A Model Curriculum for K-12 Computer Science*

Level 2 Objectives and Outlines

(Draft 02/2005)

Anita Verno
Bergen Community College
400 Paramus Road
Paramus, NJ 07652
201-447-7909
averno@bergen.edu

Debbie Carter
Lancaster County Day School
725 Hamilton Road
Lancaster, PA 17603
717-392-2916
carterd@e-lcds.org

Robb Cutler
The Harker School
500 Saratoga Avenue
San Jose, CA 95129
robbc@harker.org

Michelle Hutton
The Girls' Middle School
423 Dell Avenue
Mountain View, CA 94043
mfh@pobox.com

Lenny Pitt
University of Illinois at Urbana-Champaign
201 North Goodwin Ave
Urbana, IL 61801
217-244-6027
pitt@uiuc.edu

*Tucker, Allen, (editor), Deek, F., Jones, J., McCowan, D., Stephenson, C., and Verno, A. *A Model Curriculum for K-12 Computer Science: Final Report of the ACM K-12 Task Force Curriculum Committee.* Association for Computing Machinery (ACM), New York, New York, October, 2003.
http://www.acm.org/education/k12/k12final1022.pdf

SIGCSE'05, February 23–27, 2005, St. Louis, Missouri, USA.
ACM 1-58113-997-7/05/0002.

Overview:

The ACM K-12 Computer Science Model Curriculum was developed in 2003 to provide a broad outline from which a K-12 computer science curriculum can be developed. The Model Curriculum was a response to the view that computer science education is not clearly defined or well-established at the K-12 level. A national computer science curriculum which stabilizes the objectives and content of high school CS has implications beyond K-12 education. It will assist students with further study in computer science / information technology / information systems / engineering and will help with the larger national effort to build America's position as a global leader in technological knowledge and expertise.

This document continues efforts by the ACM and Computer Science Teachers Association[*] (CSTA) to create a comprehensive body of resources to support the implementation of a national K-12 computer science curriculum. The Model curriculum provides an overview of computer science content broken into four levels: Level 1, recommended for students in grades K-8; Level 2, recommended for students in grades 9 or 10; and Levels 3 and 4, recommended for students in upper grades. The Level 1 curriculum closely follows ISTE standards and is therefore a fairly well-developed set of learning objectives with available resource materials. The Level 2 curriculum, suggested for all students, was broadly described in the Model. This document specifies the learning objectives, assessment measures, and sample educational activities of Level 2, which can be used to encourage a uniform level of teaching across the US. Once the Level 2 Objectives and Outlines are finalized, lesson plans and activities will be developed to support each stated objective, and these will be classified and placed into the CSTA Web Repository[†] (currently under development).

The authors strongly believe that the Level 2 curriculum must reflect standards that are both educationally appropriate for all students, and accessible to all students regardless of economic pressures within individual educational systems. Your thoughts and opinions will help inform the Level 2 standards as well as future initiatives that will keep the ball rolling in the direction of national standards for computer science education in the K-12 community.

Please categorize your comments as follows:

- General document content
- Student learning objectives
- Assessment recommendations
- Resource availability – Each topic lists the resources the Authors believe are required to successfully meet the objectives. Please share your views regarding availability of stated hardware, software, and other technology, as well as textbooks, workbooks, and web access, as necessary to complete the sample activities.

[*] The Computer Science Teachers Association is a membership organization that supports and promotes the teaching of computer science and other computing disciplines by providing opportunities for K-12 teachers and students to better understand the computing disciplines and to more successfully prepare themselves to teach and to learn.

[†] The goal of the Web Repository is to create a web-based multi-grade repository of appropriate materials for K-12 computer science teachers to support their teaching and professional development. Teachers will be encouraged to submit their own original materials, to the benefit of the larger community.

- Ranking of topics according to level of importance – The numbering system currently shown within the document is for ease of discussion only. It does not reflect the Authors' view of the importance of any given topic.
- Time needed for topic – How many contact hours are needed to fully deliver each specified topic including one or more activities per focus area?
- Teacher preparedness – What type of professional development is necessary to ensure responsible delivery of the specified topics? How can this professional development best be delivered?
- Other comments

Please send all comments to cstacurriculum@csta.acm.org. Include your contact information (name, school with address, phone number if available, grade level and courses currently teaching) at the top of the email.

Thank you for taking the time to provide feedback that can be used to direct future projects towards improving CS education.

About the Level 2 Objectives and Outlines:

The document contains 14 main topics. For each main topic there is a general description, statement of necessary resources, learning objectives, assessment guidelines, list of focus areas that fall within the main topic area and a sample activity that suggests the anticipated level of student learning. The final document will cross-reference the Level 2 topics into the CSTA Web Repository, where lesson plans and additional activities will be available.

Note: For each topic, content and activities may be modified, based upon time constraints and technology available.

Topic 1: Principles of Computer Organization

Topic Description:

Principles of Computer Organization will introduce the student to the major components of the computer including: input, output, memory, storage, processing, software, and the operating system.

Textbooks and Supplies:

A computer that can be opened up and taken apart; different types of computers and operating systems recommended. Printed advertising for computers, or a computer with Internet access for web research.

Time to Complete:

Student Learning Objectives	Assessment Measures
The student will be able to:	
1. Identify the various functional components of a computer.	Lab activity Written activity
2. Match a list of computer terms and definitions/functions.	Written activity
3. Describe the interaction of the various functional components of the computer.	Lab activity Written activity
4. Make appropriate decisions when purchasing a computer for home use.	Written activity

Assessment Recommendations: An average of 60% from combined assessment measures is required to demonstrate proficiency in course material.	
Lab activities	50%
Written activities, including tests, quizzes, and written assignments	50%

Detailed Outline	
Focus	**Sample Lab / Hands-on Activity**
1. Terminology	Identify and define the key terms associated with the computer and its components.
2. Identifying hardware components	Where possible, provide each student with the opportunity to take apart an old computer and locate and identify the various components.
3. Identifying software components	Discuss the role of system software and application software.
4. Describing the interaction of components	Students perform a role play where each student actor represents one component – hardware or software. Scripts describe their general actions; the teacher provides the data for the interactions.
5. Purchasing a computer	Students locate computer advertisements in print or online. A comparative table is created that lists the advantages and disadvantages of at least three advertised computers for possible personal use.

