Computer Programming 12

Guide

NOVA SCOTIA
NOUVELLE-ÉCOSSE

2006

Computer Programming 12

© Crown copyright, Province of Nova Scotia, 2006, 2019
Prepared by the Department of Education and Early Childhood Development

This is the most recent version of the current curriculum materials as used by teachers in Nova Scotia.

Computer Programming 12

Cataloguing-in-Publication Data
Main entry under title.
Computer Programming 12 / Nova Scotia.
Department of Education. English Program Services.
ISBN: XXX-XXX-XXXX
1. Curriculum planning – Nova Scotia.   2. Computer Programming– Study and teaching – Nova Scotia.   I. Nova Scotia. Department of Education. English Program Services.
2006

Acknowledgements

The Nova Scotia Department of Education gratefully acknowledges the contribution of the Computer Programming 12 workgroup to the development of this curriculum guide. Members of the committee include the following:

Theodore Chiasson, Dalhousie University
Vic Lewis, Nova Technologies
Rhonda MacMillan, Halifax Regional School Board
Mark Richards, Annapolis Valley Regional School Board
Dennis Riordan, Dalhousie University
Sharla Rolfe-Hunter, Nova Scotia Community College
John Stewart, Halifax Regional School Board

Contents

Introduction

Background

Computers have become an integral part of everyday life in all aspects of society: at home, in the work place, and in school. As technology continues to evolve, the impact of computers and computer networks will only increase. Computer Programming 12 has been developed in response to this rapid evolution and is designed to give students a broad-based understanding of the fundamental underpinnings of computer technology and of the relationship between computer hardware and the software that controls it.

The Department of Education has made a commitment to provide a broad-based, quality education in the public school system and to expand the range of programming to better meet the needs of all students. The department is working in collaboration with school boards and other partners in education, business, industry, the community, and government to develop a variety of new courses.

New course options draw from and contribute to students' knowledge and skills in more than one discipline. Students synthesize and apply knowledge and skills acquired in other courses, including courses in English language arts, social studies, science, arts, mathematics, and technology.

New course options provide increased opportunities for senior high school students to

- earn the credits they require to attain a high school graduation diploma
- diversify their course options
- prepare for varied post-secondary destinations

Course options are designed to

- appeal to all high school students
- assist students in making connections among school, the community, and the workplace
- enable students to explore a range of career options

These courses offer students increased opportunities for hands-on experiences and for using technology within a variety of subject areas to expand and develop their learning and skills.

Rationale for Computer Programming 12

Today, computer science is not only an area of study in its own right but an important supporting area for many other disciplines. The dramatic growth of computing and the enormous impact that computing is making on virtually every field of study are well documented. The pervasiveness of computing and information technology creates a responsibility to offer an in-depth introduction to the discipline.

There is growing student interest in high school technology courses due to the proliferation of computers throughout society. Students interested in pursuing studies or careers in practically any discipline recognize that their understanding of computing technology will play an increasingly important role in their success. Computer Programming 12 introduces students to analysis, design, and problem solving skills that will greatly enhance their ability to use a wide variety of computing technologies available today, as well as to adapt to the inevitable changes in computing technology they will encounter throughout their careers.

The study of computer programming can lead to a broad range of potential occupations within computer and engineering disciplines and can also be applied across a broad range of other disciplines. Examples of potential occupations and fields include the following:

- applications programmer
- programmer/analyst
- computer architect
- games designer
- computer engineer
- software engineer
- computer scientist
- web master
- bio-informatics
- robotics
- medical informatics
- electronic commerce
- technology law
- computer security/forensics
- network architecture
- multimedia/graphic design

Due to the ubiquity of computing in most disciplines, fundamental problem-solving and programming skills will serve as an asset for practically any field of study.

While not a prerequisite for entry into post-secondary computing programs in Nova Scotia, Computer Programming 12 represents a solid preparation for entry into a community college diploma in information technology or a university degree program in computer science.

The Nature of Computer Programming 12

Computer Programming 12 is based on a learning outcomes framework that identifies knowledge, skills, and capabilities that students are expected to demonstrate as a result of their learning experiences. The structure of the learning outcomes is based on fundamental principles of computer programming and is therefore not specific to any one language.

Object-oriented analysis and design techniques are introduced as high-level tools that facilitate problem understanding and algorithm development independent of the target implementation language. Use of an object-oriented programming language will facilitate a natural transition from analysis through to design and implementation. While Java is supported as the programming language for Computer Programming 12, schools may alternatively choose, for example, C++, Visual Basic, or Python.

Computer Programming 12 is made up of four modules:

- **Module 1: Problem Solving in Computer Programming**
- **Module 2: Fundamentals of Programming**
- **Module 3: Applied Problem Solving**
- **Module 4: Project Development**

The project-based module is focussed on analysis, design, implementation, and deployment of a functional solution to a real-world problem. This allows Computer Programming 12 to engage students in community-based outreach projects that provide both a hands-on learning experience for the students and a lasting benefit for the community.

Course Designation

Computer Programming 12 is an academic credit and an eligible technology credit to meet graduation requirements. Students who complete two modules will receive a half credit, while all four modules must be completed to receive a full credit. Course codes for Computer Programming 12 are

- 100073 Computer Programming 12
- 100075 Computer Programming 12A
- 100076 Computer Programming 12B

Course Design and Components

Features of Computer Programming 12

Computer Programming 12 is characterized by the following features:

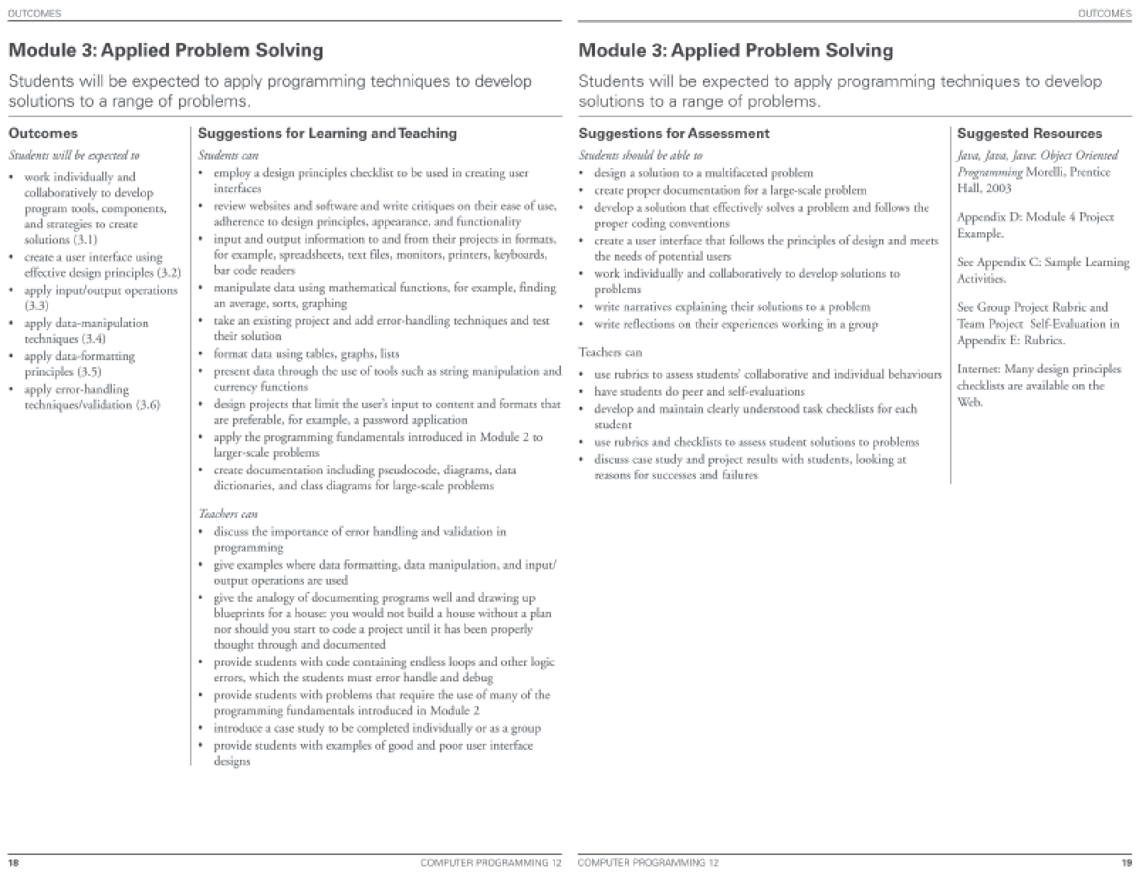
- an emphasis on integrating, applying, and reinforcing the knowledge, skills, and attitudes developed in other courses
- a connection to the Essential Graduation Learnings
- a refining of career-planning skills to explore a wide range of pathways from school
- a relationship to the community and workplace with a focus on using real community and workplace problems and situations as practical contexts for the application of knowledge and skills and for further learning
- hands-on, project-based learning experiences
- development of personal and interpersonal skills required for personal and career success
- use of technology as an integral part of the course

The Four-Column Spread

The curriculum for this course has been organized into four columns for several reasons:

- The organization illustrates how learning experiences flow from the outcomes.
- The relationship between the outcomes and assessment strategies is immediately apparent.
- Related and interrelated outcomes can be grouped together.
- The range of strategies for learning and teaching associated with specific outcomes can be scanned easily.
- The organization provides multiple ways of reading the document or of searching for specific information.

An example of the two-page, four-column spread appears on the next page.



OUTCOMES

Module 3: Applied Problem Solving

Students will be expected to apply programming techniques to develop solutions to a range of problems.

Outcomes

Students will be expected to

- work individually and collaboratively to develop program tools, components, and strategies to create solutions (3.1)
- create a user interface using effective design principles (3.2)
- apply input/output operations (3.3)
- apply data-manipulation techniques (3.4)
- apply data-formatting principles (3.5)
- apply error-handling techniques/validation (3.6)

Suggestions for Learning and Teaching

Students can

- employ a design principles checklist to be used in creating user interfaces
- review websites and software and write critiques on their ease of use, adherence to design principles, appearance, and functionality
- input and output information to and from their projects in formats, for example, spreadsheets, text files, monitors, printers, keyboards, bar code readers
- manipulate data using mathematical functions, for example, finding an average, sorts, graphing
- take an existing project and add error-handling techniques and test their solution
- format data using tables, graphs, lists
- present data through the use of tools such as string manipulation and currency functions
- design projects that limit the user's input to content and formats that are preferable, for example, a password application
- apply the programming fundamentals introduced in Module 2 to larger-scale problems
- create documentation including pseudocode, diagrams, data dictionaries, and class diagrams for large-scale problems

Teachers can

- discuss the importance of error handling and validation in programming
- give examples where data formatting, data manipulation, and input/output operations are used
- give the analogy of documenting programs well and drawing up blueprints for a house: you would not build a house without a plan nor should you start to code a project until it has been properly thought through and documented
- provide students with code containing endless loops and other logic errors, which the students must error handle and debug
- provide students with problems that require the use of many of the programming fundamentals introduced in Module 2
- introduce a case study to be completed individually or as a group
- provide students with examples of good and poor user interface designs

18 COMPUTER PROGRAMMING 12

OUTCOMES

Module 3: Applied Problem Solving

Students will be expected to apply programming techniques to develop solutions to a range of problems.

Suggestions for Assessment

Students should be able to

- design a solution to a multifaceted problem
- create proper documentation for a large-scale problem
- develop a solution that effectively solves a problem and follows the proper coding conventions
- create a user interface that follows the principles of design and meets the needs of potential users
- work individually and collaboratively to develop solutions to problems
- write narratives explaining their solutions to a problem
- write reflections on their experiences working in a group

Teachers can

- use rubrics to assess students' collaborative and individual behaviours
- have students do peer and self-evaluations
- develop and maintain clearly understood task checklists for each student
- use rubrics and checklists to assess student solutions to problems
- discuss case study and project results with students, looking at reasons for successes and failures

Suggested Resources

Java, Java, Java: Object Oriented Programming Morelli, Prentice Hall, 2003

Appendix D: Module 4 Project Example.

See Appendix C: Sample Learning Activities.

See Group Project Rubric and Team Project Self-Evaluation in Appendix E: Rubrics.

Internet: Many design principles checklists are available on the Web.

COMPUTER PROGRAMMING 12 19

Column One: Outcomes

This column provides specific curriculum outcomes for the general curriculum outcome that appears across the top of the page. While the outcomes may be clustered, they are not necessarily sequential.

Column Two: Suggestions for Learning and Teaching

This column offers a range of strategies from which teachers and students may choose. Suggested learning experiences can be used in various combinations to help students achieve an outcome or outcomes. The suggested strategies may also provide a springboard for teachers to choose other strategies that would be effective for their students. It is not necessary to use all the suggestions that are included; nor is it necessary for all students to be involved in the same learning experience.

Column Three: Suggestions for Assessment

This column provides suggestions for assessing achievement of the outcomes in Column One; these are often linked to the Suggestions for Learning and Teaching column. The suggestions are only samples; for more information, read the section Assessing and Evaluating Student Learning and see Appendix E for sample assessment tools.

Column Four: Suggested Resources

This column, entitled Suggested Resources, contains a variety of information related to the items in the other columns, including suggested resources, elaborations on strategies, successes, cautions, and definitions.

Outcomes

Essential Graduation Learnings and Computer Programming 12

The Atlantic provinces worked together to identify the abilities and areas of knowledge that they considered essential for students graduating from high school. These are referred to as Essential Graduation Learnings. Details may be found in the document *Public School Programs.*

Aesthetic Expression

Graduates will be able to respond with critical awareness to various forms of the arts and be able to express themselves through the arts.

Students will be expected to

- create a user interface using effective design principles (3.2)

Citizenship

Graduates will be able to assess social, cultural, economic, and environmental interdependence in a local and global context.

Students will be expected to

- demonstrate an understanding of ethical, moral, and legal issues in information technology (1.6)

Communication

Graduates will be able to use the listening, viewing, speaking, reading, and writing modes of language(s) as well as mathematical and scientific concepts and symbols to think, learn, and communicate effectively.

Students will be expected to

- define a problem in explicit terms using object-oriented analysis (1.3)
- create documentation associated with the project (4.6)
- present the solution (4.8)

Personal Development

Graduates will be able to continue to learn and to pursue an active, healthy lifestyle.

Students will be expected to

- investigate a range of related career opportunities (1.7)
- demonstrate the collaborative skills and behaviours required to work with others (4.3)
- reflect on the solution, the process, and their own learning (4.9)

Problem Solving

Graduates will be able to use the strategies and processes needed to solve a wide variety of problems, including those requiring language, mathematical, and scientific concepts.

Students will be expected to

- demonstrate an understanding of how data structures are used to solve problems (2.3)
- use appropriate methods and terms to develop a plan to solve a problem (2.4)

Technological Competence

Graduates will be able to use a variety of technologies, demonstrate an understanding of technological applications, and apply appropriate technologies for solving problems.

Students will be expected to

- apply data-manipulation techniques (3.4)
- apply data-formatting principles (3.5)
- apply error handling techniques/validation (3.6)

Computer Programming 12 Unifying Concepts

Module 1: Students will be expected to understand and apply the basic skills and processes of problem solving using computer programming.

Module 2: Students will be expected to identify problems, select effective strategies, and plan solutions.

Module 3: Students will be expected to apply programming techniques to develop solutions to a range of problems.

Module 4: Students will be expected to work collaboratively to define and solve a realistic problem by creating a solution.

