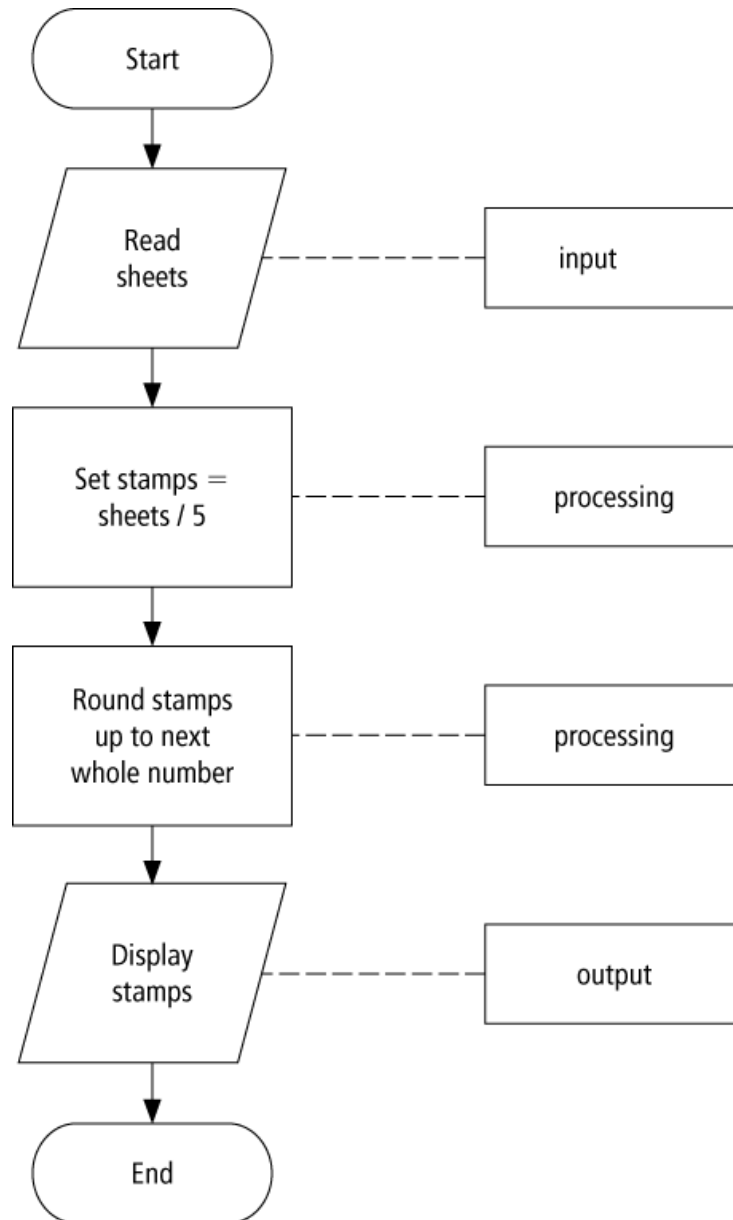
DIVIDE-AND-CONQUER METHOD

- Used in problem solving – take a large problem and break it into smaller problems solving the small ones first
- Breaks a problem down into modules