Topic 2: Problem Solving

Topic Description:

This topic covers the basic steps in algorithmic problem-solving, including the problem statement and exploration, examination of sample instances, design, program coding, testing, and verification. Tools for expressing design such as flowcharts, pseudocode, UML, and N-S charts will be introduced.

Textbooks and Supplies:

A programming language; interactive development environment recommended.

Time to Complete:

Student Learning Objectives	Assessment Measures
The student will be able to:	
1. Name and explain the steps in the problem-solving process.	Written activity
2. Solve a problem by applying the problem-solving process.	Written activity
3. Express a solution using standard design tools.	Written activity Lab activity
4. Determine if a given algorithm successfully solves a stated problem.	Written activity

Assessment Recommendations: An average of 60% from combined assessment measures is required to demonstrate proficiency in course material.	
Written activities, including tests, quizzes, and written assignments	80%
Lab activities	20%

Detailed Outline	
Focus	**Sample Lab / Hands-on Activity**
1. Problem-solving process	Students are given simple problems to solve. They write the process they used to solve each problem, defining a general problem-solving strategy.
2. Understanding the problem (analysis)	Students will be given several problems to review. Some will have all the detail stated within the text. Others will be missing detail. For each complete problem, students will restate the problem, including input, output, formulas, and other relationships between the input and the output. For each incomplete problem, students will request an interview with the problem owner to ascertain the complete problem and reformulate it into a complete statement.
3. Exploring problems: problem representation, problem-solving heuristics and strategies.	Students employ the use of various forms of representation (e.g., patterns, grids, graphs, sets) to understand problems. Solutions are explored that can make the problem simpler by considering special cases, smaller cases, solving sub-problems, or searching for patterns.

Detailed Outline	
Focus	**Sample Lab / Hands-on Activity**
4. Problem design	Students employ a design methodology (e.g., top-down, bottom-up, combination) to a restated problem and create an algorithm. The algorithm is expressed using a standard tool.
5. Problem data	Students evaluate a problem and determine an appropriate set of data that can be used to produce correct results.
6. Solution accuracy	Students perform a structured walk-through of their algorithm to ensure it solves the problem and make corrections as needed. Students trade solutions and perform a structured walk-through of each other's algorithm.
7. Program coding and testing	Where possible, students code the problem to determine if it works as anticipated.
8. Re-evaluation and refinement	Students identify errors and propose fixes via re-application of the problem solving process.
9. Communicate results	Students document their solution and share with other classmates.

Topic 3: Basic Components of Computer Networks

Topic Description:

Basic knowledge of networking will be introduced as a building block for understanding how computers communicate. The student will become familiar with the basic components of a computer network including servers, file protection, queues, routing protocols for connection/communication, spoolers, shared resources, and fault-tolerance. Computer security and protocols will be discussed.

Textbooks and Supplies:

To be determined.

Time to Complete:

Student Learning Objectives	Assessment Measures
The student will be able to:	
1. Match a list of networking terms with their definitions.	Written activity
2. State the hardware requirements for adding a computer to a network.	Written activity
3. Connect a computer to a network.	Lab activity
4. State at least three security issues that can affect a computer that is connected to a network.	Written activity
5. Describe at least two network protocols and state the reason to select one over another.	Written activity

Assessment Recommendations: An average of 60% from combined assessment measures is required to demonstrate proficiency in course material.	
Lab activities	70%
Written activities, including tests, quizzes, and written assignments	30%

Detailed Outline	
Focus	**Sample Lab / Hands-on Activity**
1. Terminology	Students define a network and various networking terms.
2. Data transfer	Each student acts as a computer on a network using a designated network configuration. One student sends an object (data) across the network to another. Discuss how the second student knows the data belongs to him/her. Have the first student pass 20 heavy books across the network. Can they all be passed at once? Explain the concept of packets.

Detailed Outline	
Focus	**Sample Lab / Hands-on Activity**
3. Data collision and network failure	Continuing the enactment from above, have a second student attempt to pass data at the same time as the original data is passing his/her connection, causing a collision. How can this be avoided? One student hurt his/her arm and can no longer pass the object. Discuss the possible problems and protections in modern networks. Students try several configurations, noting the effect on collision, fault tolerance, number of supported connections, and number of links from origin to source (points of insecurity).
4. Connecting a computer to a network	Demonstrate the process of installing a NIC card or a modem card and physically connecting a computer to the network.
5. Identity	Demonstrate, if possible, that every computer on a network has a unique address. Compare the address to a home phone number or house address. Discuss hierarchical addressing using examples like countryCode.areaCode.exchange.suffix, as compared to IP addresses and dynamic IP addresses compared to local extension within phone network.
6. Applications of networks	Discuss and list networks with which students have experience (e.g., phone, Internet, school LAN). Discuss the number of connections and the different purposes of each. Include specific applications of a network such as video conferencing and phone conferencing.
7. Mobile computing	Discuss wireless computing and specific applications that use wireless computing (e.g., PDA, handheld, laptop). Introduce mobile computing concepts and, as available, demonstrate the use of mobile devices.
8. Network Security	Lead students to a discussion of protecting privacy and security of passwords, etc. Who is responsible for maintaining important data?

