Programming Strand
IT and Me Works
Strand Outline

Strand Description:

The programming strand in IT and Me Works will provide students with an opportunity to gain an understanding of structured and object-oriented programming. Students will develop problem solving techniques and critical thinking skills through the application of programming concepts and will gain an overview of programming languages and their uses in today's computerized society.

Note to teachers: No knowledge of programming languages is required to teach this strand. Although you can incorporate programming code in the strand to demonstrate theory concepts it is not required. The lesson plans provided with this module have been developed such that they can be utilized without programming in a specific language. However, instructors who have programming experience can easily incorporate code examples into the curriculum.

Strand Objectives:

- Gain an understanding of programming history and current trends, languages and uses
- Understand how computer programming languages work
- Gain knowledge of structured and object-oriented programming languages and constructs
- Develop a simple program using common programming concepts
- Demonstrate problem solving ability and critical thinking skills
- Effectively communicate program function and flow
- Understand different programming career pathways, necessary skills and educational requirements

Suggested Text(s):

Appleman, Daniel. How Computer Programming Works. APress. 2000.
ISBN 10893115-23-2
http://www.apress.com

Teaching Materials:

Computer classroom.

Student Materials:

None

Time:

This strand has been designed as a 34-70 hour time frame. Minimum hours have been suggested for each topic based on 34 hours. The content can expanded to fit 70 hours or more based on individual school needs.

Implementation Costs:

No use of a programming language is required to teach this strand. However, instructors who have programming experience can easily incorporate coding into the curriculum. With the availability of free development environments and compilers there should be no implementation costs associated with this module if coding is included. However, costs may be incurred if instructors choose to teach a language that requires a particular software package. Choice of language(s) may also influence hardware configuration needs. Although most programming languages can be written using simple text editors and compiled using free or low cost applications some languages require systems with a minimum configuration of memory and disk space. It is highly recommended that instructors assess their classroom hardware and software configuration and make appropriate language decisions based on resources and system capability.

(See *Resources* for suggestions on languages, applications and compilers).

Content Outline:

- Introduction to Programming
  - The whats, hows and whys of programming:
    - what is programming, how is it used and what can it be used for?
  - History of programming
- Problem Solving and Critical Thinking Skills
  - logic problems
  - sequential thinking exercises
- Types of Programming
  - Structured programming
    - Suggested languages include QBasic, Basic, C, C++, VisualBasic
  - Object-Oriented programming
    - Suggested languages include Java, SmallTalk, C#
  - Web programming and markup languages
    - Suggested languages include HTML, XML
- Program Development
  - Designing a program
    - Program Flow
    - Flow Charting
    - Pseudocode
  - Writing programs
    - Statements
    - Constants
    - Variables
    - Operators
    - Input and output
    - Sequence
    - Boolean
    - Decision
    - Looping
  - Compiling
  - Testing and Debugging
  - User Documentation
- Final Project – "IT and Me Works in My Life"

Suggested Learning Activities

Introduction to Programming

Suggested Time:	2 hours	4 hours
Time Frame:	34 hours	70 hours

To introduce programming, how it is used, and why it is important divide the class into teams and have each group research an industry that uses programming. Suggested industries include: web development, gaming, marketing, manufacturing, banking, ecommerce, publishing, entertainment (TV/Movies), and computer animation. *Part A of the Final Project.*

Have each team identify how programming is used as part of the business and what programming languages are primarily used in the industry. Students should also investigate what skills are needed to work in the industry (programming skills and also general knowledge skills such as math, biology, etc...) and what types of jobs are available in each industry. Apart from library and/or internet research the group may need to contact local companies or industry representatives as part of the project. Each group should prepare a short presentation and share their findings with the rest of the class.

See *Lesson Plan for Introduction to Programming* for additional information and activities.

Problem Solving and Critical Thinking Skills

Suggested Time:	5 hours	10 hours
Time Frame:	34 hours	70 hours

Review steps to problem solving and give students the opportunity to practice their skills through the use of logic problems and sequential thinking exercises. Set up logical thought process for pseudocode and flow chart development through in-class exercises.

Problem-solving activities can be worked in throughout the curriculum. It is suggested that activities be provided at the start of each class to stimulate thinking and engage students in the problem solving process.