Specific Curriculum Outcomes

Module 1: Problem Solving in Computer Programming

Students will be expected to

- demonstrate an understanding of the role of number systems in data storage (1.1)
- apply mathematical concepts, including Boolean logic and operators (1.2)
- define a problem in explicit terms using object-orientated analysis (1.3)
- identify and outline strategies to solve a range of problems (1.4)
- apply a range of problem-solving skills (1.5)
- demonstrate an understanding of ethical, moral, and legal issues in information technology (1.6)
- investigate a range of related career opportunities (1.7)

Module 2: Fundamentals of Programming

Students will be expected to

- demonstrate an understanding of the syntax and features of a programming language (2.1)
- identify and frame problems (2.2)
- demonstrate an understanding of how data structures are used to solve problems (2.3)
- use appropriate methods and terms to develop a plan to solve a problem (2.4)
- apply and plan to solve a problem using a programming language (2.5)
- demonstrate an understanding of the effectiveness of other people's programs and documentation (2.6)

Module 3: Applied Problem Solving

Students will be expected to

- work individually and collaboratively to develop program tools, components, and strategies to create solutions (3.1)
- create a user interface using effective design principles (3.2)
- apply input/output operations (3.3)
- apply data-manipulation techniques (3.4)
- apply data-formatting principles (3.5)
- apply error-handling techniques/validation (3.6)

Module 4: Project Development

Students will be expected to

- analyse a problem (4.1)
- develop a project plan, including definition, scope, roles, resources, steps, and deadlines, for a solution (4.2)
- demonstrate the collaborative skills and behaviours required to work with others (4.3)
- identify information needs and locate, evaluate, and select resources (4.4)
- build and deploy a solution (4.5)
- create documentation associated with the project (4.6)
- test and refine the solution (4.7)
- present the solution (4.8)
- reflect on the solution, the process, and their own learning (4.9)
- explore various educational and career paths in information technology fields (4.10)

Specific Curriculum Outcomes by Module

Module 1: Problem Solving in Computer Programming

Students will be expected to understand and apply the basic skills and processes of problem solving using computer programming.

Outcomes

Students will be expected to

- demonstrate an understanding of the role of number systems in data storage (1.1)
- apply mathematical concepts, including Boolean logic and operators (1.2)
- define a problem in explicit terms using object-oriented analysis (1.3)
- identify and outline strategies to solve a range of problems (1.4)
- apply a range of problem-solving skills (1.5)
- demonstrate an understanding of ethical, moral, and legal issues in information technology (1.6)
- investigate a range of related career opportunities (1.7)

Suggestions for Learning and Teaching

Students can

- convert between binary, base ten, and hexadecimal
- use operators, logic gates, and truth tables to solve problems
- use multiple methods such as algorithms, flow charts, pseudocode, and unified modelling language to identify key elements of a problem and outline possible solutions
- create class diagrams to define a problem in explicit object orientated terms
- prepare a chart of problem-solving strategies and rate them based on effectiveness, use of time, and reliability

Teachers can

- explain the roles of other number systems in computing, such as binary in memory storage and hexadecimal in web page colours
- model the use of Boolean logic through examples like Internet searches and light switches
- explain the need to model problems and provide students with modelling tools such as flow charts, unified modelling language, pseudocode, class diagrams, Venn diagrams, logic gates, and truth tables
- provide students with a wide range of problems and potential tools to solve problems
- have students write the problem given only someone else’s pseudocode for the problem
- have students look for logic errors in flow charts, pseudocode and class diagrams

Module 1: Problem Solving in Computer Programming

Students will be expected to understand and apply the basic skills and processes of problem solving using computer programming.

Suggestions for Assessment

Students can

- complete a series of quizzes or tests on applying multiple number systems, use of operators, and logic
- prepare a survey of IT occupations and roles for their portfolios
- create an IT résumé
- debate an ethical IT issue
- complete assignments involving the modelling and solving of problems

Teachers can

- observe students and assess their collaborative behaviour using a rubric
- have students write learning narratives for inclusion in a portfolio
- develop and provide, for students, activity checklists based on specific outcomes

Suggested Resources

See Web Resources in Appendix I: Resources.

See Group Presentation Rubric in Appendix E: Rubrics.

Appendix C: Sample Learning Activities articles includes Creating an Object.

Mathematics for New Technologies Hutchison/Yannotta, Pearson, 2004

Problem Solving and Programming Concepts Sprankle, Prentice Hall, 2003

Internet: Many online resources to support this and other modules may be found at <www.dal.ca/CP12>.

Module 1: Problem Solving in Computer Programming

Students will be expected to understand and apply the basic skills and processes of problem solving using computer programming.

Outcomes

Students will be expected to

- demonstrate an understanding of the role of number systems in data storage (1.1)
- apply mathematical concepts, including Boolean logic and operators (1.2)
- define a problem in explicit terms using object-oriented analysis (1.3)
- identify and outline strategies to solve a range of problems (1.4)
- apply a range of problem-solving skills (1.5)
- demonstrate an understanding of ethical, moral, and legal issues in information technology (1.6)
- investigate a range of related career opportunities (1.7)

Suggestions for Learning and Teaching

Students can

- work in groups or individually to research and create a presentation on an ethical issue related to the computing field
- hold a debate on a current social or ethical issue relating to computing
- interview an IT professional and write a report about that person's field
- search for an information technology job listing and create an application including a proper cover letter and résumé
- begin a skills inventory for their IT résumé

Teachers can

- discuss the limitations of finite representations such as imprecision of floating-point representations and roundoff error
- have students write pseudocode for an algorithm of an everyday task such as making a sandwich or coming to class from the school's front door and then have students attempt to follow the pseudocode directions
- give different groups the same problem and have each group use a different method to model and solve the problem
- hold debates or open forums on social and ethical issues that are current in the world of computing
- invite IT professionals from a variety of fields to speak to students or act as mentors

Module 1: Problem Solving in Computer Programming

Students will be expected to understand and apply the basic skills and processes of problem solving using computer programming.

Suggestions for Assessment

Students can

- complete a series of quizzes or tests on applying multiple number systems, use of operators and logic
- prepare a survey of IT occupations and roles for their portfolios
- create an IT résumé
- debate an ethical IT issue
- complete assignments involving the modelling and solving of problems

Teachers can

- observe students and assess their collaborative behaviour using a rubric
- have students write learning narratives for inclusion in a portfolio
- develop and provide, for students, activity checklists based on specific outcomes

Suggested Resources

See Appendix F: Legal, Moral, and Ethical Issues.

See Web Resources in Appendix I: Resources.

See Group Project Rubric in Appendix E: Rubrics.

See Introduction to Object-Oriented Activities, Appendix C: Sample Learning Activities.

Mathematics for New Technologies Hutchison/Yannotta, Pearson, 2004

Problem Solving and Programming Concepts Sprankle, Prentice Hall, 2003

Module 2: Fundamentals of Programming

Students will be expected to identify problems, select effective strategies, and plan solutions.

Outcomes

Students will be expected to

- demonstrate an understanding of the syntax and features of a programming language (2.1)
- identify and frame problems (2.2)
- demonstrate an understanding of how data structures are used to solve problems (2.3)
- use appropriate methods and terms to develop a plan to solve a problem (2.4)
- apply and plan to solve a problem using a programming language (2.5)
- demonstrate an understanding of the effectiveness of other people's programs and documentation (2.6)

Suggestions for Learning and Teaching

Students can

- read and understand a problem description, purpose, and goals
- decompose a problem into classes and define relationships and responsibilities of those classes
- declare constants, variables, methods, classes, and parameters
- understand and modify existing code
- analyse and express problems in mathematical terms and use the information to create data dictionaries (account for all objects required in a program)
- role-play problems to focus on conditions, relationships, classes, variables, and objects
- comment existing code samples
- examine data types appropriate to solve specific problems
- model problems using flow charts, unified modelling language, pseudocode, or object diagrams
- design, write, and test a rudimentary object-oriented program

Teachers can

- present a matrix of the features of the language to be used, with examples of their use
- provide samples of effective and ineffective programs
- provide students with problems where one student writes the pseudocode solution and a partner writes the actual code based on the pseudocode without having seen the original problem
- stress the importance of good program design and of following coding conventions (see Web Resources in Appendix G: Resources)
- introduce a video game project that students will code together as a class, to apply programming fundamentals and introduce syntax
- provide students with tutorials, handouts, or slides and direct them to Internet resources that discuss fundamental programming concepts and explain language syntax
- assign students to teach specific concepts and present solutions

Module 2: Fundamentals of Programming

Students will be expected to identify problems, select effective strategies, and plan solutions.

Suggestions for Assessment

Students can

- create and document many small programs following good coding conventions
- work in groups and individually to define and solve problems
- read a problem and write the problem in their own words
- develop a portfolio of assignments, tests, and projects to monitor their own progress

Teachers can

- use rubrics to assess student learning and participation
- provide students with skill-based checklists
- create a series of self-scoring quizzes to help students monitor their progress and to preparer for tests
- create partially completed projects that students can debug and complete
- provide code with error for students to debug
- provide small, sequential laboratory assignments to prepare students for weekly quizzes and to work toward a larger project

Suggested Resources

Problem Solving and Programming Concepts Sprankle, Prentice Hall, 2003

Java, Java, Java: *Object-Oriented Programming* Morelli, Prentice Hall, 2003

See Web Resources in Appendix I: Resources for a Unified Modeling Language (UML) tutorial.

See Appendix E: Rubrics.

See Appendix A: Background and Information Resource.

See Appendix H: Glossary of Terms.

See Teaching Kids Programming in Appendix A: Background and Information Resources.

Module 3: Applied Problem Solving

Students will be expected to apply programming techniques to develop solutions to a range of problems.

Outcomes

Students will be expected to

- work individually and collaboratively to develop program tools, components, and strategies to create solutions (3.1)
- create a user interface using effective design principles (3.2)
- apply input/output operations (3.3)
- apply data-manipulation techniques (3.4)
- apply data-formatting principles (3.5)
- apply error-handling techniques/validation (3.6)

Suggestions for Learning and Teaching

Students can

- employ a design principles checklist to be used in creating user interfaces
- review websites and software and write critiques on their ease of use, adherence to design principles, appearance, and functionality
- input and output information to and from their projects in formats, for example, spreadsheets, text files, monitors, printers, keyboards, bar code readers
- manipulate data using mathematical functions, for example, finding an average, sorts, graphing
- take an existing project and add error-handling techniques and test their solution
- format data using tables, graphs, lists
- present data through the use of tools such as string manipulation and currency functions
- design projects that limit the user's input to content and formats that are preferable, for example, a password application
- apply the programming fundamentals introduced in Module 2 to larger-scale problems
- create documentation including pseudocode, diagrams, data dictionaries, and class diagrams for large-scale problems

Teachers can

- discuss the importance of error handling and validation in programming
- give examples where data formatting, data manipulation, and input/output operations are used
- give the analogy of documenting programs well and drawing up blueprints for a house: you would not build a house without a plan nor should you start to code a project until it has been properly thought through and documented
- provide students with code containing endless loops and other logic errors, which the students must error handle and debug
- provide students with problems that require the use of many of the programming fundamentals introduced in Module 2
- introduce a case study to be completed individually or as a group
- provide students with examples of good and poor user interface designs

Module 3: Applied Problem Solving

Students will be expected to apply programming techniques to develop solutions to a range of problems.

Suggestions for Assessment

Students should be able to

- design a solution to a multifaceted problem
- create proper documentation for a large-scale problem
- develop a solution that effectively solves a problem and follows the proper coding conventions
- create a user interface that follows the principles of design and meets the needs of potential users
- work individually and collaboratively to develop solutions to problems
- write narratives explaining their solutions to a problem
- write reflections on their experiences working in a group

Teachers can

- use rubrics to assess students' collaborative and individual behaviours
- have students do peer and self-evaluations
- develop and maintain clearly understood task checklists for each student
- use rubrics and checklists to assess student solutions to problems
- discuss case study and project results with students, looking at reasons for successes and failures

Suggested Resources

Java, Java, Java: *Object Oriented Programming* Morelli, Prentice Hall, 2003

Appendix D: Module 4 Project Example.

See Appendix C: Sample Learning Activities.

See Group Project Rubric and Team Project Self-Evaluation in Appendix E: Rubrics.

Internet: Many design principles checklists are available on the Web.

Module 4: Project Development

Students will be expected to work collaboratively to define and solve a realistic problem by creating a solution.

Outcomes

Students will be expected to

- analyse a problem (4.1)
- develop a project plan, including definition, scope, roles, resources, steps, and deadlines, for a solution (4.2)
- demonstrate the collaborative skills and behaviours required to work with others (4.3)
- identify information needs and locate, evaluate, and select resources (4.4)
- build and deploy a solution (4.5)
- create documentation associated with the project (4.6)
- test and refine the solution (4.7)
- present the solution (4.8)
- reflect on the solution, the process, and their own learning (4.9)
- explore various educational and career paths in information technology fields (4.10)

Suggestions for Learning and Teaching

Students can

- read and respond to scenarios describing a range of processes, for example: team ground rules, roles and responsibilities, identifying helpful team behaviours, decision making, conflict resolution
- discuss as a group and decide upon a project topic
- form a team where roles and responsibilities are decided upon and team ground rules are established
- read a request for proposal and outline what the problem to be solved involves
- design a project plan outlining the resources, personnel, and time required to develop a solution
- develop a response for a request for proposal, to be submitted to a fictional or real organization, that will identify steps, sequence, and schedule
- hold interviews, with their client, where project particulars are discussed and the nature of the project is negotiated

Teachers can

- provide students with a request for proposals that outline a real-world problem that requires a programming solution
- play the role of a client for a fictional business or organization that wants to hire students to solve their programming needs
- help students choose appropriate projects and give advice on the scope of projects

Module 4: Project Development

Students will be expected to work collaboratively to define and solve a realistic problem by creating a solution.

Suggestions for Assessment

Students can

- maintain an ongoing project plan
- create a team identity document that contains the roles, responsibilities, and ground rules that the group decided on and implemented
- write journal entries discussing their experiences in creating the project

Teachers can

- provide students with peer and self-evaluation rubrics
- identify and address individual/team concerns as they arise
- set milestones for groups where portions of the project must be submitted for evaluation
- develop and maintain clearly understood task checklists for each student
- monitor students' collaborative behaviours using checklists and rubrics

Suggested Resources

See Appendix E: Rubrics.

See Appendix B: Group Work and Team Building.

See Team Project Self-Evaluation in Appendix E: Rubrics.

See Appendix D: Module 4 Project Example.

See Appendix C: Sample Learning Activities.

Module 4: Project Development

Students will be expected to work collaboratively to define and solve a realistic problem by creating a solution.

Outcomes

Students will be expected to

- analyse a problem (4.1)
- develop a project plan, including definition, scope, roles, resources, steps, and deadlines, for a solution (4.2)
- demonstrate the collaborative skills and behaviours required to work with others (4.3)
- identify information needs and locate, evaluate, and select resources (4.4)
- build and deploy a solution (4.5)
- create documentation associated with the project (4.6)
- test and refine the solution (4.7)
- present the solution (4.8)
- reflect on the solution, the process, and their own learning (4.9)
- explore various educational and career paths in information technology fields (4.10)

Suggestions for Learning and Teaching

Students can

- create a user manual and help files to be included as part of the project
- develop a working solution to a large-scale programming problem that uses object-oriented analysis and design
- develop a solution that follows good programming conventions such as commenting, error handling, validation, and naming conventions
- save and maintain multiple versions of their project
- test and debug their project until it functions to the client's specifications
- create a marketing campaign for their product
- develop a multimedia presentation for their project to be presented to clients and class members
- write reflections on the software development cycle and on group dynamics

Teachers can

- provide students with team-building activities that force students to study helpful group behaviours
- survey students to identify specific skills
- provide a rational for and examples of effective project planning
- negotiate collaboration with independent learners
- make contact with community businesses and organizations and have them act as clients for the project
- bring in guest speakers from IT consulting firms, community colleges, and universities to speak with students
- act as facilitators, keeping student projects within the scope intended, aiding groups in finding and using resources, and helping settle disagreements between student groups and clients and within groups

Module 4: Project Development

Students will be expected to work collaboratively to define and solve a realistic problem by creating a solution.