Topic 4: Internet Concepts

Topic Description:

Internet concepts will allow students to consider how Internet elements are organized, engage in effective searching, and use e-mail productively.

Textbooks and Supplies:

Internet access; software to demonstrate Internet elements

Time to Complete:

Student Learning Objectives	Assessment Measures
The student will be able to:	
1. List at least three strengths and weaknesses of each of three Internet elements and at least one use for each.	Lab activity Written activity
2. Use at least two Internet elements.	Lab activity
3. Execute Internet searches yielding requested data.	Lab activity
4. Develop and use a rubric to evaluate the results of web searches and reliability of information found on the web.	Lab activity

Assessment Recommendations: An average of 60% from combined assessment measures is required to demonstrate proficiency in course material.	
Lab activities	70%
Written activities, including tests, quizzes, and written assignments	30%

Detailed Outline	
Focus	**Sample Lab / Hands-on Activity**
1. Internet elements, including their history and uses	Students use a variety of Internet elements, including those used primarily for one-way communication, private and group discussion, and collaboration. Students articulate the benefits and limits of different Internet elements and identify uses for each.
2. Search engine fundamentals	Explanation of what a search engine is and how it functions. Cover similarities and differences between search engines such as advanced search commands, organization of output, whether they are influenced by hidden factors such as money. Compare identical searches in various search engines.
3. Search engines, directories, and authorities	Brainstorm web-based resources and categorize them. Compare and contrast their features. Try to find identical/similar information at a search engine, directory, and authority. Articulate strengths and weaknesses of each and when it would be appropriate to use each.

Detailed Outline	
Focus	**Sample Lab / Hands-on Activity**
4. Refining search parameters	Perform searches and use various methods to refine the search. Include advanced search features; Boolean operators; factors that affect results, such as spelling, wildcards, and quotes; use of different strategies, including keyword search. Compare results to expected results and continue to refine until expected results are met. Express desired document characteristics using logical (AND/OR/NOT) or set (union, intersection) operations.
5. Evaluating web sites	Create an evaluation tool to use in determining reliability and quality of individual web sites. Consider authorship, whether or not the author is an authority and/or has credibility. Evaluate several web pages from authorities to bogus resources such as the committee to ban dihydrogen monoxide.
6. Security on the Internet	Encryption: The teacher creates several different encryption keys. Students encrypt messages to one another using a designated key. The message is sent to another student to be decrypted, but the key is not. Discussion follows regarding protection when providing personal identification information like social security numbers and credit card numbers.

Topic 5: Hierarchy and Abstraction in Computing

Topic Description:

The notion of hierarchy in computing includes not only the specific programming language hierarchy (from logic circuits, instruction sets, up through high-level languages) and translation (compilers, interpreters, linking), but also the general principle that complex software is built in layers of increasingly sophisticated function, relying on simpler primitives from the layer below and often residing in software libraries.

Likewise, the notion of abstraction in computing is central, as it guides the development of software that has broad utility as opposed to being useful for only a specific narrowly-defined instance of a problem.

This unit makes the abstract ideas of "hierarchy and abstraction" more concrete by focusing first on real-life (non-computing) examples, and then on the specific uses of hierarchy and abstraction in computer science.

Textbooks and Supplies:

A programming language; interactive development environment recommended.

Time to Complete:

Student Learning Objectives	Assessment Measures
The student will be able to:	
1. Diagram at least three hierarchical levels of a given object/activity based on structural, functional, or procedural decomposition.	Written Activity
2. List at least three different high-level programming languages.	Written Activity
3. Given a diagram showing how source code becomes an executable application, state the activity occurring at each step and the intermediate files produced.	Written Activity
4. Measure the difference in time and amount of coding when using a higher-level language vs. assembly/machine language.	Lab Activity Written Activity

Assessment Recommendations: An average of 60% from combined assessment measures is required to demonstrate proficiency in course material.	
Written activities, including tests, quizzes, and written assignments	70%
Lab activities	30%

Detailed Outline	
Focus	**Sample Lab / Hands-on Activity**
1. Decomposing the complex	Students are asked to describe the parts of a car. Discussion results in the description of at least three layers of a car object such as JohnCar.door.front and JohnCar.door.rear; JohnCar.seat.front and JohnCar.seat.rear. Students then state the similarities of properties and actions at each level and determine why the next level is necessary. Appropriate dot notation should be used.

Detailed Outline	
Focus	**Sample Lab / Hands-on Activity**
	Using the same technique, students then describe the components of a baseball game providing at least two layers such as MyTeam.outfield.left and MyTeam.outfield.right and MyTeam.infield.first and MyTeam.infield.shortstop.
2. Comparing high-level languages	Present pieces of code in various languages (including at least one with which students are familiar) that accomplish similar tasks (e.g., loop, conditional).
3. Assembly and Machine Language	Ask students to write a program that solves a simple problem (e.g., add two numbers). Provide students with a short list of low-level instructions for an imaginary assembler (e.g., ADD, SUB, GET, and STO). Students translate the high-level program into a set of low-level instructions using this language. Students compare the number of steps in the higher-level language and the assembly language.
4. From source code to execution: interpreting, compiling, linking, etc.	For a language with which students are familiar, students role play the process that a program in a high-level language goes through en route to execution as machine code. Teacher provides source code and scripts; roles have names like “Chris the Compiler” and “Lynn the Linker.”
5. The role of circuits	Students simulate the execution of a simple machine instruction at the gate level: addition of two binary numbers. After introduction to gates, a schematic of a one or two-bit adder is laid out with masking tape. Students are bits with 0/1 signs, and move through the circuit, obeying the rules of each “gatekeeper.”
6. The power of hierarchy	Students apply top-down design techniques to write an algorithm for preparing a research paper that begins with three main parts: introduction, body, and conclusion. Each part is further refined into sub-steps. Point out the hierarchy by drawing a tree or similar graph. Can the algorithm be applied to a paper for a history class? How about in an English class? The ability to re-use the algorithm in a different context rather than begin again demonstrates the power of hierarchy.
7. Abstraction	Use models as an example of abstraction. Students identify a model and list what details are abstracted vs. shown. Students articulate why the abstracted details were not important. Students create a software model (e.g., a *PowerPoint* model of the digestive system, a *Squeak* model of a car, a *Flash* model of anything). They explain how the model is an abstraction of the original. Students give examples of multiple layers of abstraction in computers (e.g., if hardware is lowest level, what is next, what is next, etc., up to highest level of abstraction.)