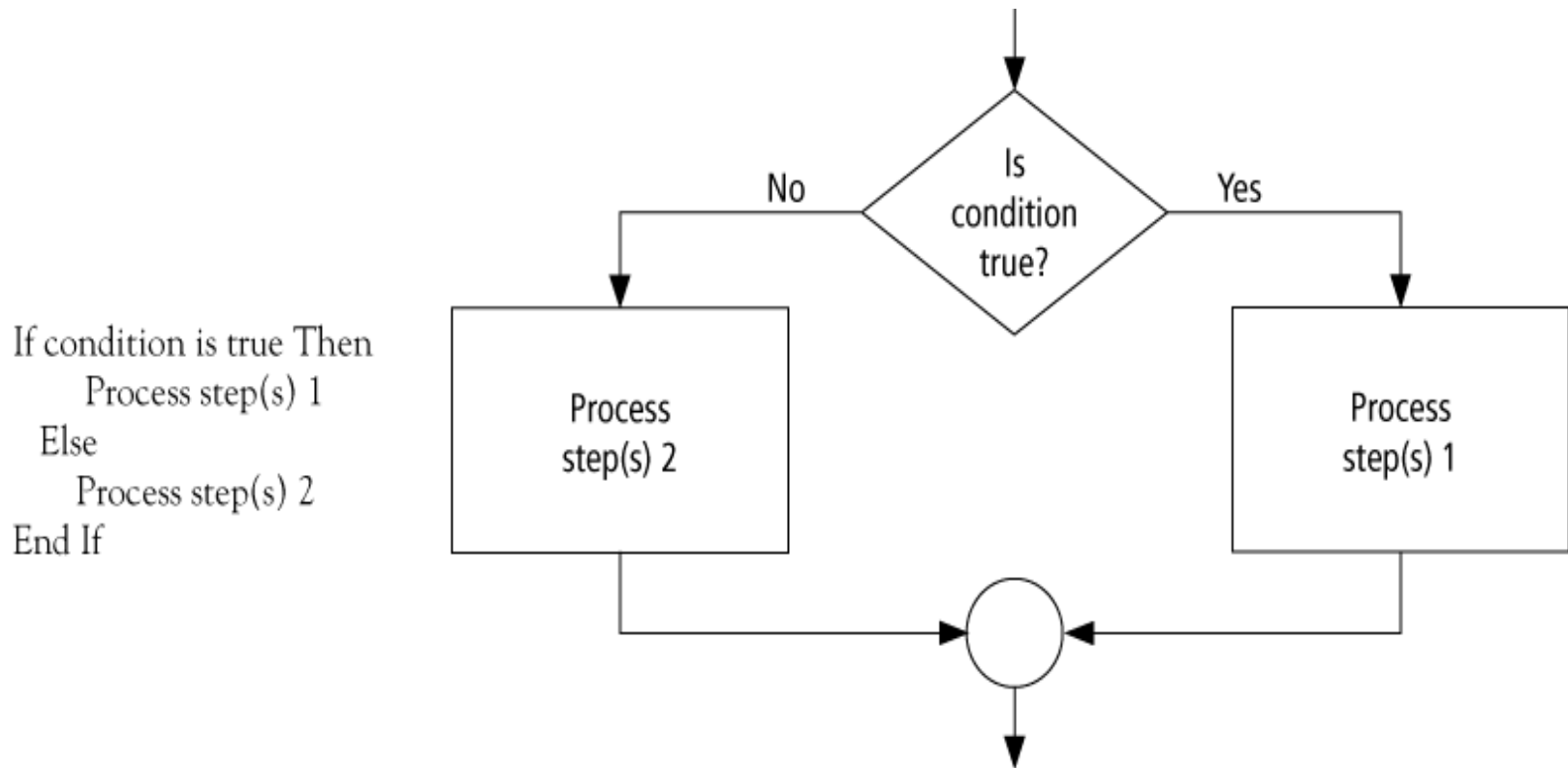
STATEMENT STRUCTURES

- Sequence – follow instructions from one line to the next without skipping over any lines
- Decision - if the answer to a question is “Yes” then one group of instructions is executed. If the answer is “No,” then another is executed
- Looping – a series of instructions are executed over and over