Suggestions for Assessment

Students can

- complete self- and peer evaluations
- present their final project
- write journal entries discussing their experiences in creating the project

Teachers can

- evaluate final projects based on a list consisting of acceptance factors and features
- provide students with formal/informal feedback on their projects
- invite community businesses and organizations involved in the project to take part in the presentations and acquire feedback from them

Suggested Resources

Suggested Resources

See Appendix E: Rubrics.

See Appendix B: Group Work and Team Building.

See Appendix D: Module 4 Project Example.

See Appendix C: Sample Learning Activities.

Contexts for Learning and Teaching

Principles of Learning

The public school program is based on principles of learning that teachers and administrators should use as the basis of the experiences they plan for their students. These principles include the following:

1. Learning is a process of actively constructing knowledge.

   Therefore, teachers and administrators have a responsibility to
   - create environments and plan experiences that foster inquiry, questioning, predicting, exploring, collecting, educational play, and communicating
   - engage learners in experiences that encourage their personal construction of knowledge, for example, hands-on science and math, drama, creative movement, artistic representation, and writing and talking learning activities
   - provide learners with experiences that actively involve them and are personally meaningful

2. Students construct knowledge and make it meaningful in terms of their prior knowledge and experiences.

   Therefore, teachers and administrators have a responsibility to
   - find out what students already know and can do
   - create learning environments and plan experiences that build on learners' prior knowledge
   - ensure that learners are able to see themselves reflected in the learning materials used in the school
   - recognize, value, and use the great diversity of experiences and information students bring to school
   - provide learning opportunities that respect and support students' racial, cultural, and social identities
   - ensure that students are invited or challenged to build on prior knowledge, integrating new understandings with existing understandings

3. Learning is enhanced when it takes place in a social and collaborative environment.

   Therefore, teachers and administrators have a responsibility to
   - ensure that talk, group work, and collaborative ventures are central to class activities
   - see that learners have frequent opportunities to learn from and with others
   - structure opportunities for learners to engage in diverse social interactions with peers and adults
   - help students to see themselves as members of a community of learners

4. Students need to continue to view learning as an integrated whole.

   Therefore, teachers and administrators have a responsibility to
   - plan opportunities to help students make connections across the curriculum and with the world outside and structure activities that require students to reflect on those connections
   - invite students to apply strategies from across the curriculum to solve problems in real situations

5. Learners must see themselves as capable and successful.

   Therefore, teachers and administrators have a responsibility to
   - provide activities, resources, and challenges that are developmentally appropriate to the learner
   - communicate high expectations for achievement to all students
   - encourage risk taking in learning
   - ensure that all students experience genuine success on a regular basis
   - value experimentation and treat approximation as signs of growth
   - provide frequent opportunities for students to reflect on and describe what they know and can do
   - provide learning experiences and resources that reflect the diversity of the local and global community
   - provide learning opportunities that develop self-esteem

6. Learners have different ways of knowing and representing knowledge.

   Therefore, teachers and administrators have a responsibility to
   - recognize each learner's preferred ways of constructing meaning and provide opportunities for exploring alternative ways
   - plan a wide variety of open-ended experiences and assessment strategies
   - recognize, acknowledge, and build on students' diverse ways of knowing and representing their knowledge
   - structure frequent opportunities for students to use various art forms—music, drama, visual arts, dance, movement, crafts—as a means of exploring, formulating, and expressing ideas

7. Reflection is an integral part of learning.

   Therefore, teachers and administrators have a responsibility to
   - challenge their beliefs and practices based on continuous reflection
   - reflect on their own learning processes and experiences

- encourage students to reflect on their learning processes and experiences
- encourage students to acknowledge and articulate their learning
- help students use their reflections to understand themselves as learners, make connections with other learning, and proceed with learning

Learning Styles and Needs

Learners have many ways of learning, knowing, understanding, and creating meaning. Research into links between learning styles and preferences and the physiology and function of the brain has provided educators with a number of helpful concepts of and models for learning. Howard Gardner, for example, identifies eight broad frames of mind or intelligences: linguistic, logical/ mathematical, visual/spatial, body/kinesthetic, musical, interpersonal, intrapersonal, and naturalistic. Gardner believes that each learner has a unique combination of strengths and weaknesses in these eight areas, but that the intelligence can be more fully developed through diverse learning experiences. Other researchers and education psychologists use different models to describe and organize learning preferences.

Students' ability to learn is also influenced by individual preferences and needs within a range of environmental factors, including light, temperature, sound levels, availability of food and water, proximity to others, opportunities to move around, and time of day.

How students receive and process information and the ways they interact with peers and their environment in specific contexts are both indicators and shapers of their preferred learning styles. Most learners have a preferred learning style, depending on the situation and the type and form of information the student is dealing with, just as most teachers have a preferred teaching style, depending on the context. By reflecting on their own styles and preferences as learners and as teachers in various contexts, teachers can

- build on their own teaching-style strengths
- develop awareness of and expertise in a number of learning and teaching styles and preferences
- identify and allow for differences in student learning styles and preferences
- identify and allow for the needs of students for whom the range of ways of learning is limited
- organize learning experiences to accommodate the range of ways in which students learn

Learning experiences and resources that engage students' multiple ways of understanding allow them to become aware of and reflect on their learning processes and preferences. To enhance their opportunities for success, students need

- a variety of learning experiences to accommodate their diverse learning styles and preferences
- opportunities to reflect on their preferences and the preferences of others to understand how they learn best and how others learn differently
- opportunities to explore, apply, and experiment with learning styles other than those they prefer, in learning contexts that encourage risk taking
- opportunities to return to preferred learning styles at critical stages in their learning
- opportunities to reflect on other factors that affect their learning, for example, environmental, emotional, sociological, cultural, and physical factors
- a flexible time line within which to complete their work

Meeting the Needs of All Students

Learners require inclusive classrooms, where a wide variety of learning experiences ensure that all students have equitable opportunities to reach their potential.

In designing learning experiences, teachers must accommodate the learning needs, preferences, and strengths of individuals and consider the abilities, experiences, interests, and values that they bring to the classroom. In recognizing and valuing the diversity of students, teachers should consider ways to

- create a climate and design learning experiences to affirm the dignity and worth of all learners in the classroom community
- consider the social and economic situations of all learners
- acknowledge racial and cultural uniqueness
- model the use of inclusive language, attitudes, and actions supportive of all learners
- adapt classroom organization, teaching strategies, assessment practices, time, and learning resources to address learners' needs and build on their strengths
- provide opportunities for learners to work in a variety of contexts, including mixed-ability groupings
- identify and apply strategies and resources that respond to the range of students' learning styles and preferences
- build on students' individual levels of knowledge, skills, and attitudes
- use students' strengths and abilities to motivate and support their learning
- provide opportunities for students to make choices that will broaden their access to a range of learning experiences
- acknowledge the accomplishment of learning tasks, especially those that learners believed were too challenging for them

Teachers must adapt learning contexts, including environment, strategies for learning, and strategies for assessment, to provide support and challenge for all students, using curriculum outcomes to plan learning experiences appropriate to students' individual learning needs. When these changes are not sufficient for a student to meet designated outcomes, an individual program plan may be developed. For more detailed information, see *Special Education Policy Manual* (1996), Policy 2.6.

A range of learning experiences, teaching and learning strategies, motivation, resources, and environments provide expanded opportunities for all learners to experience success as they work toward the achievement of designated outcomes. Many of the learning experiences suggested in this guide provide access for a wide range of learners, simultaneously emphasizing both group support and individual activity. Similarly, the suggestions for a variety of assessment practices provide multiple ways for students to demonstrate their achievements.

The Role of Technology

Vision for the Integration of Information

The outcomes in Computer Programming 12 are, by nature, technology dependent; students are required to utilize an object-orientated programming language and integrated development environment. Students also need access to the information and communication technologies available in schools to facilitate learning across the curriculum.

The Nova Scotia Department of Education has articulated five strands in the learning outcomes framework areas for the integration of information and communication technology within Public School Programs.

Basic Operations and Concepts: concepts and skills associated with the safe and efficient operation of a range of information technologies

Social, Ethical, and Human Issues: the understanding associated with the use of information/communication technology, which encourages in students a commitment to pursue personal and social good, particularly to build and improve their learning environments and to foster strong relationships with their peers and others who support their learning

Productivity: the efficient selection and use of information and communication technology to perform tasks such as the exploration of ideas, data collection, data manipulation, including the discovery of patterns and relationships, problem solving, and the representation of learning

Communication: specific, interactive technology use that supports student collaboration and sharing through communication

Research, Problem Solving, and Decision Making: students' organization, reasoning, and evaluation of their learning, which rationalize their use of information and communication technology

The Computer Programming 12 Learning Environment

The Classroom

Learning in Computer Programming 12 should take place, for the most part, in a computer laboratory. There should be one computer per student, and students should have access to a printer. Arranging the computers in a u-shape with the open end of the facing the front of the class allows the teacher to view all screens simultaneously and also allows all students clear access to the front of the room to view the teacher, data screen, or board. Tables or counter space are necessary for students to carry out group work, write documentation, and take notes. It is anticipated that a great deal of file sharing will need to take place in this course and that file storage requirements may become considerable. It would be helpful for computers to be networked and for students have access to in and out boxes stored on a server. An alternative to this would be the use of memory sticks, which would also allow students to store and transfer files with the added benefit of being able to take their work home.

The Learning Culture

It is important to establish a culture in the Computer Programming 12 classroom where critical thinking, problem solving, and collaboration are valued and encouraged. Students should perceive the teacher as an instructor when necessary, but more frequently the teacher should be perceived as a facilitator, guiding and encouraging students as they acquire problem-solving, collaboration, and technical skills. The continuous evolution of information technology in general, and computer programming in particular, requires the teacher to be a lifelong learner, to apply prior knowledge, to be actively curious, and to model these qualities for students.

By taking an active learning approach, teachers become part of the learning community and communicate to students that problem solving is a dynamic process with multiple paths to success. It is essential that students be free to collaborate and feel comfortable to take risks in their learning. Students should be encouraged to peer teach, and teachers should be comfortable learning alongside their students.

Assessing and Evaluating Student Learning

Assessment is the systematic process of gathering information on student learning.

Evaluation is the process of analysing, reflecting upon, and summarizing assessment information and making judgments or decisions based upon the information gathered.

The Principles of Assessment and Evaluation articulated in the document *Public School Programs* should be used as the basis of assessment and evaluation, policies, procedures, and practices.

Effective Assessment and Evaluation Practices

Effective assessment improves the quality of learning and teaching. It can help students to become more reflective and to have control of their own learning, and it can help teachers to monitor and focus their instructional programs.

Assessment and evaluation of student learning should accommodate the complexity of learning and reflect the complexity of the curriculum. Evaluation should be based on the full range of learning outcomes towards which students have been working during the reporting period, be proportionate to the learning experiences related to each outcome, and focus on patterns of achievement as well as specific achievement.

In reflecting on the effectiveness of their assessment program, teachers should consider the extent to which their practices

- are fair in terms of the student's background or circumstances
- are integrated with learning
- provide opportunities for authentic learning
- focus on what students can do rather than on what they cannot do
- provide students with relevant, supportive feedback that helps them to shape their learning
- describe students' progress toward learning outcomes
- help them to make decisions about revising, supporting, or extending learning experiences
- support learning risk taking
- provide specific information about the processes and strategies students are using
- provide students with diverse and multiple opportunities to demonstrate their achievement
- accommodate multiple responses and a range of tasks and resources
- provide evidence of achievement in which students can take pride
- acknowledge attitudes and values as significant learning outcomes
- encourage students to reflect on their learning and to articulate personal learning plans

- help them to make decisions about teaching strategies, learning experiences and environments, student grouping, and resources
- include students in developing, interpreting, and reporting on assessment

Involving Students in the Assessment Process

When students are aware of the outcomes they are responsible for and the criteria by which their work will be assessed or evaluated, they can make informed decisions about the most effective ways to demonstrate what they know, are able to do, and value.

It is important that students participate actively in the assessment and evaluation of their learning, developing their own criteria and learning to judge a range of qualities in their work. Students should have access to models in the form of scoring criteria, rubrics, and work samples.

As lifelong learners, students assess their own progress, rather than relying on external measures, for example, marks, to tell them how well they are doing. Students who are empowered to assess their own progress are more likely to perceive their learning as its own reward. Rather than asking, What does the teacher want? students need to ask questions such as, What have I learned? What can I do now that I couldn't do before? What do I need to learn next?

Effective assessment practices provide opportunities for students to

- reflect on their progress toward achievement of learning outcomes
- assess and evaluate their learning
- set goals for future learning

Diverse Learning Styles and Needs

Teachers should develop assessment practices that affirm and accommodate students' cultural and linguistic diversity. Teachers should consider patterns of social interaction, diverse learning styles, and the multiple ways oral, written, and visual language are used in different cultures for a range of purposes. Student performance takes place not only in a learning context, but in a social and cultural context as well.

Assessment practices must be fair, equitable, and without bias, providing a range of opportunities for students to demonstrate their learning. Teachers should be flexible in evaluating the learning success of students and seek diverse ways for students to demonstrate their personal best. In inclusive classrooms, students with special needs have opportunities to demonstrate their learning in their own ways, using media that accommodate their needs, and at their own pace.

Using a Variety of Assessment Strategies

When teachers make decisions about what learning to assess and evaluate, how to assess and evaluate, and how to communicate the results, they send clear messages to students and others about what learning they value; for example, teachers can communicate that they value risk taking or lateral thinking by including these elements in determining marks.

Assessment involves the use of a variety of methods to gather information about a wide range of student learning and to develop a valid and reliable snapshot of what students know and are able to do that is clear, comprehensive, and balanced. The assessment process provides information about each student's progress toward achievement of learning outcomes that teachers can use to assign marks, to initiate conversations with students, or to make decisions in planning subsequent learning experiences.

Teachers align evaluation and assessment practices with student-centred learning practices when they

- design assessment and evaluation tasks that help students make judgments about their own learning and performance
- provide assessment and evaluation tasks that allow for a variety of learning styles and preferences
- individualize assessment and evaluation tasks to accommodate specific learning needs
- work with students to describe and clarify what will be assessed and evaluated and how it will be assessed and evaluated
- provide students with regular and specific feedback on their learning

Assessment activities, tasks, and strategies include, for example,

- anecdotal records
- artifacts
- audiotapes
- checklists
- certifications
- conferences
- demonstrations
- dramatizations
- exhibitions
- interviews (structured or informal)
- inventories
- investigations
- learning logs or journals
- media products
- observations (structured or informal)
- peer assessments
- performance tasks
- presentations
- portfolios
- presentations
- projects
- questioning
- questionnaires

- quizzes, tests, examinations
- rating scales
- reviews of performance
- reports
- sorting scales (rubrics)
- self-assessments
- surveys
- videotapes
- work samples
- written assignments

Portfolios

Portfolios engage students in the assessment process and allow them to participate in the evaluation of their learning. Portfolios are most effective when they provide opportunities for students to reflect on and make decisions about their learning. The students and teacher should collaborate to make decisions about the contents of the portfolio and to develop the criteria for evaluating the portfolio.