See *Resources* for references and activity links and *Lesson Plan for Problem Solving* for additional information and activities.

Types of Programming

Suggested Time:	5 hours	10 hours
Time Frame:	34 hours	70 hours

See *Lesson Plan for Types of Programming – Structured Programming*, *Lesson Plan for Types of Programming – Object-Oriented Programming*, and *Lesson Plan for Types of Programming – Web Programming* for additional information and activities.

If because of time constraints it is not possible to cover all three types of programming it is suggested that instructors focus on the *Structured Programming* lesson plan and *Part B of the Final Project*. The Peanut Butter and Jelly activity will provide students with the necessary logical skills to develop a step-by-step program.

For those instructors who are knowledgeable in programming or in a specific language this topic is a good place to introduce coding examples and other hands-on activities.

Program Development

Suggested Time:	12 hours	26 hours
Time Frame:	34 hours	70 hours

See *Lesson Plan for Program Development* for information on developing a structured tic-tac-toe application in order to understand general programming concepts.

A sample tic-tac-toe application with documented user comments will be available for download from the http://www.nheon.org web site.

For those instructors who are knowledgeable in programming or in a specific language this topic is a good place to introduce coding examples and other hands-on activities.

Final Project – "IT and Me Works in My Life"

See *Lesson Plan for Final Project –" IT and Me Works in My Life"* and *Final Project Rubric* for additional information.

Resources:

History of Programming

The History of Computer Programming Languages
http://www.princeton.edu/~ferguson/adw/programming_languages.shtml
http://perso.wanadoo.fr/levenez/lang/
http://www.cs.iastate.edu/~leavens/ComS541Fall97/hw-pages/history/

Types of Programming

Structured Programming
http://www.robelle.com/library/smugbook/structpr.html
http://acweb.colum.edu/users/rcourington/Progclass/struprog.html

Object-Oriented Programming
http://catalog.com/softinfo/objects.html

Web programming and markup languages

W3C HTML Homepage	http://www.w3.org/MarkUp/
W3C Getting Started with HTML Tutorial	http://www.w3.org/MarkUp/Guide/
W3Schools.com HTML Tutorial	http://www.w3schools.com/html/
HTML Goodies Tutorial	http://www.htmlgoodies.com/tutors/
Barebones Downloadable Reference Guide	http://werbach.com/barebones/

Problem Solving and Critical Thinking Skills

The Little Giant Encyclopedia of IQ Tests. Philip J. Carter and Kenneth A. Russell Sterling Publishing Company. 2000. ISBN 0-8069-2889-1

The Little Giant Encyclopedia of Logic Puzzles. Norman D. Willis. Sterling Publishing Company. 2000. ISBN 0-8069-2689-9

Logic Problems

The Logic Problems Page
http://www.geocities.com/Heartland/Plains/4484/logic.html
Bob's Problem Page
http://www.geocities.com/CapeCanaveral/Launchpad/5952/problems.html
CRPuzzles
http://crpuzzles.com/logic/
Ed's Puzzles, Brain Teasers and Logic Problems
http://www.inficad.com/~ecollins/logic-prob.htm
Symbolic Logic
http://home.earthlink.net/~lfdean/carroll/puzzles/logic.html
Logic Problems
http://www.bogiegraphics.com/logicproblems/home.htm
Puzzlers Paradise
http://www.puzzlersparadise.com/article1021.html

Sequential Thinking Exercises

- Peanut Butter and Jelly Programming
  http://www.teachers.net/lessons/posts/2166.html

Program Development

Programmable Legos – ROBOLAB, LEGO Mindstorms
http://www.pldstore.com

Programming for Poets
http://hobbes.ncsa.uiuc.edu/poets/outline/

QBasic (QuickBasic)

Many free compilers, tutorials and programs available online
Will run on all versions of MS-DOS and Windows 95, 95, NT, 2000

- http://www.qbasic.com
- http://qb4all.com
- http://qbpathfinder.hypermart.net/trs/
- http://users.pandora.be/nicvroom/implement.htm
- http://geocities.com/Area51/5967/qbasic.html

Basic

Programming lessons in Basic:

- Magic Number Guessing Program http://web.ai/club/basic01.html
- Create a Story Program http://web.ai/club/basic02.html
- Guess an Animal Program http://web.ai/club/basic03.html