Topic 6: Connections Between Mathematics and Computer Science

Topic Description:

Computer science is fundamentally connected to several elements of mathematics. This unit focuses on three foundational themes that are central to computation: binary numbers as related to digital computation and data representation, Boolean logic relating to conditional statements, and representations for problem modeling such as sets and functions.

Textbooks and Supplies:

A programming language; interactive development environment recommended.

Time to Complete:

Student Learning Objectives	Assessment Measures
The student will be able to:	
1. Convert decimal numbers to binary, and vice-versa.	Written activity
2. Create and translate an encoded message using a simple method of encryption.	Written activity
3. Write conditional statements that include simple and complex Boolean expressions to solve stated problems.	Written activity Lab activity
4. Convert a problem description into correct set notation and apply appropriate set operations.	Written activity
5. Write a function (method, etc.) that returns the correct value, given a function definition in mathematical notation.	Written activity Lab activity

Assessment Recommendations: An average of 60% from combined assessment measures is required to demonstrate proficiency in course material.	
Written activities, including tests, quizzes, and written assignments	70%
Lab activities	30%

Detailed Outline	
Focus	**Sample Lab / Hands-on Activity**
1. Binary number system	Presentation about how data and instructions are stored in binary code, including conversions from decimal to binary and binary to decimal. Students work additional problems at their seats and/or at the board, perhaps as part of a game.
2. Understanding various forms of encryption	Using a simple form of encryption, encrypt a text message and pass it to a classmate, along with a key. The receiver will use the key to translate the encrypted message back to the original text.

Detailed Outline	
Focus	**Sample Lab / Hands-on Activity**
3. Conditionals and Boolean logic	Students implement a movie rating system that assigns a rating (from 1 to 100) based on the user's answers to a series of questions (e.g., "Does it involve a car chase?" "Is there romance?" "Are there teenagers?"). If available, the movie rating system is coded. Students use cards with whimsical characters defined by several Boolean attributes (hairy/bald one/two eyes, etc.) One team chooses a hidden logical rule such as "(hairy AND one-eye) OR tall". The other team chooses a card and guesses whether it satisfies the rule, with the goal of getting the fewest incorrect guesses until the rule is inferred and no further mistakes occur.
4. Functions including parameters and mathematical notation	Students practice writing functions and methods.
5. Set representation, and set operations	Students use whimsical binary character cards to populate a three-ring Venn diagram with circles corresponding to three attributes (E.g., hairy, two-eyes, short). Challenge questions are to describe regions and combinations of regions using logical notation (e.g., "hairy OR short" for the union, "(hairy AND short) OR (hairy AND two-eyes) OR (short AND two-eyes)". Similarity of union and OR, intersection and AND, complement and NOT should be stressed.

Topic 7: Models of Intelligent Behavior

Topic Description:

Intelligent software is becoming increasingly prevalent. This unit explores seven broad areas:

- What is Intelligence?
- Natural language
- Knowledge-based Systems
- Machine Learning
- Game-playing, Problem-solving, and Search
- Robotics
- Vision and Speech

The primary goal is to demonstrate that "intelligent" machine behavior is not "magic" but is based on algorithms applied to useful representations of information. A secondary goal is for students to appreciate those characteristics that make certain tasks easy or difficult for computers, and how these differ from those that humans characteristically find easy or difficult.

Textbooks and Supplies:

To be determined.

Time to Complete:

Student Learning Objectives	Assessment Measures
The student will be able to:	
1. Given a list of tasks from several application areas of artificial intelligence, indicate whether or not computers can do those tasks, using current technology.	Written Activity

Assessment Recommendations: An average of 60% from combined assessment measures is required to demonstrate proficiency in course material.	
Written activities, including tests, quizzes, and written assignments	TBD
Lab activities	TBD

Detailed Outline ***(Select activities based on student needs and available resources.)***	
Focus	**Sample Lab / Hands-on Activity**
1. What is Intelligence?	Teacher provides web sites and material that will help students research and debate this question.
2. Natural Language	Syntax: Students play a game of "Mad Libs." Discussion: Are the resulting paragraphs syntactically correct? Can meaning be extracted? Meaning: Students examine headlines or other short passages in which the literal meanings differ from the meanings that would be understood by humans (e.g., "Kicking Baby Considered to be Healthy" and "Man Shoots Neighbor with Machete.") and then create some of their own. Discussion: Can a computer be taught to infer something that is not stated in the text, ignoring a literal