SEQUENCE FLOW CHART

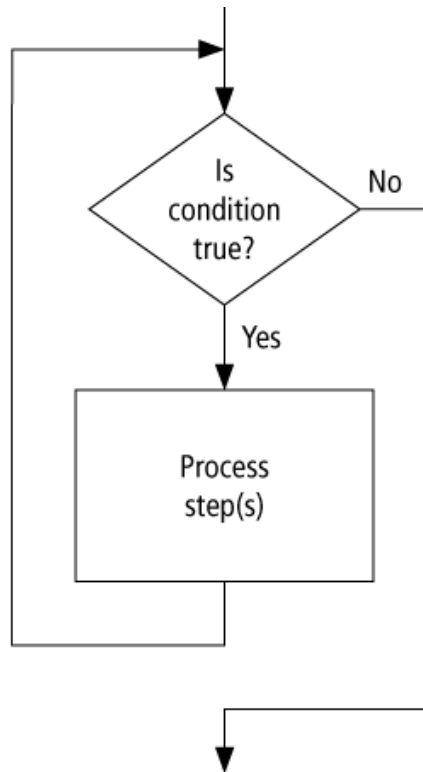


DECISION FLOW CHART



LOOPING FLOW CHART

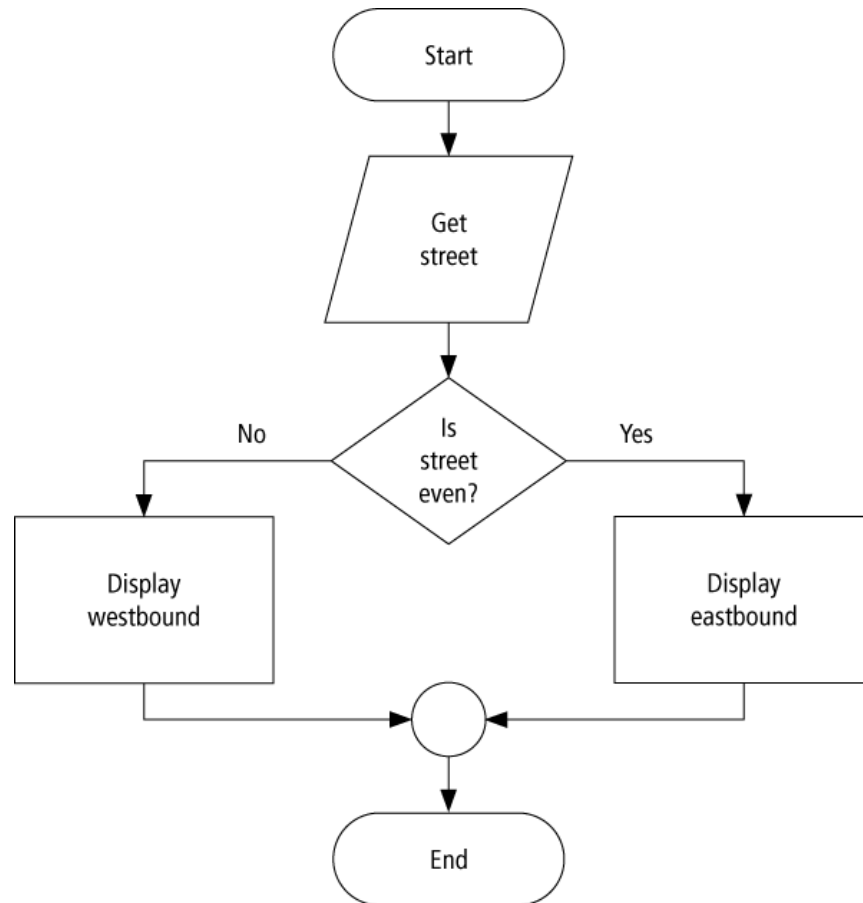
Do While condition is true
Process step(s)
Loop



DIRECTION OF NUMBERED NYC STREETS ALGORITHM

- **Problem:** Given a street number of a one-way street in New York City, decide the direction of the street, either eastbound or westbound
- **Discussion:** in New York City even numbered streets are Eastbound, odd numbered streets are Westbound