Portfolios should include
- the guidelines for selection
- the criteria for judging merit
- evidence of student reflection

Portfolio assessment is especially helpful for the student who needs significant support. Teachers should place notes and work samples from informal assessments in the portfolio and use the portfolio to collaborate with the student in identifying strengths and needs, selecting learning experiences, and selecting work that best reflects the student's progress toward achievement of learning outcomes.

It is important that students share their portfolios with other students so that all students may see exemplars that represent a range of strategies for expression and levels of complexity in ideas and understanding.

Outlines and other evidence of planning allow students to examine their progress and demonstrate achievement to teachers, parents, and others.

Students should be encouraged to develop a portfolio that demonstrates their achievements in a context beyond a particular course, including letters, certificates, and photographs, for example, as well as written documents. A portfolio can be very helpful when students need to demonstrate their achievements to potential employers or admission offices of post-secondary institutions.

Tests and Examinations

Traditional tests and examinations are not, by themselves, adequate to assess student learning. The format of tests and examinations can be revised and adapted to reflect key aspects of the curriculum. Some teachers, for example, have designed tests and examinations based on collaborative or small-group learning, projects, or portfolio learning.

Creating opportunities for students to collaborate on a test or examination is an effective practice in the interactive classroom to assess learning of a higher order than recall of information, for example, learning that requires synthesis, analysis, or evaluation.

In learning activities that involve solving a design problem, for example, students might work collaboratively to clarify and define the task and then work either collaboratively or individually to develop a solution. Students might be given a range of questions, issues, or problems and work collaboratively to clarify their understanding of the assignments and plan responses in preparation for the examination for which only one of the questions, issues, or problems will be assigned.

The initial list of questions, issues, or problems can be developed by the teacher, negotiated by the teacher with students, or developed by students and screened by the teacher.

Process-based tests and examinations allow students to demonstrate knowledge and skills and apply strategies at multiple stages in learning processes, for example, in identifying problems, challenges, and opportunities; gathering, evaluating, and synthesizing information; generating options; and developing and evaluating solutions.

Traditional tests and examinations may present a number of problems in scheduling and resource allocation. Process-based tests and examinations may be undertaken in steps during several class periods over a number of days. Students have opportunities to revise, reflect on, and extend their knowledge and understanding. Teachers have opportunities to develop comprehensive assessments, to monitor and evaluate learning at multiple points in a process, and to use time flexibly.

Certification

In some courses, students will need to prepare to demonstrate their learning through entrance tests and examinations or to obtain or upgrade a certification. Replicating this type of assessment in the classroom can help students prepare for the conditions and assessment formats they may encounter in workplace and post-secondary situations.

To make this kind of assessment an effective learning experience, teachers should define a specific context and purpose, for example, the operation of a device, the identification of materials labels, or the demonstration of a technique or procedure.

Appendices

Appendix A: Background and Information Resources

Appendix A consists of the following articles:

- A Strategy for Business Problem Definition
- Teaching Kids Programming: Even Younger Kids Can Learn Java
- Programming the Computer
- Tips for Teaching High School Programming

A Strategy for Business Problem Definition

The purpose of a business requirements capture is to identify the goals and objectives of a project and to document them so that you don't lose sight of them.

In addition to maintaining a list of goals, the development of use cases will further enhance the understanding of the business problem and lead to a better solution.

Requirements

The first step in any business solution is to decide what the requirements are. Requirements are a description of needs or desires for a product. The task of clearly stating requirements is very difficult. Sources may include a request for proposals (RFP), interviews with the company, surveys of users, and various other documentation.

Solving the Business Problem

Answer the "What" questions.

1. What is the problem to solve?
2. What information will it manage?
3. What are the roles of the end users?
4. What technologies are desired for use in the solution?

Do not answer the "Hows" here; keep it high level.

Unified Modeling Language (UML)

Unified Modeling Language is an evolving standard for modelling software systems. UML is a language or notation—it is not a process.

Use Case Analysis

Use cases are an extremely useful way to describe processes. Use cases capture the requirements of the system and are suitable for viewing by the client. Use cases are derived from the business problem statement and additional information collected.

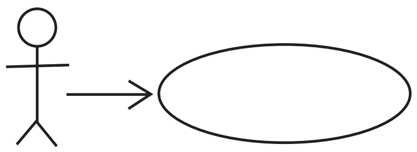
Use Case Terms

Actor: Someone or something that actively interchanges information with the system.

Scenario: An instance of a use case. A use case has one or more scenarios. Each use case has a "Typical Course of Events" scenario, where everything goes as planned, as well as an "Alternative Course of Events," scenario where things don't go as planned. This would include any error handling built into the system.

Use Case Notation

- The stick figure represents the ACTOR.
- The circle is where the name of the use case is entered.
- A line drawn from an actor to a use case indicates that the actor can interact with the use case.



Use Case Example: Processing a Grocery Order

Actor: Cashier

Purpose: To allow a cashier to process grocery items efficiently

Scenario: A customer decides to buy an item. The customer reaches the cashier, where the cashier will scan the item. The system records the bar code so the price can be displayed and totalled.

Typical Course of Events

1. Customer arrives at cashier
2. Cashier scans each item
3. System Response—Price of item displayed
4. Once all items have been scanned the cashier hits the total button
5. System Response—Total displayed

Alternative Scenario

Line 3. No price is available. The cashier manually enters the bar code.

Review

1. Name 3 sources for gathering information.
2. What is an actor?
3. What is a scenario?
4. What do use cases help with?
5. Assignment

Teaching Kids Programming: Even Younger Kids Can Learn Java

By Yakov Fain (used with permission of the author)

Summary

One day my son Dave (10) showed up in my office with my rated "R" Java tutorial in his hands. He asked me to teach him programming so he could create computer games. By that time I've already written a couple of Java books and have taught multiple classes on programming, but all of this was for grownups! A search on Amazon could not offer anything but books for dummies! After spending hours on the Internet, I could only find either some poor attempts to create Java courses for kids, or some reader-rabbit-style books on our friends' computers.

My solution to the problem? I've written my own e-book on it: *Java Programming for Kids, Parents and Grandparents*. Dave became my first kid student and this has helped me a lot to understand the mentality of the little people. This is what I've learned while working on this project:

- Most of the programming tasks require minimal knowledge of arithmetic and algebra skills. To start programming, a kid needs to understand what $x = y + 2$ means. Another important concept to understand is an if statement.
- Kids develop the abstract reasoning abilities by the fourth–fifth grade, and they also easily perform such tasks as browsing the Web, downloading and installing software. Dave has learned how to type, compile and run Java programs in Eclipse IDE in no time.
- Kids learn much faster than adults, but they do not have "previous programming experience," which may actually be a good thing, because they do not have to switch from a procedural to object-oriented way of thinking. After learning about inheritance, Dave called my wife a superclass.
- Adults are responsible creatures, and they can keep doing boring operations much longer than kids. Programming lessons with kids have to be short. One or two 45-minute lessons per week is enough. High school kids should be able to study more, but I do not have such experience yet.
- Some people suggest using Logo as a first programming language for kids, mainly because it has a tool that lets you program with pre-drawn shapes like dogs and turtles. There is an opinion that REALbasic is also good for young programmers. Yes, this language has such concepts as objects, inheritance, casting and overloading, but who uses this language in the real world? Some recommend using Python. The syntax of these languages is not that much easier than Java. There are also plenty of people who write nostalgically about their first Atari computer ...
- Illustrations help. In my book I've included lots of color cartoon-like characters that are like a Java-fabric softener.
- Kids like to see immediate results and enjoy playing with shorter programs, for example a class Fish has a method dive(int howDeep). Let me try to call this method several times with different arguments ...
- Graphical programming is the most fun, and even a fairly large program like Calculator, Tic-Tac-Toe or Ping Pong can be explained to children.
- There is a middle school in Berkeley, CA, where sixth graders start learning Java. This school is equipped with Sun Server and multiple workstations. There is a long waiting list of kids that want to get accepted into this school.

Now I'd like to offer you a fragment from my e-book *Java Programming for Kids, Parents and Grandparents*. I hope you'll agree with me that even younger kids can learn Java.

Java programs consist of classes that represent objects from the real world. Even though people may have different preferences as to how to write programs, most of them agree that it's better to do it in a so-called object-oriented style. This means that good programmers start with deciding which objects have to be included in the program and which Java classes will represent them. Only after this part is done, they start writing Java code.

Classes and Objects

Classes in Java may have *methods* and *attributes*.

Methods define actions that a class can perform.

Attributes describe the class.

Let's create and discuss a class named *VideoGame*. This class may have several methods, which can tell what objects of this class can do: start the game, stop it, save the score, and so on. This class also may have some attributes or properties: price, screen color, number of remote controls and others.

In Java language this class may look like this:

```
class VideoGame {
String color;
int price;
void start () {
}
void stop () {
}
void saveScore(String playerName, int score) {
}
```

Our class *VideoGame* should be similar to other classes that represent video games—all of them have screens of different size and color, all of them perform similar actions, and all of them cost money.

We can be more specific and create another Java class called *GameBoyAdvance*. It also belongs to the family of video games, but has some properties that are specific to the model GameBoy Advance, for example a cartridge type.

```
class GameBoyAdvance {
String cartridgeType;
int screenWidth;
void startGame() {
}
void stopGame() {
}
```

In this example the class GameBoyAdvance defines two attributes—cartridgeType and screenWidth and two methods—startGame() and stopGame(). But these methods can't perform any actions just yet, because they have no Java code between the curly braces.

In addition to the word *class*, you'll have to get used to the new meaning of the word *object*.

The phrase "to create an instance of an object" means to create a copy of this object in the computer's memory according to the definition of its class.

A factory description of the GameBoy Advance relates to an actual game the same way as a Java class relates to its instance in memory. The process of building actual games based on this description in the game factory is similar to the process of creating instances of GameBoy objects in Java.

In many cases, a program can use a Java class only after its instance has been created. Vendors also create thousands of game copies based on the same description. Even though these copies represent the same class, they may have different values in their attributes—some of them are blue, while others are silver, and so on. In other words, a program may create multiple instances of the GameBoyAdvance objects.

Creation of a Pet

Let's design and create a class Pet. First we need to decide what actions our pet will be able to do. How about eat, sleep, and say? We'll program these actions in the methods of the class Pet. We'll also give our pet the following attributes: age, height, weight, and color.

Start with creating a new Java class called Pet in My First Project. Now we are ready to declare attributes and methods in the class Pet. Java classes and methods enclose their bodies in curly braces. Every open curly brace must have a matching closing brace:

```
class Pet{
}
```

To declare variables for class attributes we should pick data types for them. I suggest an int type for the age, float for weight and height, and String for a pet's color.

```
class Pet{
int age;
float weight;
float height;
String color;
}
```

The next step is to add some methods to this class. Before declaring a method you should decide if it should take any arguments and return a value:

The method sleep() will just print a message *Good night, see you tomorrow*—it does not need any arguments and will not return any value.

The same is true for the method eat(). It will print the message *I'm so hungry ... let me have a snack, like nachos!*.

The method say() will also print a message, but the pet will "say" the word or a phrase that we give to it. We'll pass this word to the method say() as a *method argument*. The method will build a phrase using this argument and will return it back to the calling program.

The new version of the class Pet will look like this:

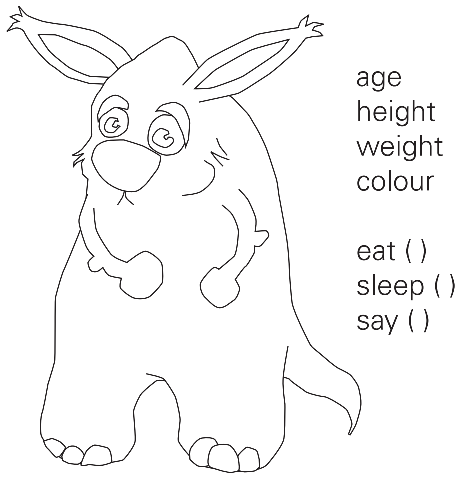
```
public class Pet {
int age;
float weight;
float height;
String color;
}
public void sleep(){
System.out.println("Good night, see you tomorrow");
}

public void eat(){
System.out.println(
"I'm so hungry ... let me have a snack, like nachos!");
}
public String say(String aWord){
String petResponse =  "OK!! OK!! " +aWord;
return petResponse;
}
```

This class represents a friendly creature from the real world:

Let's talk now about the signature of the method sleep():

```
public void sleep()
```



ˇIt tells us that this method can be called from any other Java class (public), it does not return any data (void). The empty parentheses mean that this method does not have any arguments, because it does not need any data from the outside world—it always prints the same text. The signature of the method say() looks like this:

```
public String say(String aWord)
```

This method can also be called from any other Java class, but has to return some text, and this is the meaning of the keyword String in front of the method name. Besides, it expects some text data from outside, hence the argument String aWord.

How do you decide if a method should or should not return a value? If a method performs some data manipulations and has to give the result of these manipulations back to a calling class, it has to return a value.

You may say, that the class Pet does not have any calling class! That's correct, so let's create one called PetMaster. This class will have a method main() containing the code to communicate with the class Pet. Just create yet another class PetMaster, and this time select the option in Eclipse that creates the method main(). Remember, without this method you cannot run this class as a program. Modify the code generated by Eclipse to look like this:

```
public class PetMaster {

public static void main(String[] args) {

String petReaction;

Pet myPet = new Pet();

myPet.eat();
petReaction = myPet.say( Tweet!! Tweet!! );
System.out.println(petReaction);

myPet.sleep();

}
}
```

Do not forget to press *Ctrl-S* to save and compile this class! To run the class PetMaster click on the Eclipse menus *Run, Run...*, New and type the name of the main class: PetMaster. Push the button Run and the program will print the following text:

```
 I'm so hungry ... let me have a snack like nachos
OK   OK   Tweet   Tweet
Good night, see you tomorrow
```

The PetMaster is the calling class, and it starts with creating an instance of the object Pet. It declares a variable myPet and uses the Java operator new:

```
Pet myPet = new Pet();
```

This line declares a variable of the type Pet (that's right, you can treat any classes created by you as new Java data types). Now the variable myPet knows where the Pet instance was created in the computer's memory, and you can use this variable to call any methods from the class Pet, for example:

```
myPet.eat();
```

If a method returns a value, you should call this method in a different way. Declare a variable that has the same type as the return value of the method, and assign it to this variable. Now you can call this method:

```
String petReaction;
petReaction = myPet.say( Tweet!! Tweet!! );
```

At this point the returned value is stored in the variable petReaction and if you want to see what's in there, be my guest:

```
 System.out.println(petReaction);
```

Inheritance—A Fish Is Also a Pet

Our class Pet will help us learn yet another important feature of Java called inheritance. In the real life, every person inherits some features from his or her parents. Similarly, in the Java world you can also create a new class, based on the existing one.

The class Pet has behavior and attributes that are shared by many pets—they eat, sleep, some of them make sounds, their skins have different colors, and so on. On the other hand, pets are different—dogs bark, fish swim and do not make sounds, parakeets talk better than dogs. But all of them eat, sleep, have weight, and height. That's why it's easier to create a class Fish that will inherit some common behaviors and attributes from the class Pet, rather than creating the classes Dog, Parrot, or Fish from scratch every time.