Detailed Outline *(Select activities based on student needs and available resources.)*	
Focus	**Sample Lab / Hands-on Activity**
	interpretation if necessary?
3. Knowledge-based Systems	Students explore some ways that knowledge can be represented and put to use, including rule-based deduction and tree-based classification. Students engage in a game of deduction using existing (teacher-specified) rules to deduce new facts. For example, using rules based on a student's grade level and native language, assign her to a foreign language course. Present everyday examples of expert systems such as medical diagnostic screening and car repair. Students draw decision trees to represent portions of a system, illustrating the how responses to specific questions lead to other questions and eventual results.
4. Machine Learning	Students discuss simple learning algorithms employed by online retailers where previous purchases lead to suggested purchases. Each student creates a simple learning algorithm to predict a classmate's likes/dislikes of songs or clothes based on relevant features. Discussion: programs can be written to observe patterns of behavior, resulting in modification of a program-controlled activity.
5. Game Playing, Problem Solving, and Searching	Students play game of Nim (or other game with simple guaranteed strategy) with each other and against the computer. The winning strategy is discussed. Students play games against a computer opponent (Othello, chess). How can a computer make decisions about each move? Problem Solving and Tree Search: Students try to solve the 4-queens problem (place four queens on a 4 x 4 chessboard so that none are attacking each other). Afterwards, the teacher models search and backtrack by generating a search tree of board configurations that lead to a solution. Students employ the tree-search method to solve a logical problem like the river-crossing (farmer/fox/chicken/corn). Discussion: how "intelligent" behavior can arise from a good understanding of a problem or an exhaustive or principled search.
6. Robotics	A large selection of robotics materials at all levels exists on the World Wide Web. Select appropriate materials for your students.
7. Vision and Speech	Presentation/demo, where possible, of a speech recognition and computer vision system.
8. The Myth of Intelligent Behavior	Explore how intelligent behavior suddenly loses its luster after one understands the little man behind the curtain. Points to explore: sometimes cheap tricks go a long way (Nim). Sometimes, a lot of work goes only a little way (natural language). Sometimes, a lot of work goes a long way (chess).

Topic 8: Interdisciplinary Utility of Computers and Problem Solving in the Modern World

Topic Description:

Students will gain an appreciation for the many ways (types of use) in which computers have had an impact across the range of human activity, as well as for the many different fields in which they are used. Examples should illustrate the broad, interdisciplinary utility of computers and algorithmic problem solving in the modern world.

Textbooks and Supplies:

Computer with Internet connection, magazines, newspapers.

Time to Complete:

Student Learning Objectives	Assessment Measures
The student will be able to:	
1. Find (in newspapers, magazines, or on the Internet) and describe three examples of the use of technology in non-computer fields.	Presentation Written activity
2. Choose the appropriate category for each item in a list of technology applications.	Written activity

Assessment Recommendations: An average of 60% from combined assessment measures is required to demonstrate proficiency in course material.	
Lab activities	75%
Written activities, including tests, quizzes, and written assignments	25%

Detailed Outline ***(Select activities based on student needs and available resources.)***	
Focus	**Sample Lab / Hands-on Activity**
1. How are computers used in our world?	Student list various ways they have seen computers used. The teacher assists in categorizing them based on common use (e.g., personal finances, business, entertainment, government, education, etc.) Teacher adds examples to be sure list contains many common applications. Several applications are selected for student analysis. The following types of questions are asked about the application: What is the general computational task? What is the input? What is the output? Does the task rely on an underlying body of data? What security or ethical considerations are involved? What makes this application different from the other applications under analysis?

Detailed Outline ***(Select activities based on student needs and available resources.)***	
Focus	**Sample Lab / Hands-on Activity**
2. Information storage and retrieval, including examples from the web, such as tables and searchable databases	Students participate in a scavenger hunt by visiting a variety of designated web sites to retrieve specified information. (Ideally, this activity would be integrated with another course, so that students were using the tools to solve a problem in another discipline.)
3. Decision-making support	Discuss how computers offer support for decision-making (e.g., knowledge bases, sales data analysis, expert systems.)
4. Data visualization	Presentation by teacher or guest speaker about uses of data visualization (e.g., CAT scan). If available, take a field trip to a medical facility to view data visualization and learn how this application assists the population. Visit the hospitals data center and discuss the effect of HIPPA on how each patient's computerized data is handled.
5. Communications	Demonstration and hands-on (where possible) experimenting with various types of communications (e.g., web page, web forum, e-mail, mobile phone, PDA, telemeeting, multimedia presentation).
6. Modeling and design, including CAD and simulations	Stage a simulation of a simple process or event, such as the transmission of a (biological) virus. This can be accomplished using computer software, PDAs, or by passing colored paper in sealed envelopes among the students (with one color being the "virus"). Why do computers excel at simulation? Select some real-world examples for discussion. Use modeling software to model a simple structure from another subject area (e.g., the structure of a molecule or a set design for a theater class play).
7. Art, music, and video for entertainment, including editing of media, graphics, games, and 3D animation	Develop a simple graphical game. Record and manipulate music or sounds. Visit web sites that display animated graphics. Locate on the web and then list applications that permit creation and editing of images, audio, and video. Discuss advantages and disadvantages of the various products listed.
8. Education	Students use tutorial software to learn something new.
9. E-Commerce	Discussion of activities that fall within E-commerce such as advertising, online ordering, and auctions. Create a list of items to purchase online. Students locate the best price(s), add the item(s) to their shopping cart, and save the order.
10. Embedded systems	Discussion of example uses of computer technology embedded in various devices (auto diagnostics, pacemakers, security cameras that respond to and record motion, sensors and control, appliances). Divide students into teams. Each team lists as many examples as possible of computers (chips) working in school, businesses, and homes, describing their functions, if known.

Detailed Outline ***(Select activities based on student needs and available resources.)***	
Focus	**Sample Lab / Hands-on Activity**
11. More examples of computing applications	Students find three examples of the use of technology and prepare an in-class and/or written presentation, according to the classifications listed above. Resources: newspapers, magazines, Internet news sites.

Topic 9: Ethical Issues

Topic Description:

The proliferation of computers and networks raises a number of ethical issues. Technology has had both positive and negative impacts on human culture. Students will be able to identify ethical behavior and articulate both sides of ethical topics.

Textbooks and Supplies:

Word processing and presentation software recommended.