FLOWCHART



Determine the direction of a numbered NYC street

Get street

If street is even Then

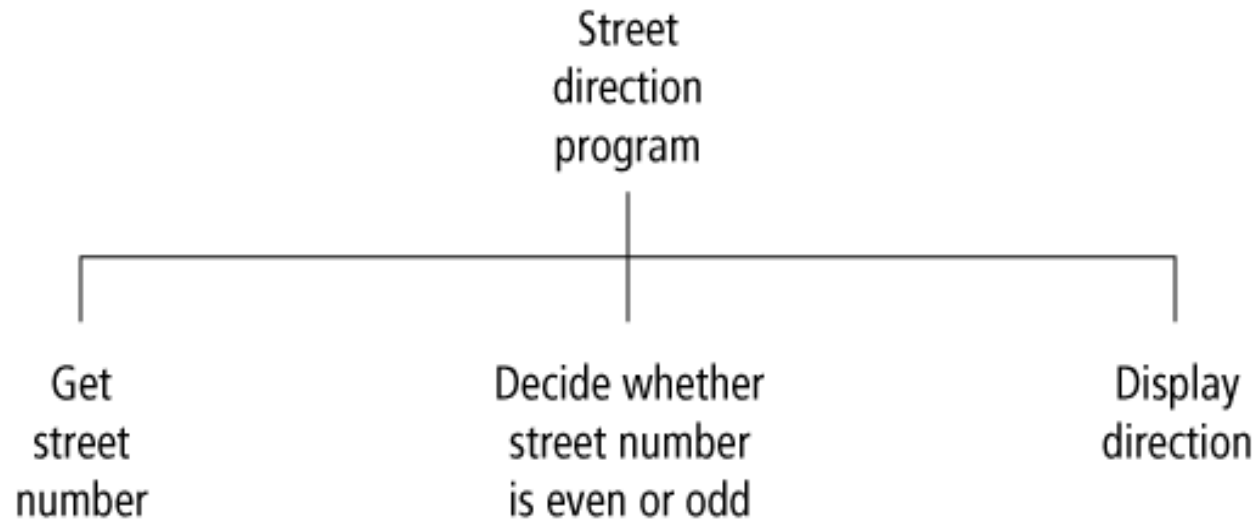
 Display Eastbound

Else

 Display Westbound

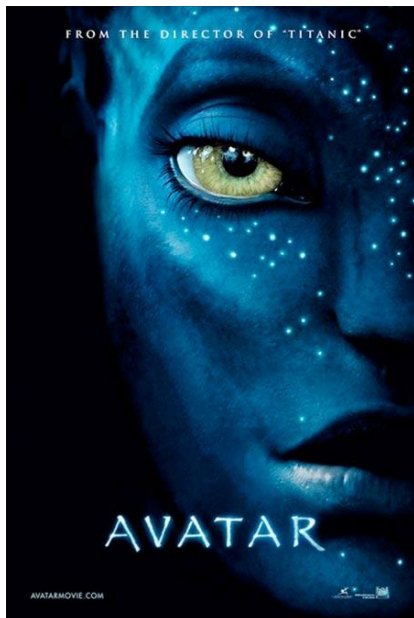
End If

HIERARCHY CHART



MOVIE RATING EXAMPLE

- Kids want to watch the movie, “Avatar”.
- Decide whether he or she can watch the movie based on his or her age.



Solution?

CLASS AVERAGE ALGORITHM

- **Problem:** Calculate and report the grade-point average for a class
- **Discussion:** The average grade equals the sum of all grades divided by the number of students

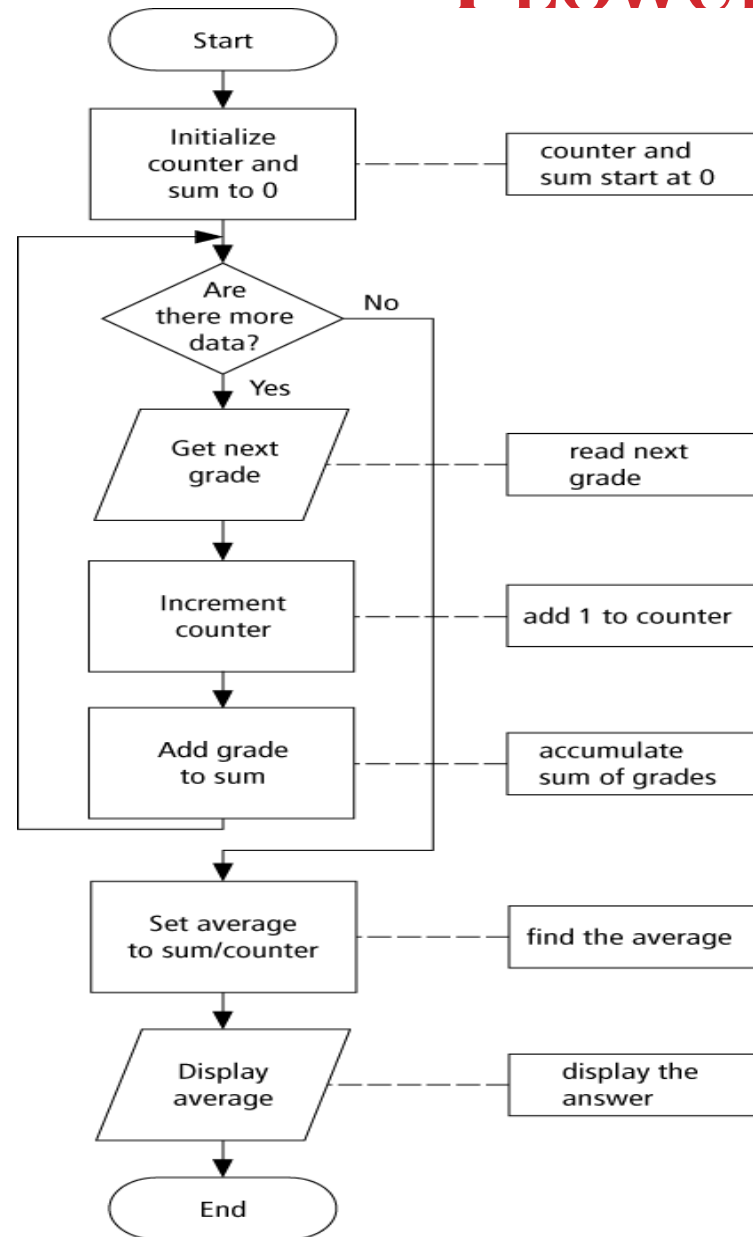
Output: Average grade

Input: Student grades

Processing: Find the sum of the grades; count the number of students; calculate average