A special keyword extends that will do the trick:

```
 class Fish extends Pet{
}
```

You can say that our Fish is a subclass of the class Pet, and the class Pet is a superclass of the class Fish. In other words, you use the class Pet as a template for creating a class Fish.

Even if you will leave the class Fish as it is now, you can still use every method and attribute inherited from the class Pet. Take a look:

```
Fish myLittleFish = new Fish();
myLittleFish.sleep();
```

Even though we have not declared any methods in the class Fish yet, we are allowed to call the method sleep() from its superclass!

Not all pets can dive, but fish certainly can. Let's add a new method dive() to the class Fish now.

```
public class Fish extends Pet {
int currentDepth=0;
public int dive(int howDeep){
currentDepth=currentDepth + howDeep;
System.out.println( Diving for   + howDeep +  feet  );

System.out.println( I'm at   + currentDepth +  feet below sea level );
return currentDepth;
}
}
```

The method dive() has an argument howDeep that tells the fish how deep it should go. We've also declared a class variable currentDepth that will store and update the current depth every time you call the method dive(). This method returns the current value of the variable currenDepth to the calling class.

Please create another class FishMaster that will look like this:

```
public class FishMaster {

public static void main(String[] args) {

Fish myFish = new Fish();

myFish.dive(2);
myFish.dive(3);

myFish.sleep();
}
}
```

The method main() instantiates the object Fish and calls its method dive() twice with different arguments. After that, it calls the method sleep(). When you run the program FishMaster, it will print the following messages:

```
Diving for 2 feet
I'm at 2 feet below sea level
Diving for 3 feet
I'm at 5 feet below sea level
Good night, see you tomorrow
```

Have you noticed that beside methods defined in the class Fish, the FishMaster also calls methods from its superclass Pet? That's the whole point of inheritance—you do not have to copy and paste code from the class Pet—just use the word extends, and the class Fish can use Pet's methods!

One more thing, even though the method dive() returns the value of currentDepth, our FishMaster does not use it. That's fine, our FishMaster does not need this value, but there may be some other classes that will also use Fish, and they may find it useful. For example, think of a class Fish Traffic Dispatcher that has to know positions of other fish under the sea before allowing diving to avoid traffic accidents :)

Yakov Fain's e-book, *Java Programming for Kids, Parents and Grandparents* can be found at <smartdataprocessing.com/java4kids.htm>.

Programming the Computer

by Professor Gilberto E. Urroz (reproduced with permission)

Programming the computer consists of producing a series of commands that the computer can understand in order to produce a desired action. Typical actions that electronic computers can perform are input and output of data, data processing, and control of interfaces or other devices. At the most basic level, an electronic computer simply interprets voltage pulses as data or commands. From a mathematical point of view, the basic processing of information by the computer consists in the manipulation and interpretation of binary digits or *bits* (i.e., zeros and ones). At this level, thus, commands are represented by strings of zeros and ones.

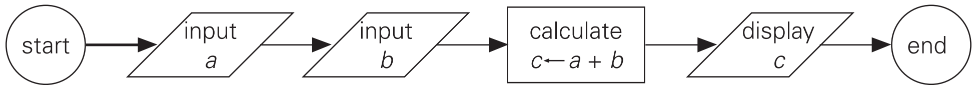
Communicating with the computer at its most basic level takes place through the use of *binary* or *machine language*. Trying to type in these binary commands for computer programming would be overwhelmingly slow and tedious. The use of human-like language significantly facilitates the programming of computers. The next level of computer language is referred to as *assembly* or *assembler language*. It consists of simple commands like ADD, STORE, RECALL, etc., followed by memory addresses within the computer. Although an improvement over machine language, assembly or assembler languages are still quite primitive. High-level languages such as *FORTRAN 90*, *C++*, *C#*, *Java*, *Visual Basic 6.0*, etc., with their human-language-like syntaxes (typically, English) facilitate the programming of the computer for most human programmers. Each one of these high-level languages possesses its own syntaxes or language rules. Violation of the syntaxes of a specific language will result in failure of programming the computer. Programs in high-level languages are typically typed into the computer in the form of text files, and then run through a special program referred to as an *interpreter* or a *compiler*. The interpreter or compiler translates the programs into machine language, the basic language that all computers understand.

Tools for Programming

Typically, to write a program, the programmer starts by selecting or designing an *algorithm*, i.e., a plan for performing the action required from the computer. An algorithm can be simply a series of sequential steps that the computer must perform to produce a result. For example, if we intent to use the computer to add two numbers (a relatively simple operation, mind you), we can describe the corresponding algorithm in words as follows: *input the first number, input the second number, add the two numbers, store the resulting number into a memory location, show the number in the screen*.

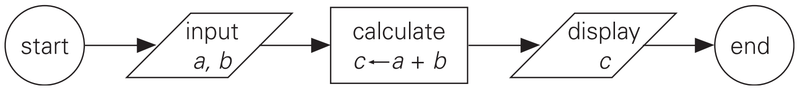
Flowcharts

An algorithm can be presented or described using a flowchart. A flowchart is simply a collection of geometric figures connected by arrows that illustrate the flow of the algorithmic process. The geometric figures contain information in the form of commands or mathematical operations describing the algorithm and the arrows indicate the sequence of steps in the algorithm. The following flowchart describes the algorithm written above for the addition of two numbers.

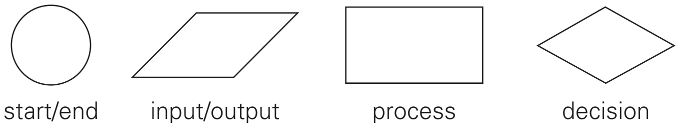


This flowchart is quite detailed for a simple operation.

A more simplified version is shown below:



The basic components of a flowchart are the following geometric shapes, plus the arrows:



In the previous example we used all but the "decision" shape. We will show examples of decision algorithms later in the class.

Coding a Program

The listing of a program written in a high-level language is also referred to as *code*. For example, the code corresponding to the flowchart shown earlier, when written in Visual Basic 6.0, would look like this:

```
Private Sub Add_Click()
Dim a As Single
Dim b As Single
Dim c As Single
a = Val(txtA.text)
b = Val(txtB.text)
c = a + b
picOutput.Print  "c =" , c
End Sub
```

The line Private Sub Add_Click() can be thought of as representing the *start* circle in the flowchart, while the line End Sub can be thought as representing the *end* circle in the

flowchart. The statements *a* = Val(txtA.text)and *b* = Val(txtB.text) represent the input boxes in the flowchart, while the statement picOutput.Print "*c* =", c represents the output box in the flowchart.

The statement *c* = *a* + *b* represents the process box in the flowchart above. The lines that start with Dim are specification statements that are used to identify the variables used in the code. For example, the statement Dim a As Single, for example, identifies variable *a* as a *single-precision* variable. (Single-precision variables are used to store real numbers—i.e., numbers that may have a decimal part—using one word [8 bytes] of memory. To carry more decimals, *double-precision* variables—i.e., those incorporating two words of memory, or 16 bytes—can be used. Integer numbers can be stored in *integer* variables.)

Specification statements need not be included in a flowchart. Furthermore, some high-level programming languages do not require the inclusion of specification statements. For example, the code for the previous flowchart can be written in Java as follows:

```
class AddTwoNumbers
{
public static void main(String args[])
{
System.out.println("This program adds two numbers.");
int x = 5;
double y = 3.5;
double total = x+y
System.out.println("The sum of " + x + " and " + y " is: ");
System.out.println(total)
}
}
```

Pseudocode

While flowcharts are useful guides for describing an algorithm, producing a flowchart can become quite complicated, particularly if done by hand. Also, modification of a hand-made flowchart can be quite involved. Luckily, flowcharting software are now available that can simplify the process. If one doesn't want to get involved in producing a flowchart, one can use another technique known as *pseudocode*. *Pseudocode* simply means writing the algorithm in brief English-like sentences that can be understood by any programmer. For example, a pseudo-code corresponding to the algorithm described in the flowchart shown earlier is presented next:

```
Start
Input a, b
a = b + c
Display c
End
```

Notice the use of the algorithmic sentence $a \leftarrow b + c$, in both the flowchart and the pseudocode, to indicate that the addition of *a* and *b* is to be stored in variable *c*. This algorithmic sentence was translated in the Visual Basic 6.0 code as $c = a + b$ because, in that high-level language, the equal sign represents an assignment operation (i.e., the value $a + b$ is assigned, or stored into, *c*). Since the equal sign (=) represents assignment in most high-level languages, the same is true in Java, where above total is set equal to $x + y$.

Basic Programming Structures

There are three basic programming structures: *sequence*, *decision*, and *loops*.

Sequential Structure

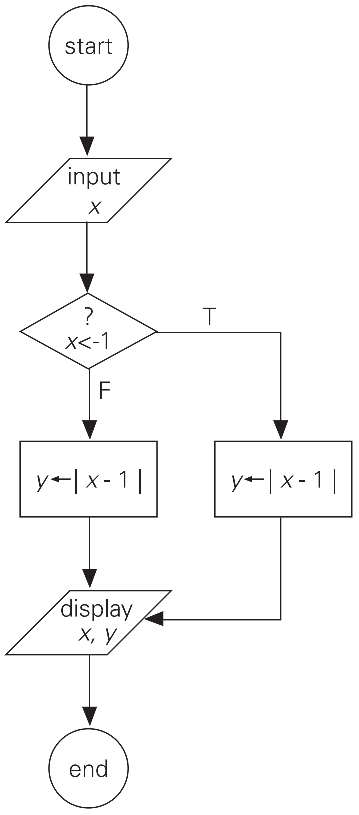
A *sequential structur*e was used in the previous section to illustrate the use of flowcharts. Sequential structures have a single entry point and a single output point, and consist of a number of steps executed one after the other. Sequential structures can be useful in simple operations such as the addition of two numbers as illustrated earlier. The following pseudo-code illustrates a sequential structure consisting of entering a number and evaluating a function given by a single expression:

```
start
request x
calculate y = 3 sin(x)/(sin(x)+cos(x))
display x, y
end
```

Decision Structure

A decision structure provides for an alternative path to the program process flow based on whether a logical statement is true or false. For example, suppose that you want to evaluate the function

$$f(x)=\begin{cases}|x+1|, \text{if } x<-1\\ |x-1|, \text{if } x>-1\end{cases}$$



The flowchart above indicates the process flow of a program that requests a value of x and evaluates y = f(x). The diamond contains the logical statement that needs to be checked to determine which path (T–true, or F–false) to follow. Regardless of which path is followed from the diamond, the control is returned to the display statement. Notice that the input statement and the decision statement form a sequence structure in this flowchart. As in this example, the three types of structures under consideration (sequence, decision, loop) do not appear alone, but two or three are commonly combined in many algorithms.

The algorithm illustrated above can be written in pseudo-code as follows:

```
start
input x
if x<-1 then
y = |x+1|
else
y = |x-1|
display x,y
end
```

The following function represents a possible translation of this pseudo-code into Java code:

```
class Compare
{
public static void main(String Args[])
{
int x,y;

x = toInt(GetConsoleString());

if( x <-1 )
{
y = Math.abs(x+1);

}
else
{
y = Math.abs(x-1);

}
System.out.println( x =   + x  +   ,   +   y =  + y);
}// end of main method
} // end of class Compare
```

A possible translation into Visual Basic 6.0 is shown next:

```
Private Sub f00_click()
Dim x As Double, y As Double
x = Input("Enter x:")
If x<-1 Then
y = abs(x+1)
Else
y = abs(x-1)
End If
MsgBox( x =   & x &  , y =  , y)
End Sub
```

The decision structure shown above is such that an action is taken whether the condition tested is true or false. In some cases, if the condition tested is false, no action is taken.

Programming the Computer was written by Professor Gilberto E. Urroz, Utah State University, and has been adapted to include Java examples. (Used and adapted with permission.)

Tips for Teaching High School Programming

- Have students create a CP12 folder for all course work.
- Have students save often and keep multiple versions of files.
- If possible have network tools or software installed to allow for remote viewing and remote control of student machines.
- Always have students model and pseudocode all programs before they begin writing code.
- Stress the importance of following proper coding conventions such as giving descriptive names to objects, classes, and variables.

- Stress the need for a programmer to be patient and that attention to detail is absolutely crucial in writing programs.
- Show students how to debug their own programs so that they can answer many of their own questions.
- You can place Styrofoam cups painted red on one side and green on the other and have students turn the cup red side out when they have a question; that way you are not facing a sea of waving arms or students calling out for help from all over the room.
- Follow good coding conventions yourself in any examples you share with students.

Appendix B: Group Work and Team Building

Establishing Team Ground Rules

The following activities are designed to help students work together as effective teams. When a group of people gets together to work on a task, they often carry with them certain expectations about how effective teams work. It is important that all group members feel part of the team and that their input and skills are valued. It is also important that all teams discuss their expectations for a project so that individuals have a sense of what is expected of them. One step in the process is establishing team ground rules that all team members agree to abide by.

Activity 1: Discussing Teamwork

Take five minutes to read the following statements, and for each statement, identify whether you believe it to be true or false. The purpose of this activity is to generate discussion about teamwork.

- Team members should always express what they are thinking and feeling, even when it might offend other team members.
- Teams need someone in charge in order to get work done.
- A team has reached consensus when almost everyone agrees with a decision that has been suggested.
- Teams always produce better work than individuals.
- In order to be effective, team members should have a personal liking for each other.
- Effective teams try to reach consensus on all issues.
- It is better to avoid discussing interpersonal disputes with team members—it always leads to more trouble.
- Teams are only as strong as their weakest member.
- If you don’t have anything nice to say, you shouldn’t say anything at all
- Being able to work on a team is a natural ability—you’re either born with it or you’re not.

Activity 2: Setting Ground Rules

As a group, you have decided which statements you believe to be true and which ones are false.

Make a list of ground rules that all members of your group agree to follow.

Activity 3: Identifying Helpful Team Behaviour

This activity is designed to help you understand that different people may see the same problem in different ways. It is important for teams to decide on a decision-making process and also to identify helpful and harmful team behaviours.

Read “Justice Deserved?” individually and answer the questions that follow.

Justice Deserved?

by Sean Bennett (reprinted with permission)

Rain, rain and more rain was the order of the day, making everyone cranky and disgusted. It had been a tough winter, with major blizzards, power outages and shovelling that had never been matched. Now the fleeting snow was being replaced with inches of rain, coming down in sheets.

Harley awoke late, having hit his snooze button three times. He rocketed out of bed, bypassed the shower, threw on his clothes and jacket and jumped out into the rain—without an umbrella. The bus stop was two blocks away and he needed to get there within a minute or so. Now in full sprint, water splashing up his legs, Harley could see the bus stop in the near distance, with the bus making its way in the opposite direction. Picking up his pace, Harley saw an old woman, hunched over and moving as quickly as a tortoise with a baby grand on its back, lumbering along the middle of the sidewalk. With the bus approaching the stop, his heart now pounding at a feverish pace, Harley made a split-second decision, buried his right shoulder into the lady's back, never missing a step. The old woman flew as if shot from a cannon, landing a good ten feet away, crumpled up against the side of the dry cleaner's front door.

"You idiot! Police, police! He just tried to kill me!"

Not missing a beat, Harley made it to the bus stop, never looking back to see if the woman was okay—the bus still about five hundred feet away.