Time to Complete:

Student Learning Objectives	Assessment Measures
The student will be able to:	
1. Distinguish between ethical and legal issues in a case study by listing the issues that can be resolved through the legal system and those issues that cannot be legally resolved.	Written activity
2. Defend an ethical stance given a controversial or ethically ambiguous situation in a debate.	Written or Oral activity
3. List and explain at least two positive and negative effects of one technological innovation on human culture.	Written activity
4. Discuss the anticipated impact of technology on society and list and defend possible ethical and social implications of such technological innovations.	Written activity
5. Define intellectual property and state the impact of provisions to protect it.	Written activity
6. Identify at least two benefits and two drawbacks of using commercial, public domain, open source, and shareware.	Written activity

Assessment Recommendations: An average of 60% from combined assessment measures is required to demonstrate proficiency in course material.	
Written activities, including tests, quizzes, and written assignments	90%
Oral activity	10%

Detailed Outline	
Focus	**Sample Lab / Hands-on Activity**
1. Terminology	The difference between legal and ethical, where they overlap and where they don't. Differences between values and ethics. Foundation in philosophical ethics. Create a list of criteria for determination of whether a decision is ethical. Create guidelines for ethical use and creation of technology.

Detailed Outline	
Focus	**Sample Lab / Hands-on Activity**
2. How technology has changed ethical and legal issues	Discuss the ways that technology has increased access to personal and classified information, facilitated copying, and blurred jurisdictional boundaries. Include security issues, such as the effects of the Patriot Act. Examine case studies.
3. The effect of technology on human culture, including historical considerations	Choose one technological innovation from a provided list and write a paper explaining at least two positive and negative effects on users and others impacted by the innovation. Writing assignment to predict how technology may affect human culture in the future.
4. Privacy and sharing of personal information	Ask students if they have secrets that they do not want to share. How would they feel if someone were able to obtain that secret by snooping in their locked secure journal? How would they feel if the person then shares this private knowledge with the rest of the class? What can they do to prevent someone from obtaining their personal information, e.g., hide the key? Discuss possible hiding places and why some are better than others. Lead students to a discussion of protecting privacy on the Internet and security of passwords, etc.
5. Intellectual property, copyright, and fair use	Create a guide for other students outlining copyright and fair use within the school context.
6. Responsible use of software	Compare and contrast the responsibilities of using commercial, public domain, shareware, and open source software. What are the benefits and limitations of each?

Topic 10: Careers in Computing

Topic Description:

Careers in Computing will introduce the student to a range of jobs and careers available in the field of information technology and computing.

Textbooks and Supplies:

Internet access; Employment section of newspaper.

Time to Complete:

Student Learning Objectives	Assessment Measures
The student will be able to:	
1. Create a chart of at least five careers available in the field of information technology and computing, including the career name, educational level necessary, experience requirements, job description, and salary range.	Written activity
2. Use an online job search site to determine if there is an opening for a specified job title.	Lab activity

Assessment Recommendations: An average of 60% from combined assessment measures is required to demonstrate proficiency in course material.	
Written activities, including tests, quizzes, and written assignments	60%
Lab activities	40%

Detailed Outline	
Focus	**Sample Lab / Hands-on Activity**
1. Job/Career Titles and Descriptions	Students use the Internet to research jobs available on job listing sites. Titles, descriptions, requirements, and, salary ranges are to be charted.
2. Guest Lecturers	Guest lecturers, from various technology-related careers, can be invited to talk with students about their jobs. Students have the opportunity to ask questions and solicit information.
3. Current Events	Students bring in articles and lead discussions on current events in the technology-related career sector.
4. Job Availability	Students use web sites and the library to research labor statistics (local, national, and international) for a group of assigned computer careers.
5. The Perfect Job	Students write up a description of their ideal technology-related job/career.

Topic 11: Programming Languages

Topic Description:

Programming Languages will introduce the student to some basic issues associated with program design and development. The focus of this unit is to establish an appreciation of the work being done by software.

Textbooks and Supplies:

A programming language; interactive development environment recommended.

Time to Complete:

Student Learning Objectives	Assessment Measures
The student will be able to:	
1. Code, test, and execute a program.	Lab activity
2. Convert a word problem into code using top-down design.	Written activity Lab activity
3. Select appropriate data types.	Written activity Lab activity
4. Write structured program code.	Lab activity
5. Describe the changes occurring in RAM as code executes.	Written activity

Assessment Recommendations: An average of 60% from combined assessment measures is required to demonstrate proficiency in course material.	
Lab activities	50%
Written activities, including tests, quizzes, and written assignments	50%

Detailed Outline	
Focus	**Sample Lab / Hands-on Activity**
1. Terminology	Identify and define key terms associated with programming.
2. Representation of text inside the computer	Each student writes a sentence in binary and exchanges it with a neighbor. The neighbor translates the sentence into text. Students stand or sit to mime a secret word in binary. Flashlights can also be used to represent binary code.
3. Representation of numbers inside the computer, including the largest and smallest values which can be represented in each of several types	Numbers are converted between binary, hexadecimal, and decimal. The numbers are placed into imaginary bytes in a grid, each imaginary byte having a unique address. (A spreadsheet can be used for this purpose.) Instructions are provided to add and subtract values by address. Some of the resulting numbers should be too large to store in the imaginary byte and will overflow. Students experiment with hexadecimal specification of RGB color values.