FLOWCHART

- We need a loop to read and then add the grades for each student in the class
- Inside the loop, we also need to count the number of students in the class
- $\text{grade} = \text{sum of grades} / \text{number of students}$



Program: Determine the average grade of a class

Initialize Counter and Sum to 0

Do While there are more data

 Get the next Grade

 Add the Grade to the Sum

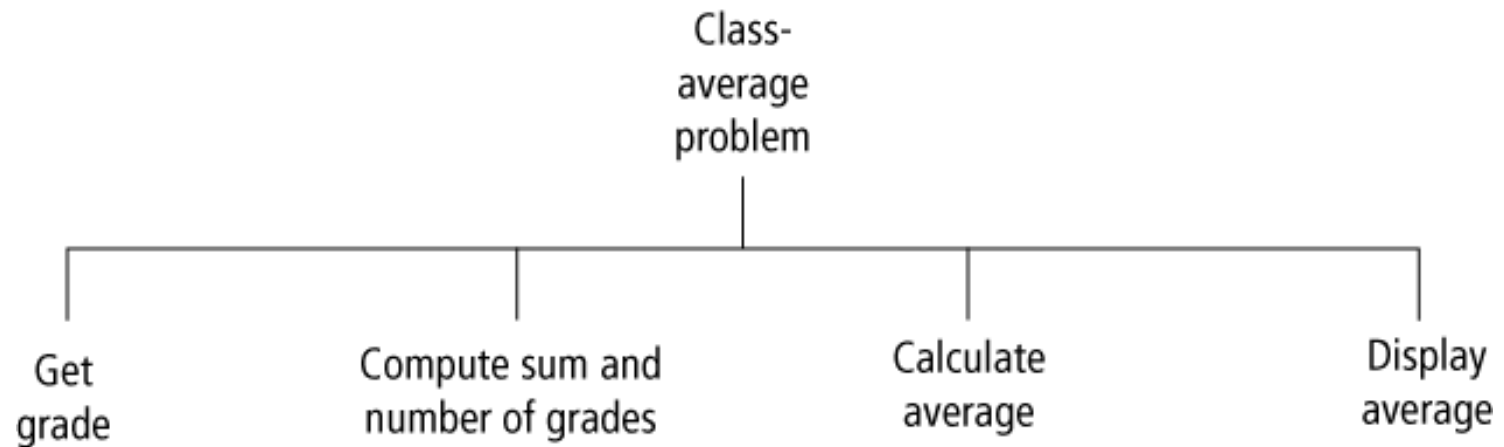
 Increment the Counter

Loop

Compute Average = Sum / Counter

Display Average

HIERARCHY CHART



- When tracing a flowchart, start at the start symbol and follow the flow lines to the end symbol
- Testing an algorithm at the flowchart stage is known as desk checking
- Flowcharts, pseudocode, and hierarchy charts are program planning tools that are not dependent on the programming language being used

- There are four primary logical programming constructs
 - sequence
 - decision
 - loop
 - unconditional branch
 - Appear in some languages as GOTO statements
 - Involves jumping from one place in a program to another
 - Structured programming uses the sequence, decision, and loop constructs but forbids the unconditional branch

TIPS AND TRICKS OF FLOWCHARTS

- Flowcharts are time-consuming to write and difficult to update
- For this reason, professional programmers are more likely to favor pseudocode and hierarchy charts
- Because flowcharts so clearly illustrate the logical flow of programming techniques, they are a valuable tool in the education of programmers

TIPS AND TRICKS OF PSEUDOCODE

- There are many styles of pseudocode
- Some programmers use an outline form
- Some use a form that looks almost like a programming language
- The pseudocode in the case studies of this text focus on the primary tasks to be performed by the program and leaves many of the routine details to be completed during the coding process

TIPS AND TRICKS OF HIERARCHY CHARTS

- Many people draw rectangles around each item in a hierarchy chart
- In the text, rectangles are omitted to encourage the use of hierarchy charts by making them easier to draw

FOR NEXT WEEK

- Read Chapter 2 & Appendix D
- Install Visual Studio 2010

PROGRAMMING IS USEFUL!