At the same time, young Billy was walking towards school, having missed the school bus for the same reason as Harley—having slept in. He had been up all night online gaming, and was now paying for this by having to walk to school in the rain. Angry and soaked, Billy picked up a rock and skipped it atop a large puddle that had accumulated near his family's apartment. He biffed a few more stones as he ambled along, rain pounding off his rain jacket and Yankees ball hat. As he approached a bus stop along his route to school, he let a rock fly that took an unusually high skip and made its way onto the sidewalk, hitting an old woman who was slowly gathering herself from the pavement. Billy thought she must be a homeless person—just getting up from her spot in front of the dry cleaners. The rock caught her square in the forehead, sending her spilling onto her backside. She seemed stunned, and rightly so, but she also seemed fine, considering a rock had just been zinged and had smashed itself firmly between her eyes. She started hollering something about the police, but Billy was not going to stick around to see the conclusion to this, so he took off running past the bus stop, snickering to himself.

The driver of the Route 27 bus was a younger man, not overly happy with his spot in life. No one ever cared about his problems, but they'd always shower their misfortunes onto him as he drove them to their destinations. Who cares if you have to go to the doctors? Who cares if your cat has been missing a week? Certainly not the driver of bus number 27.

He was running a couple of minutes behind schedule on account of the rain being so heavy, and because of this the people getting onto the bus were giving him an unusually heavy amount of griping. He didn't care. He needed a laugh. He needed something to charge his day. Then, for reasons beyond his control, he drove his bus into a large puddle that had formed about a block from his next stop. The water flew onto the sidewalk like a tidal wave, dousing three poor, unsuspecting souls: a kid and man both running along the sidewalk and an old lady who was sitting on her behind in front of the dry cleaners. The bus driver let out a large and robust laugh, the people sitting on the bus sitting in disbelief at what they had just witnessed.

"Police! Police! Help me!"

Officer Jones grumbled as he turned from the counter of the coffee shop, strawberry filling from his doughnut running down his chin. An older woman was lying on the sidewalk while a man in his 20s ran past her quickly. Wiping the filling off his chin, Jones decided to "pretend" he hadn't heard the woman's cries for help. Not even a couple of minutes later, however, the same screams were heard; this time the woman was on her backside, rubbing her forehead.

"Hmm," grunted Jones, sipping on his hot coffee, ordering another doughnut. Within seconds, the sound of the bus roared past the shop. As Officer Jones turned he saw the tidal wave smash onto the pedestrians on the sidewalk, the old lady once again an unwilling recipient of being in the wrong place at the wrong time.

The officer grudgingly decided that he could not sit idly by anymore, his coffee cup now empty of its liquid. Sticking his baton into its holster, rain gear on, he headed out into the elements to help the old lady, now rolling about, drenched, sore and angry. The police officer approached the lady and assessed her plight.

"Seems like you're having a rough one, lady …"

The lady did not find humour in her situation. Her body was soaked, a welt the size of an orange protruded from her forehead and she felt as if she had a Mack truck stuck between her shoulder blades. To top it all off, her voice was going from all the screaming she had let fly, all the while Officer Jones enjoyed his doughnut and fresh coffee.

"Listen here, I have a right to smack you one over the head. Where were you when I needed ya? Probably sitting in that coffee shop over there with strawberry jam filling running down your chin!"

Jones shuffled uncomfortably at the lady's accuracy, yet was also getting more than a little tired of her ranting. "Look, do you need some help? I'm busy here, you know." Officer Jones leaned over the lady and placed his hand on her left arm.

"Why you …" was all he heard, as a massive handbag, certainly the lady's equal in terms of weight, came crashing down on his head.

Being a large man and with pride that would not allow an old woman to do him harm, Jones stumbled up quickly and pushed the lady, now standing, up against the front window of the dry cleaners. Jones placed handcuffs on the elderly woman and called in back-up. "Lady, you're going for a ride … "As the old lady was placed in the police car, the young man stepped onto the bus, wet yet none the worse for wear. Billy trotted into first period mathematics, a little late yet also fine. And the bus driver whistled behind the wheel of the bus, oblivious to the passengers' complaints about his tardiness, reliving in his head the "perfect puddle job" that he had accomplished minutes earlier.

Activity 4: Rank the Characters in Order of Offensiveness.

- Who, in your opinion, is the most offensive person in the story? Rank this person as #1 and record the name in the space below. Write a brief (two- or three-word) reason for your choice.
- Who, in your opinion, is the second most offensive person in the story? Rank this person as #2 and record the name in the space below. Write a brief (two- or three-word) reason for your choice.
- Who, in your opinion, is the third most offensive person in the story? Rank this person as #3 and record the name in the space below. Write a brief (two- or three-word) reason for your choice.
- Who, in your opinion, is the fourth most offensive person in the story? Rank this person as #4 and record the name in the space below. Write a brief (two- or three-word) reason for your choice.
- Who, in your opinion, is the fifth most offensive person in the story? Rank this person as #5 and record the name in the space below. Write a brief (two- or three-word) reason for your choice.

Activity 5: Agree

Appoint an observer—someone who is prepared to not participate in the following discussion but simply to observe silently how your team works together in the next task. The observer is to make note of any behaviour that might be helpful or not helpful to team discussions..

Once you have appointed your observer, as a group, discuss your individual rankings. You and your team must reach agreement about who deserves to be #1, 2, 3, 4, and 5.

Activity 5 Report

Take some time to listen to what your observer has to say regarding how the group handled the task of ranking the characters. Feel free to offer your own observations after you have heard his or her report. As a group decide on a decision-making process for your team and make any additions or deletions to your team ground rules that you feel are needed.

Activity 6: Create a Team Identity Document

Treat your team as a software development company. Decide on a team/company name. As part of developing a team identity, your team should decide on a logo.

Team Building and Delegation of Team Roles

Make a list of roles and responsibilities that are required to make your team work efficiently. Decide who will fill the various roles. How long will each team member keep his/her title/role? Discuss the skills each team member brings to the collective and try to put people into roles where they can best help the team.

Team Ground Rules

Include your team ground rules that you have already decided on.ˇCreating the Document

Compile your team identity, delegation of team roles, and ground rules into a document and hand it in to your teacher.

Activity 7: (Optional) Create a Team Web Page

The team identity document you created in Activity 6 should be web-enabled.

Your web page should include:

- Team Name
- Logo
- Team Roles
- Team Ground Rules

You may also want to include team member profiles as well as links to team member home pages.

Your web page is a form of advertisement for your team. Be creative. Keep information clear and concise.

Appendix C: Sample Learning Activities

Quiz: Binary, Hexadecimal, and Decimal Numbers

Name: ____

Instructions:

Perform the conversions required to complete the following table.

Binary	Hexadecimal	Decimal
1100110011		
	3A	
		55
10110111		
	4A3	
		28

Binary Addition:

```
1011+        011001+       10110011+

1010         111101        11001100

_____        ______        _________
```

TOTALS:

Logic Gate Problems

Super Bowl

One of the more interesting public works problems is the "Super Bowl" problem.

At the beginning of halftime during the Super Bowl, 35 million toilets are flushed almost simultaneously. The resulting loss of water pressure wreaks havoc on many municipal water systems.

Here you will solve the problem for a "three toilet" system. Devise a logic circuit whose "1" inputs represent "flushes" and whose "1" outputs represent opened water-feed valves. If no more than one toilet is flushed, then that toilet's water valve opens, the others remaining closed. If more than one toilet is flushed, then all the water valves remain closed.

Robots

You are designing a robot to move toward a light source.

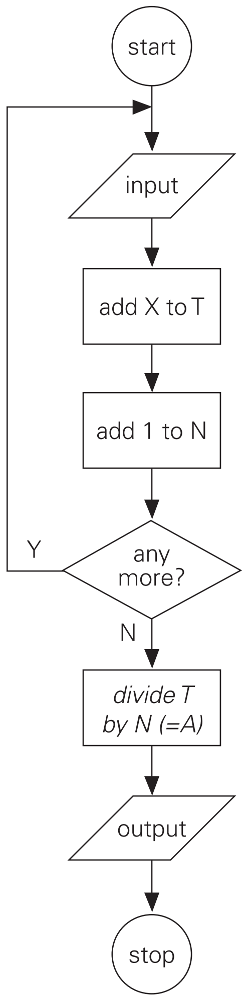
Three photosensors SL, SC, and SR are mounted at the front of the robot pointing 45o to the left, straight ahead, and 45o to the right, respectively. Two wheels WL and WR are powered depending on the output of the sensors. If SL detects light, the robot is pointing too far to the right, and the right wheel WR must be powered up to turn the robot to the left. The opposite is necessary if SR receives light. If only the forward-pointing sensor SC is lit, then both wheels WL and WR should be powered to propel the robot forward.

If one treats the sensors as having binary outputs, i.e., either "on" or "off," and the powered wheels as being "on" or "off," a simple logic circuit can be used to actuate the wheels under each sensor condition. Create such a logic circuit using only NAND gates, and using the least number of these.

Flow Charts

Flow Chart Problem

Explain what is happening in this flow chart.



Flow Charts

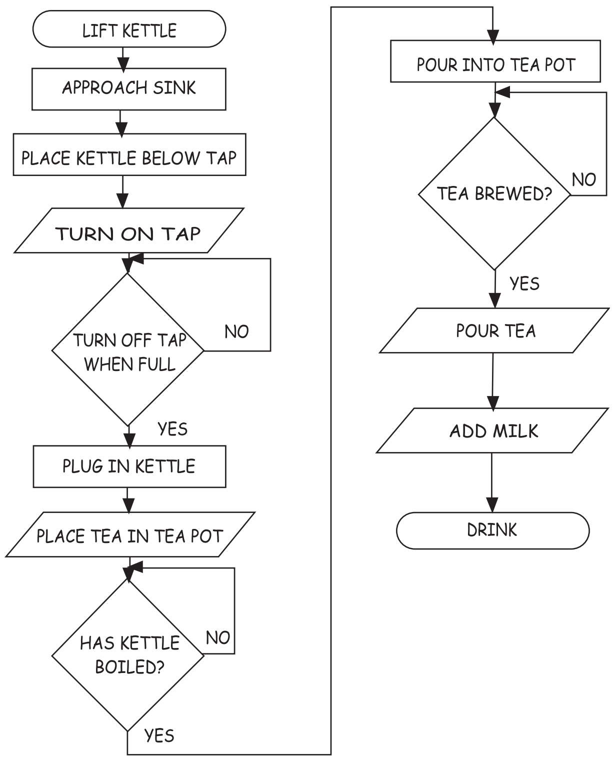
(Reproduced by permission of V. Ryan of the World Association of Technology Teachers)

Flow charts are used to help programmers in the early stages of programming. A flow chart is a chart that flows from one stage to the next and it will show what stage or event is first, second, third, etc ...They are very useful when programming because they allow the programmer to set out, in a very simple way, the sequence that he/she wants for each line of the program. (See the example below.)

The First Stage of Programming

This is a flow chart representing the making of tea. It starts with filling the kettle with water all the way through every possible stage. Imagine a robot had to be programmed to perform this basic task. The programmer would have to give the robot every instruction. Remember—computers will only do what we instruct then to do. They cannot decide anything for themselves.

If you examine the flow chart you will see that every stage of the tea-making process is identified. However, you may be able to add even more detail. If you were to write a program for all the stages involved, at least one line would be needed for each stage. The shape of each box is important. For instance, a diamond represents a decision, a rectangle is an ordinary process box, a parallelogram is an input or output, and the start and end boxes also have their own shape. A decision box has two possible outcomes—YES or NO. In the example shown, a NO means that the flow 'loops' backwards.



Assignment

Draw a flow chart to represent every stage of getting up in the morning and coming to school. Include the full range of boxes. The best way to start this question is to carefully list each stage.

The above activity is reproduced with permission from the technology student website, <technologystudent.com> and was written by V. Ryan of the World Association of Technology Teachers.

Activity: Introduction to Object-Oriented Programming

Describing an Object

In object-oriented programming everything is an object. Every object has properties and methods. Properties are like adjectives that describe the object. Methods are like verbs; they are the things an object is able to do.

A real-world example of an object is a blue bird. A blue bird has properties and methods, and these can be shown in an object diagram as shown below.

Blue Bird
colour weight feathers wings feet beak
fly walk chirp eat drink

The first row is used to give the object's name. The second row is the object's properties, and the third section gives the object's methods.

Students can use the above format to create object diagrams for real-world objects like cars or planes or create object diagrams for programming specific objects like command buttons or text fields. This introduces the concept of object-oriented analysis and serves as practice for describing an object and its attributes.

Object Properties Methods

Computer Science in the Modern World

(Used with permission from the Association for Computing Machinery (ACM) K–12 CS Model Curriculum document 2005.)

Activity

Number Systems

Time

4 hours

Description

Students develop an understanding of the relationship between the binary number system and computer logic. Also, students learn how to convert base 10 numbers into binary and vice versa. Character representation of binary codes is explored. Students have the opportunity to experiment in writing their own message and decoding.

Topics

The connection between elements of mathematics and computer science, including binary numbers, logic, sets, and functions

Prior Knowledge

Understanding of the decimal number system and place value

Planning Notes

- Review how programming software handles character representations.
- Have eight pennies for each pair of students and either a handout and/or overhead of bit information.
- Review binary and base 10 conversions.
- Prepare coded messages for the students to decipher.
- Have copies of ASCII code available (both standard and extended).

Teaching/Learning Strategies

- Show segment 3 of The Journey Inside video (8 min., 25 sec., Intel Corporation, part of The Journey Inside education kit available on the Internet at Intel/education), or any other video that shows how computers turn pictures and colours into codes. Students gain an understanding of how information is communicated through the use of codes.
- Hand each pair of students eight pennies and work through the questions on bit information. Ask students what pattern they can see forming in the right column (numbers double).
- Students are challenged to count as high as they can on one hand and told the answer is greater than _______.

- While students ponder the challenge, teachers demonstrate, with the aid of a simple series circuit, the binary logic states of ONE and ZERO (TRUE and FALSE, HIGH and LOW) by equating them to series circuit lamp ON and OFF condition.
- Binary numbers are introduced by initiating finger counting on one hand—no fingers up is 0, thumb up is a 1, index finger up is 2, middle finger up is a 4, ring finger is 8, and pinkie finger represents 16. Students demonstrate counting to 31 on one hand.
- This sets the stage for demonstrating how to convert numbers from base 10 to base 2 (binary). Work through several examples with students.
- Give students a quiz on binary conversion to assess their grasp of the concept.
- Hand out the ASCII conversion information. Since computers cannot think like we do, they need a code to translate our language into data that they can process and then convert that data back into recognizable language.
- Students complete conversion exercises.

Assessment/Evaluation Techniques

- Formative assessment of quiz at the end of the binary conversion exercise to prompt students on progress and show changes required for success of conversion application
- Summative assessment of conversion exercises

Accommodations

- Use extensive visual aids and demonstrations to assist students as needed.
- Provide an enlarged copy of conversion methodology in classroom as well as ASCII character chart. Use a variety of teaching styles to accommodate learning styles.
- Provide appropriate adaptive devices or implementation accommodations for identified students.

Resources

- Adapted from the course profile for Computer Engineering Technology, Grade 10, Unit 2: Integrated Circuits (page 53), Ontario Ministry of Education.

Careers in Computer Engineering

(Used with permission from the Association for Computing Machinery (ACM) K-12 CS Model Curriculum document 2005.)

Time

3 3/4 hours

Description

A guest speaker is invited to share information about his or her job/career with students. Students expand on their computer-industry knowledge. Students look at degrees and certifications available and opportunities they have at the high school level and beyond to move toward careers in the computer industry.