Detailed Outline	
Focus	**Sample Lab / Hands-on Activity**
4. Data types: integer, floating point, character/string, and Boolean types and appropriate operations	Evaluate the results of mathematical and logical expressions using integer, floating point, and mixed arithmetic. Evaluate the results of expressions using relational and Boolean operators. Problems should require students to understand order of operations.
5. Program execution	Draw a flowchart showing the process from source to executable, including the return flow for syntax and semantic errors.
6. Programming design techniques	Students are given a word problem and state the input required, output to be produced, and formulas required. The program flow is diagramed before the coding process begins.
7. Programming style	Distribute the written code for a program that has no comments, one-letter non-descriptive variable names, multiple statements on a line, etc. What does this program do? Have students enter and run the program to determine what it does and make necessary changes to employ good style.
8. Programming statements for output	Write a program that displays output.
9. Declaring and using constants and variables of simple types	Write a program that performs operations and generates output.
10. Programming statements for input	Write a program that accepts keyboard input, performs operations and displays the results.
11. Subprograms; scope; parameters for communication between program parts	Write a program that requires the same code to be executed several times and break it up into units. For example: Distribute the words to a favorite song that contains at least two verses and a chorus. Write one subroutine for each verse and the chorus. Call the subroutines in order. Modify the words of the song with parameters.
12. Structured programming: sequence, selection, repetition	Use each structure in a program as it is introduced.
13. Array variables and/or other aggregate data type	Write a program that utilizes an array or other aggregate data type such as a list, set, etc., depending upon the language employed.
14. Object-oriented programming	Discuss advantages of object-oriented programming. Write a simple program using a real-life object (e.g., Rover is an object of type *dog;* properties: *color* and *breed*; methods: *speak* and *beg*.
15. Program results	Predict the output, given a program and sample input.

Topic 12: Web Page Design and Development

Topic Description:

The Web Page Design and Development topic will expose students to the steps needed to create simple web pages. Students will learn to plan and code their web pages and check for usability.

Textbooks and Supplies:

Computer with Internet access; ability to access a text editor; multiple browsers helpful; space on a web server for publication of completed web pages recommended.

Time to Complete:

Student Learning Objectives	Assessment Measures
The student will be able to:	
1. Correctly use HTML tags to create web pages.	Written activity Lab activity
2. Apply styles to HTML documents to control presentation.	Written activity Lab activity
3. Express the design of a web site using standard tools.	Lab activity
4. Create a web site given design specifications.	Lab activity
5. Publish a web site.	Lab activity
6. Apply good design techniques when creating a web site.	Lab activity
7. Explain the different use of server-side and client-side scripts.	Written activity

Assessment Recommendations: An average of 60% from combined assessment measures is required to demonstrate proficiency in course material.	
Lab activities	70%
Written activities, including exams, quizzes, written assignments	30%

Detailed Outline	
Focus	**Sample Lab / Hands-on Activity**
1. Terminology	Identify and define key terms associated with web page development.
2. What makes a good web site?	Visit various web sites and discuss good and bad qualities. A discussion should include ease of navigation, clear and concise message, intended audience, and accessibility.

Detailed Outline	
Focus	**Sample Lab / Hands-on Activity**
3. Design a web site	Choose a topic of common interest. Brainstorm the content as a class activity. Design the home page. Each student selects one content area to design that must represent at least three related pages (e.g., dogs – dog shows, breeds, dogs as pets). A site map and storyboards are created for each content area. Site map the entire site to show all related parts.
4. Discuss HTML tags for these elements: forms, text, graphics, hyperlinks, multimedia, tables	Use each element in an HTML document as it is explained. Check to see that the elements are added in a manner that conforms to World Wide Web Consortium recommendations.
5. Discuss the role of styles for presentation vs. presentation within markup	Apply various styles to the HTML elements as they are explained. Check to see that the styles are added in a manner that conforms to World Wide Web Consortium recommendations.
6. Create a web site from the design	Develop all pages. Check to see that the pages conform to World Wide Web Consortium recommendations.
7. Publish a web site	Upload the pages and visit the site to test the links.
8. Evaluate the site	Discussion: Is this a good web site?
9. Client and server-side scripts	Introduce the necessity for client- and server-side scripts to process form data. Discuss the dynamic capabilities of the web and the need for scripting. Visit a dynamic web site. How does the information that you provide affect what you see?
10. Web development tools	Create a web site using a web development tool. View the HTML and discuss the advantages and disadvantages of WYSIWYG software.

Topic 13: Multimedia

Topic Description:

This multimedia topic will introduce basic file formats for audio, video, and graphics, and creation of multimedia objects. Appropriate use of multimedia is covered.

Textbooks and Supplies:

Any graphical editing program; ability to play audio/video; Internet connection; word processing and presentation software, and graphical, audio, and video editing freeware recommended.

Time to Complete:

Student Learning Objectives	Assessment Measures
The student will be able to:	
1. Explain the differences, advantages and disadvantages between vector and bit-mapped images.	Written activity
2. Based upon the file extension, determine if a given file type is audio, video, or an image and then select the correct tool for viewing the file.	Written activity Lab Activity
3. State the difference between current image formats regarding accessibility and size.	Written activity
4. Convert between image formats.	Lab activity
5. Display a multimedia object within a web page or document.	Lab activity
6. Determine if a given multimedia object can legally be duplicated and/or distributed.	Written activity

Assessment Recommendations: An average of 60% from combined assessment measures is required to demonstrate proficiency in course material.	
Lab activities	70%
Written activities, including tests, quizzes, and written assignments	30%

Detailed Outline	
Focus	**Sample Lab / Hands-on Activity**
1. Terminology	Identify and define key terms associated with the use of multimedia objects.
2. Hardware to support multimedia	Research computer hardware necessary to support multimedia – sound cards, video and digital capture cards, digital cameras, scanners, web-cams, digital video camcorders; processor speed; RAM.

Detailed Outline	
Focus	**Sample Lab / Hands-on Activity**
3. Software to support multimedia, including plug-ins for specific media	Research computer software titles that support multimedia, including shareware and freeware; create a list of the media types that each supports. View a web page without the appropriate plug-in. Download and install the plug-in. Visit the web page again.
4. Digital imaging	Use a digital camera, microphone, web-cam, digital video camcorder, scanner, etc. if available. Insert the multimedia objects into documents when appropriate using word processing and/or presentation software, or by adding the object into a web page using HTML.
5. Create, edit, and save bit-mapped and vector images	Create a simple image in any graphical editor (e.g., *Paint*) to become familiar with the various tools. Draw the same simple picture using vector-graphics (e.g., Drawing tools in *Word*).
6. Bit-mapped representation of images, and resolution	Show an original photograph using gradations from very fine to very coarse resolution. Discuss quality vs. amount of information to be stored for uncompressed bit-map. Hand-digitize an image at various (coarse) resolutions using graph paper, and compare storage requirement vs. quality. Scan a document at lowest and highest resolutions.
7. Vector vs. bit-mapped images	Place both a bit-mapped image and a vector-based image within a document. Stretch the graphics and compare the results.
8. Image file types and compression	Students compress and uncompress objects like a soft sponge and a balloon to observe that the original object remains unchanged when compressed and decompressed even though the method of compression may differ. Point out that the file extensions for images represent different methods of compression. Use a graphical editor to save the same image in several formats. Evaluate the changes in file size and image quality.
9. Accessibility issues	View web pages with images suppressed to open a discussion of accessibility for the visually impaired. Expand the discussion to consider the impact of multimedia decisions for individuals with other disabilities.
10. Audio and video file types	Discuss the process by which sound and video become digitized. Reinforce the tradeoff between quality and file size. Visit a web site that contains multiple versions of an audio file with different qualities. Play each file and compare the quality and file size. Provide a demonstration of working with sound/video if equipment is available.
11. Intellectual property rights; ethical issues surrounding the use of images, sound, and video files	Read together some Terms of Use Agreements and discuss appropriate use. Discuss how the editing process affects “truth” (in images, audio and video). If appropriate software is available, modify a graphic in an inappropriate manner (e.g., use the head of one student and the body of another).

Topic 14: Applications

Topic Description:

Students will be exposed to several major application types, including word processing, spreadsheets, databases, and presentation software. Students will gain skills in using applications and understanding layout.

Textbooks and Supplies:

Appropriate application software.

Time to Complete:

Student Learning Objectives	Assessment Measures
The student will be able to:	
1. Select the appropriate application to use for a particular project.	Written activity Lab activity
2. Create, edit, save, and print a properly-formatted word-processed document.	Lab activity
3. Use a spreadsheet to solve problems, manipulate data, produce charts, and display and print data in an appropriate manner.	Lab activity
4. Design, create, update, query, and generate and print reports using a relational database.	Written activity
5. Using presentation software, design, create, update, print, and deliver an effective presentation (appropriately formatted for use in the classroom).	Lab activity Oral activity

Assessment Recommendations: An average of 60% from combined assessment measures is required to demonstrate proficiency in course material.	
Lab activities and projects	80%
Written activities, including tests, quizzes, and written assignments	20%

Detailed Outline	
Focus	**Sample Lab / Hands-on Activity**
1. Terminology	Explain terminology as needed throughout unit (e.g., range, record, field, etc.)
2. Strengths and weaknesses of various software types	Have students brainstorm various applications they've used and categorize them. Discuss the purpose of various types of software and how they improve certain tasks.
3. Word processing	Students create a document using various formatting and layout features.
4. Use of variables and formulae in spreadsheets	Students create a spreadsheet with both numeric and text data, manipulated with formulae. This activity should be algorithmic in nature, e.g., determining the answers to some problem.

Detailed Outline	
Focus	**Sample Lab / Hands-on Activity**
5. Spreadsheet as a table	Students create a small database using a spreadsheet. Examine the benefits and limitations of using a spreadsheet for database purposes.
6. Relational database design	Students are given a situation, engage in needs analysis, and design a relational database, considering the relationships between tables, data types, and normalization.
7. Creating and maintaining a database	Students create, and modify a database. The implementation must match the design.
8. Retrieving information from a database: queries and SQL	Students use queries to ask specific questions of the database. They revise queries where necessary to narrow their search. The importance of using a structured method of finding information should be stressed.
9. Formatting output: reports	Students create reports to format the information in particular ways. If available, forms can are used to help students understand how to format input.
10. Presentations	Discuss the difference between a standalone presentation and a visual aid, as well as how to format an effective presentation. Students create presentation(s) to demonstrate their ability to apply good design.

References Used to Create these Standards:

Eide, Arvid, et al. *Introduction to Engineering Design and Problem Solving*. 2nd ed. McGraw-Hill, 2001.

"Funny Headlines." *Quotez*. <http://www.quotations.co.uk>

G. Polya, *How to Solve It*, 2nd ed. Princeton University Press, 1957.

Gottleber, Timothy T., and Timothy N. Trainor. *Even More Excellent HTML with XML, XHTML, and JavaScript*. McGraw-Hill, 2003.

ITEA. *Standards for Technological Literacy: Content for the Study of Technology.* International Technology Educational Association, 2000.

Knowlton, Todd. *C++ Basics*, South-Western Computer Education, 2001.

Litvin, Maria, and Gary Litvin. *C++ for You++: An Introduction to Programming and Computer Science*. Skylight Publishing, 1998.

Logo Foundation. 2003. <http://www.logofoundation.org>

Norton, Peter. *Introduction to Computers.* 5th ed. Glencoe/McGraw-Hill, 2003.

Snyder, Lawrence. *Fluency with Information Technology: Skills, Concepts, & Capabilities*. Pearson Education, 2004.

Squeakland. <http://www.squeakland.org>

Acknowledgements:

Mr. Bill Madden, Instructor, Bergen Community College, NJ

Mr. Walter M. Mayes, Librarian, The Girls' Middle School, CA