Topic

Students will gain a conceptual understanding of the identification of different careers in computing and their connection with the subjects studied in this course (e.g., information technology specialist, web page designer, systems analyst, programmer, CIO).

Prior Knowledge

Word-processing skills

Planning Notes

- Guest speakers may include the school sysop, board technician, or someone from the local community.
- Collect information from a local university or community college, including school course calendars and college/university catalogues.
- Gather copies of recent computer trade magazines.
- Arrange ahead of time for a student to introduce the guest speaker and another student to thank him or her.
- Collect newspaper advertisements for jobs in the computer industry.
- Distribute a sample certification worksheet

Teaching/Learning Strategies

- Teachers introduce the expectations of the activity.
- Teachers review with students (ahead of time) questioning techniques for the guest speaker.
- One student may introduce the guest speaker. Students take brief notes in order to ask relevant and interesting questions. One student may thank the guest speaker.
- Discuss the speaker information with the students, after which they write their personal views on the information.
- Students look through trade magazines to see advances in the computer industry. Each student picks one article from a magazine to summarize or review using a word processor.
- Finally, students look at opportunities for different computer designations ranging from MCSE (MicroSoft Certified Systems Engineer) to computer engineering at the university level. Students use newspaper advertisements to explore what skills and designations are requested by potential employers.

- Students retrieve the certification chart file (either electronically or via handout) and, using designations discovered in the advertisement exercise above, they complete the chart and add it to their portfolios.
- Students create a plan on how to pursue a computer career, beginning with the completion of this course, and save the information in their portfolios (long-term goal).

Assessment and Evaluation

- Review student portfolios to provide written/oral feedback on completion and comprehension of tasks given.
- Evaluate the article review using the rubric provided.

Accommodations

- Allow flexible time lines for due date of report.
- Use career centre videos if available.
- Videotape the guest speaker(s) presentation to allow students an opportunity to watch it again.

Resources

- Course Profile: Computer Engineering Technology, Grade 10, Unit 3: Networking (page 93), Ontario Ministry of Education.

Computer Science as Analysis and Design

(Used with permission from the Association for Computing Machinery (ACM) K–12 CS Model Curriculum document 2005.)

Activity

New Solutions for Old Problems

Time

5 hours

Description

Students examine problems that can be solved using more than one algorithm (e.g., determining the factorial value of a number). Using brainstorming or other group problem-solving techniques, students develop alternative algorithms using recursive and non-recursive techniques. Students identify the components of a recursive algorithm and develop criteria for recognizing when a recursive algorithm may be applied.

Topics

1. Fundamental ideas about the process of program design and problem solving, including style, abstraction, and initial discussions of correctness and efficiency as part of the software design process
2. Simple data structures and their uses

Prior Knowledge and Skills

- Use of problem-solving models, the ability to develop appropriate algorithms to solve problems, and the ability to write pseudocode

Planning Notes

- Review the nature of recursion.
- Gather examples of problems that can be solved using more than one method, including recursion, and determine which problems may be solved using a recursive algorithm.

Teaching/Learning Strategies

- Divide the class into groups of two or three students.
- Review the brainstorming problem-solving technique.
- Present a problem that can be solved using a familiar but complex algorithm and may also be solved using a less familiar but simpler algorithm (e.g., determining the quotient and remainder of the division of two integers).
- Students, in their groups, develop more than one algorithm for the solution.
- Facilitate a class discussion to develop criteria for the evaluation of algorithms, including the efficiency of the solution and the complexity of the required coding. Both processing and user interface efficiencies are considered.
- Groups evaluate the algorithms using the developed criteria and share their algorithms and evaluations with the class.

- Introduce the recursive method of problem solving and illustrate a recursive algorithm for the solution to a different problem (e.g., calculating the factorial value of a number).
- Groups develop a recursive algorithm to the initial problem and evaluate its efficiency.
- Facilitate a class discussion to establish criteria for determining if a recursive algorithm is an appropriate solution and identify additional problems that may be solved using recursion.
- Working in groups, students develop recursive and non-recursive algorithms for additional, assigned problems.

Assessment and Evaluation

- A formative assessment of the assigned in-class work in the form of roving conferences
- A summative assessment in which students complete an assignment requiring the development of both a recursive and a non-recursive algorithm

Accommodations

- Provide print copies of examples of algorithms using recursive and non-recursive methods, including graphic illustrations, and use models to illustrate the algorithms.

Planning a Solution

(Used with permission from the Association for Computing Machinery (ACM) K–12 CS Model Curriculum document 2005.)

Time

6 hours

Description

Students work in groups to analyse complex problems (e.g., Towers of Hanoi) and to develop appropriate algorithms using recursive and non-recursive techniques. Students create pseudocode and design charts to assist them in planning a solution and assess these representations of code as problem-solving tools.

Topics

1. Fundamental ideas about the process of program design and problem solving, including style, abstraction, and initial discussions of correctness and efficiency as part of the software design process
2. Principles of software engineering: software projects, teams, the software life cycle

Prior Knowledge and Skills

- Students can apply the steps in the software design life cycle; use pseudocode, diagrams, and charts to summarize program design; and develop appropriate algorithms to solve problems.

Planning Notes

- Review top-down problem solving.
- Select a problem to use in developing a model solution and prepare the appropriate models.

Teaching and Learning Strategies

- The class is divided into groups of two or three students, and each group is assigned a problem.
- Groups investigate the problem, using a variety of problem-solving techniques to analyse it.
- Each group uses brainstorming or other group problem-solving techniques to develop an algorithm for the solution to the problem. In a class discussion, groups present and share their algorithms.
- Students compare the effectiveness and efficiency of the algorithms presented, and then the groups refine their algorithms.
- Each group develops a flow chart, structure chart, and/or pseudocode to represent the application of the algorithm. The teacher conferences with each group to discuss and assess the solution design.

Assessment and Evaluation

- A formative peer assessment of the presented algorithms
- A formative assessment of the design for the solution to the problem

Accommodations

- Provide print sample algorithms similar to the one studied.
- Use graphical models to illustrate the problem.
- Selectively pair/group students to assist problem solving.
- Provide problems of varying complexity to provide an appropriate challenge.

CP 12 Project (Module 3)

Your task is to design and create a piece of store management software. The type of store or business is up to you; your application must include the following features and capabilities:

Inventory listing that can be updated by the user using an array

- user validation
- background image, colour, graphics, and sound
- multiple forms, menu
- help file
- documentation—diagram, pseudo-code, list of objects and names
- ability to generate totals and running totals, show tax
- list boxes
- combo boxes
- option buttons
- check boxes
- message boxes for errors
- input boxes
- labels
- text boxes
- calculation of a monthly payment on a purchase
- decision statements
- looping structures
- formatting
- import/export files
  - password protection

Marks will be based on

- functionality
- ease of use
- design of your application

Students may work in pairs or individually. It is expected that students will need to research a number of the mandatory features in order to complete this project. Planning will be very important in creating a quality project, and considerable value will be given for proper documentation.

E-Business Project Overview (Module 4)

This is an example of the type of project students could complete where the teacher assumes the role of a fictional client. This example could be connected to a relational database to give it real functionality, or depending on the skills and abilities of the students involved, it could be left as a stand-alone application where orders are generated by the software but not stored or processed any further.

Background

As a member of a software development team, you have been asked by CompuMatch, a Canadian computer system company, to develop their point of sale software. The software will contain a home screen and other elements you have discovered through your analysis. You will develop use case diagrams and use cases to model and understand the requirements of the software program. The graphical interface must implement formatting and incorporate good design principles.

Objectives

- Conduct interviews with the company to complete a business requirement analysis.
- Demonstrate your ability to create and implement use cases and a use case diagram to capture the functionality of the program.
- Use a programming language to develop and manage a software project. Implement good design principles

Problem Definition/Business Narrative

CompuMatch is a computer sales company servicing all of Canada. It sells and leases computers, building custom machines for customers. Currently, customers contact the company via a toll-free number, where the company collects client information, including the client's name, address, whether the client is a new or repeat customer, the brand name, processor type, processor speed, RAM, hard drive size, and accessories required, as well as the cost of the computer and whether the client wants to buy or lease. Right now, this is all done by hand using pen and paper.

CompuMatch wants a software program to make its operations more effective. The program will allow employees to enter all the information on what kind of computer system the client is looking for by allowing employees to select the components and features through lists, check boxes, option buttons and other input devices. The software will keep an inventory of components and prices that is updateable through the software program. The program should calculate a running total and also calculate monthly payments and allow for special discounts. The interface should be attractive and easy to use and have a common look and feel throughout. Each form should display the CompuMatch logo, and the program must be able to print a completed work order, which includes customer information, payment information, and system requirements.

Client Acceptance Factors

These are the basic requirements of the CompuMatch software project:

- Is a user-friendly interface with consistent design used?
- Are there multiple fonts, text boxes, drop-down menus, push buttons?
- Have background images been used?
- Has the corporate logo been displayed?
- Have use cases been developed and implemented?

- Have use case diagrams been developed?
- Does the form include whether the client is a new or repeat customer, the brand name, processor type, processor speed, and hard drive size, as well as the cost of the computer and whether the client wants to buy or lease?
- Is a monthly payment calculated?
- Is a running total kept?
- Can the user clear all or part of the form?
- Will the system print out a complete work order?
- Is the inventory list updateable?
- Is the program bug free?
- Is the program scalable?
- Have proper naming, coding, and design conventions been followed?
- Are prices formatted, and is data input validated?

Appendix D: Module 4 Project Example

Background

This case study was created by a group of Dalhousie University Computer Science students and illustrates the documentation, scope, and project planning expected in completing the project outlined in Module 4. The content of the project is not based on the outcomes of this curriculum, but this project serves as an excellent guide on how student projects should be produced.

Explanation of Case Study

The objectives of the following document are
- to outline the project that was the basis for this case study
- to explain the steps required to successfully complete a computer project for a community organization
- to relate these steps to the Computer Programming 12 (CP 12) curriculum (see Table 1)

The Project

The need to encourage physical activity, especially in youth, is becoming more and more apparent. With obesity rates at record levels, people must be educated about making healthy lifestyle choices and including physical activity in their daily lives. At Millwood High School, the administration is taking action against sedentary habits by introducing the 10000 Steps-A-Day program. This program, principally aimed at inactive individuals, encourages people to walk more by providing pedometers so participants can measure their progress. With a grant from the Province of Nova Scotia, Millwood High School plans to distribute pedometers to interested students and teachers in an effort to promote physically active lifestyles in their community.

To be successful, the 10000 Steps-A-Day program must motivate participants to choose to walk. Individually, participants are motivated by the pedometers, which give a quantitative measurement of their achievements over the course of a day. Quantitative measures allow participants to set specific goals and evaluate their performance. To sustain motivation for long periods, additional tactics can be employed. Graphical displays can be provided that show how individual participants have increased their physical activity over time. Other statistics that may have more meaning to participants than the number of steps (such as calories burned and equivalent food items or distance travelled) can also be calculated and displayed to give participants a more concrete impression of their accomplishments. Within the context of a high school, it is possible to take advantage of "school spirit" sentiments by providing data on how far the school as a whole has walked. In order to use these ideas to motivate participants, there must be a system in place to record and analyse the number of steps taken by each participant, each day, over the course of the program.

To this end, a web-accessible database was developed for 10000-Steps-A-Day participants to input the number of steps they take each day. Participants are able to log in to this database and input their steps using a calendar-like interface. The participants can enter their steps for just the current day or go back and enter steps for any days they might have missed. When first registering to use the website, each participant fills out a user profile containing personal information (date of birth, height, weight, sex, stride length, pedometer correction factor, etc.). This information is used to calculate distance travelled and calories burned. Distance travelled and calories burned will be inserted into motivational material for participants. For example, the website might display "Since registering in the program, you have walked the equivalent of the distance from Sackville, NS, to Ottawa, ON."

Because the 10000 Step-A-Day program is grant-funded by the Province of Nova Scotia, tracking the outcome of participants in the program is important to Millwood High School. A survey was implemented in consultation with Millwood High School to assess the outcomes of the program. The website is programmed to prompt the user to fill out the survey at regular intervals, so that Millwood High School can better chart the progress of the participants. Information on participant outcome will be reported to the Province of Nova Scotia so that the program can be evaluated.

A group of four fourth-year university students completed this project in approximately two months, although the primary focus of two of the four students was development of this case study rather than contributing to the database/website. The main contact person for Millwood High School was the vice-principal, who is referred to as the client in this document.

Technical Information

The website was developed using Hypertext Preprocessor (PHP). PHP is often used as server side scripting language. It is an open source programming language for web development and is designed to be embedded into HTML. PHP is compatible with many different platforms and is easy to learn. PHP borrows much of its syntax from C, Java, and Perl. It is an effective tool for creating dynamic web pages quickly. In addition, it is ideal for websites that will be updated and administrated by a non-technical person, as PHP can be used to load the content of the website from text files that can be easily edited. A downside to PHP is that it must be run on a web server with PHP installed. Thus, testing requires uploading to a web sever to test every change. This can be troublesome if bandwidth is scarce or the PHP project is large.

The database was developed using MySQL. MySQL is a relational database management system. A database is an organized collection of information that a computer uses to select and display data. MySQL is a powerful tool, but it is not suitable for very large and complex databases.

In accordance with the requests of Millwood High School, the web interface to the database was designed with the same general layout as the alumni section of the Millwood High School website <http://www.millwood.ednet.ns.ca/alumninew/index.htm> in terms of having an uncluttered design that is relatively simple and straightforward. In addition, Millwood High School also required the ability to edit the content of the website with little knowledge of HTML or PHP. The website was designed so that the content could be accessed and changed through text files that were loaded by the PHP files. The added benefit of loading text files is that the actual formatting of the work is liquid—that is, flexible to adapt to any user's personal browser settings (resolution, font size, etc.).

The client can access empirical data through the website's administrative software, PHPMyAdmin. The database was designed so that stored data can be readily converted to an Excel spreadsheet when it is exported.

The Steps

The following section gives a brief introduction to the steps involved in completing a computer project for a community group. It also highlights how each of the steps relates to the most recent version of the CP 12 curriculum.

Identifying Potential Clients

- Find candidate groups within your school and your community. For example, teams and clubs have a need for websites to communicate with team members or find like-minded clubs from other parts of the world. Likewise, fundraising committees might benefit from a database. At first, recruiting potential clients from the community at large may be difficult, as you will have no reputation for completed projects. However, soon you will have finished products that will showcase the abilities of your students and attract community groups.

- You may also wish for students to identify unique problems that could benefit from technology. For example, a student may wish to start a battery recycling program in which members of the community would post their address on a website. In these cases, there is no specific "client," however, a teacher themself can act in this capacity by evaluating proposals/etc.
- Students might also have their own ideas in terms of what organizations would benefit from a computer project.

Divide Students into Groups

- In order to attract clients, it is important that the groups of students are productive. Minimizing internal conflicts by distributing skills appropriately and accounting for personality will facilitate a healthy and more productive work environment. It is a good idea to have the students fill out a form indicating what computer and group work/leadership skills they have, what people in the class they would be comfortable working with, and what projects interest them the most. A template form is attached that you can modify for this purpose.

Defining Client Requirements

- Once clients have been recruited and the groups have been assigned, the next step is to define the client's requirements. It is important for you to attend these first meetings with every group, as they will set the tone and workload for the remainder of the project. It is ultimately your responsibility that the scopes of the projects are both feasible and sufficient for credit in your class. Have the students draft the client requirements, as they understand them, and send them to the client to make sure there have been no miscommunications.

Proposal

- The proposal is the formal version of the client requirements as well as the plan for how these requirements will be met. Refer to the proposal for the Millwood project to get an idea for the necessary components. For sake of consistency, you might consider handing this proposal out to be used by your students as a template for their own. This will make marking easier for you as well as ensure that all clients are being treated equally. Both you and the client should approve the proposed work plan before the students begin any significant work on the project.
- The work plan is the most important part of the proposal as it sets the tasks and their deadlines for the project. The students should clearly define who is responsible for each task, how long it will take to complete the task, and the expected date of completion. Make sure the students have taken into account any dependencies between tasks and that the work loads are fair.

Development of the Project

- During development of the project, the students will work in their groups on the aspects of the project that best suit their skills. Encourage students to use a logical approach to complete the tasks outlined in the work plan. Encourage groups to discuss problems they are having in class in case members of other groups have already solved similar problems.
- Group meetings should be as frequent as necessary, and minutes should be recorded. The groups should strive to follow the work plan to avoid falling behind, and any changes to the work plan should be documented for future reference.

- This step includes testing, debugging, and fixing problems in the project. This is an excellent learning experience for students, as students can learn to identify the cause of problems to avoid them in the future.
- It is a good idea to have students review the Common Errors section of this document to avoid pitfalls wherever possible.

Progress Reports

- Progress reports allow both students and teachers to assess the ongoing progress of the project. Progress reports from the Millwood project can be found at the end of this section. Progress reports should be prepared throughout the development of the project at regular intervals (every one to two weeks) depending on the scope of the project. The progress reports should list the tasks that have been completed since the last progress report, the tasks that are currently under way, and the problems that have been encountered. In order for students to be aware of the changes that they have made to their original proposed work plan, students should critically assess their project in terms of their proposal in each progress report. This will also expose potential problems and delays quickly and allow solutions to be developed. Brief oral progress reports should be presented to the class at regular intervals, not only to make students aware of the progress of other groups, but also to allow groups to share ideas for solving problems.

Final Report

- The final report is an opportunity for students to reflect on their projects. A written final report could include a discussion of what worked, what did not, and what the students would have liked to go back and change in their original proposals. An oral report will allow students to share their successes and failures with the class.

Evaluation

- The client assessment of the project can be used carefully to gauge the performance of individual members as well as the group as a whole. While each group member will probably not be corresponding with the client on a regular basis, you can compare the client's assessment of the components of the project with the proposal to determine which group members fulfilled their duties. For example, if the client states that he or she found the documentation provided with the project very helpful, you could conclude that Student X listed on the proposal as being responsible for the documentation fulfilled his or her duties. Depending on your preferences, you could give the client a form to fill out or have a more informal interview. After you have digested the assessments from the client, you should give students some kind of feedback (a mark or a list of comments) so they can reflect on how their work on the project was evaluated by the client.

- It is important to understand the dynamics of the group as well. It is highly recommended that students evaluate each group member's contributions to the whole. You should account for the fact that individuals have different skills, aptitudes, and effort levels in any student evaluations.

Table 1. CP 12 curriculum outcomes in terms of the steps required to complete a computer project for a community group.

Step	Modules	Sub-Modules
Identify potential clients	N/A	N/A
Divide students into groups	N/A	N/A
Define client requirements	*Module 2:* Fundamentals of Programming	2.3 Demonstrate an understanding of how data structures are used to solve problems 2.2 Identify and frame problems
	Module 3: Applied Problem Solving	3.1 Work individually and collaboratively to develop program tools, components, and strategies to create solutions
	Module 4: Project Development	4.1 Analyse a problem
Proposal	*Module 1:* Problem Solving in Computer Programming	1.6 Demonstrate an understanding of ethical, moral, and legal issues in information technology
	Module 2: Fundamentals of Programming	2.4 Use appropriate methods and terms to develop a plan to solve a problem 2.6 Demonstrate an understanding of the effectiveness of other people's programs and documentation (applies if students take on a project that has already been started)
	Module 4: Project Development	4.2 Develop a project plan, including definition, scope, roles, resources, steps, and deadlines for a solution 4.4 Identify information needs and locate, evaluate, and select resources 4.5 Build and deploy a solution
Development of the project	*Module 2:* Fundamentals of Programming	2.5 Apply and plan to solve a problem using a programming language
	Module 3: Applied Problem Solving	3.2 Create a user interface using effective design principles 3.6 Apply error-handling techniques/validation 3.1 Work individually and collaboratively to develop program tools, components, and strategies to create solutions
	Module 4: Project Development	4.4 Develop a solution to a programming problem 4.6 Create documentation associated with the project 4.7 Test and refine the solution

Step	Modules	Sub-Modules
Progress reports	*Module 4:* Project Development	4.9 Reflect on the solution, the process, and their own learning
Final report	*Module 4:* Project Development	4.8 Present the solution 4.9 Reflect on the solution, the process, and their own learning
Evaluation	*Module 4:* Project Development	4.9 Reflect on the solution, the process, and their own learning

Common Errors in Project Completion

The following problems are commonly met during completion of any project. They have been explained here to demonstrate how they relate to the type of computer projects for community groups that students in CP 12 will be completing.

Project Scope

- Be realistic. Sketch out a project proposal and determine if students would reasonably be able to complete the project in the time allotted.
- A small-scale, but fully successful project will in many ways be more beneficial than a larger, unmanageable project because the students will have the opportunity to experience the entire process of development.
- In addition, you have a responsibility to the clients that the projects be completed. If you develop a reputation for incomplete projects, it will be harder and harder to find clients every year.

Unclear Client Requirements

- Ensure that you and your students schedule enough time to fully understand client requirements. The exact amount of time depends on the scope and type of project.
- Generally, people from community groups do not have the technical expertise to know what they want and/or if what they want is even possible.
- This means that multiple meetings are required to assess their requirements.
- A preliminary meeting with the teacher should identify the scope of the project and determine if it would fit within the class. In addition, it should get the client thinking about what they want/need.
- Additional meetings with the students in the group (accompanied by the teacher for at least the first meeting) are then required to determine how the client requirements can be met by the skills of the students.
- The client should have a copy of the proposal outlined by the students. They should be given the opportunity to approve the proposal or make changes. This is the best way to ensure there have been no misunderstandings between the students and the clients.
- Once the requirements have been agreed upon, avoid the desire to work outside the proposed work plan. Unexpected changes, regardless of how good they are, may change the client's expectations, thus requiring another redefinition of the project.

Telling the Client What They Want

- If the client has requirements that cannot be met, there should be mutual discussion regarding why they cannot be met and the alternative options.
- Clients should never feel like they are being told what they want. If the client feels like they designed the project they will have fewer complaints in the end.

- If the client has an idea to solve a problem, but the students feel they have a better idea, again, mutual discussion is the best way to reach an agreement. Never develop a solution without client approval, even if you think it is the "right" way to do it. You could misunderstand the client requirements or just make them unhappy despite a functional final product.

Scheduling More Than One Task to Be Completed on the Same Day

- A proposal in which more than one deliverables due on the same day makes it difficult to gauge progress.
- Instead, each deliverable should have its own deadline in the proposal.
- Larger tasks should be divided into reasonably sized units that can be completed in smaller periods of time.

Underestimating Time Requirements For Tasks

- It should be stressed to students that they carefully estimate their time requirements for tasks.
- Underestimation could lead to students trying to complete at the last minute tasks that they have no hope of finishing.

Common Errors in Database and Website Design

Two of the most common projects required by community groups are databases and websites. The project featured in this case study involved the development of both, and as could be expected, we ran into errors along the way. We have documented these errors and our solutions with the objective of providing some guidance to future students completing similar projects.

Common Errors in Database Design

Conforming the Project to the Available System

- This is especially important when working with community groups. They often have no budget for the project and so cannot go out and buy software/hardware that might be required for the project.
- Solutions:
  - Creative problem solving is often the best way to deal with insufficient client resources. Encourage students to rephrase the client's requirements in terms of the resources that are available.
  - Occasionally, free resources are available to eligible community groups. Locally, the Chebucto Community Net <http://www.chebucto.ns.ca/> is able to provide inexpensive web space to some non-profit organizations.

Common Errors in Website Design

Unclear Labelling of Links

- Links must be labelled clearly so that users understand where they will be directed when they click.
- Link labels must please the client. For example, if the client wants the "home" link to go to his personal website when it should really go to the homepage of the community group's website, the importance of consistency should be explained to the client.

Poor Navigation

- Try to separate distinct content as much as possible. Failure to do so creates a frustrating "maze" of links.
- Most pages in a website should be accessible in a minimum of three mouse-clicks. Having to follow a lengthy path of links is time-consuming for website users and challenges the users' interest.
- Provide "exits" for users that allow them to navigate to 1) the previous page and 2) the main page. Being forced to follow a series of links is undesirable and frustrates users.

Simple Aesthetic

- Use colour to your advantage. Use colour themes to group content relating to specific concepts. For example, helpful tips may be provided in a different coloured box than warnings.
- Do not overload the user with activity on the screen! Use animations sparingly and only when absolutely necessary.
- Attempt to separate concepts and content physically on the screen. If links are provided in a menu, for example, inserting important text into the menu confuses the user of its purpose.
- Be consistent! The user will come to associate specific meaning or purpose with your usage of colour, space, and animations. Ensure that the meaning of each of these areas is consistent throughout the website.
- The adage "less is more" is critical to website design. Providing unimportant information detracts from the purpose of the website.

Communicate Effectively

- Use language on your website that will be easily understandable to all users of your site. If you use rare or unfamiliar terms, such as technical jargon or slang, provide explanations.
- Try to use correct spelling and grammar.
- Providing access to versions of your page in other languages is always a good idea, if you have the time and knowledge to do so. Do not try to provide both languages on a single page.

Exceptions to the Rules

These rules are meant as guidelines only. Use your own discretion in making sure your website is user friendly.

Useful Design Resources

- Baecker, R., J. Grudin, W. Buxton, and S. Greenberg. Readings in Human Computer Interaction: Towards the Year 2000. 2nd Edition. San Francisco, CA: Morgan Kaufmann Publishers, 1995.
- Norman, D. A. The Psychology (Design) of Everyday Things. New York: Basic Books, 1989.
- Williams, Robin. The Non-designers Design Book. Berkeley, CA: Peachpit Press/Addison Wesley, 1994.
- Nielsen, Jakob. Usability Engineering. Cambridge, MA: AP Professional, 1993.
- Raskin, Jeff. The Humane Interface: An Engineering Guide for Product and Software Developers, Addison Wesley, 2000.
- Lewis, C. and J. Rieman. Task-Centered User Interface Design: A Practical Introduction. Shareware book. 1994. <www.hcibib.org/tcuid/>
- Diamond Bullet Design. <www.diamondbullet.com>
- Thom, James. The Usability Methods Toolbox <jthom.best.vwh.net/usability/>

Project Description Template

Term Project: Community Outreach

Teacher Name: ____

Computer Programming 12

You will be part of a team of students developing a computer-related project for a community group (the client). Your team will write a proposal for the project, develop the project, and deliver progress reports, a mid term report, and a final report describing how your project is going, what problems you are having, what successes you are having, etc. Please see the templates attached for details on what is required in each report. The oral reports will be a 10-minute summary of your written reports that you will give to the class. Every student on the team must participate in the oral reports.

Table 2

Component	**Due Date**	**Value**
Proposal (written, oral)		15%
Progress Report #1		5%
Progress Report #2		5%
Mid-term Report (written, oral)		20%
Progress Report #3		5%
Progress Report #4		5%
Final Report (written, oral)		25%
Assessment of Final Product	N/A	25%

The assessment of the final product is based on student effort as well as the finished product from the client's perspective. Therefore, it is very important that the projects are completed and functional for the clients.

Please fill out the Skill Survey to indicate what computer skills you have, what projects you would like to work on, and what people you would like to work with. Note that all requests may not be accommodated.

CP#12 Skills Survey Template

Student Name: ____

Date: ____

Please indicate your experience in the following areas with a number from 1 to 5, with 5 being the best. Then list up to 5 computer-related skills that you have and indicate your experience with them.

	Exp.
• Website design	____
• Graphic design	____
• Team leadership	____
• Interpersonal communications	____
• Planning and group organization	____
• Computer hardware skills	____
• Home networking	____
• ____________________	____
• ____________________	____
• ____________________	____
• ____________________	____
• ____________________	____

Please rank the projects in order of interest to you (1 is first preference, 2 is second, etc.)

Project	Rank
1.	_____
2.	_____
3.	_____
4.	_____
5.	_____
6.	_____

If there are persons enrolled in the class whom you would like to have on your team, please list their names below. Also, note that you may not end up with the team members or project that you prefer.

Proposal Template

André Dion
Erin Mazerolle
Andrew Shand
Mark Zwicker

Mr. Charles Blinn
Vice-Principal
Millwood High School
Sackville, Nova Scotia
October 19, 2004

Dear Mr. Blinn:

Thank you for meeting with our group to discuss the technical needs of the 10000 Steps-A-Day program, which is getting under way at Millwood High School. We understand that you are interested in developing a website that will allow participants of the 10000 Steps-A-Day program to record and view their progress in a database. The following report includes

- an executive summary highlighting important information in this proposal
- background information regarding the needs of the 10000 Steps-A-Day program
- a description of the proposed approach with a detailed work plan
- a description of the required resources for the project

Executive Summary

The requirements for Millwood High School's 10000 Steps-A-Day program, as understood by our team, include the development of a web-accessible database for use by program participants to record the number of steps they take each day. In addition, a survey is required to track the program's outcome and determine if it was beneficial to the participants. Millwood High School's vice-principal, Mr. Charles Blinn, is the client for this project.

Our team intends to complete the objectives mentioned above (and discussed further in this report) by November 25, 2004. By this date, we foresee the website and database being fully functional. We estimate that the project will take between 77 and 145 hours of work, a value of $1540–$2900. As part of the Community Outreach course at Dalhousie University, this project will be provided to Millwood High School free of charge.

Members of the team are André Dion, Andrew Shand, Erin Mazerolle, and Mark Zwicker. André is the designer and technical lead for the website and database. With aid from the rest of the team, André will create the website and database. Andrew will develop documentation for both the people maintaining and using the database and will also be designing and implementing the survey. Erin and Mark will be documenting the project by recording in detail the steps the group takes to reach completion and explaining the design process.

This will allow teachers to use this project as a case study in the new Computer Programming 12 class that is currently being piloted in Nova Scotia high schools. For more information, please see the proposal for Dalhousie Computer Science Community Outreach Committee, which is attached to the end of this document.

Background Information: 10000 Steps-A-Day Program

The need to encourage physical activity, especially in youth, is becoming more and more apparent. With obesity rates at record levels, people must be educated about making healthy lifestyle choices and including physical activity in their daily lives. At Millwood High School, the administration is taking action against sedentary habits by introducing the 10000 Steps-A-Day program. This program, principally aimed at inactive individuals, encourages people to walk more by providing pedometers so participants can measure their progress. With a grant from the Province of Nova Scotia, Millwood High School plans to distribute pedometers to interested students and teachers in an effort to promote physically active lifestyles in their community.

To be successful, the 10000 Steps-A-Day program must motivate participants to choose to walk. Individually, participants are motivated by the pedometers, which give a quantitative measurement of their achievements over the course of a day. Quantitative measures allow participants to set specific goals and evaluate their performance. To sustain motivation for long periods, additional tactics can be employed. Graphical displays can be provided that show how individual participants have increased their physical activity over time. Other statistics that may have more meaning to participants than the number of steps (such as calories burned and equivalent food items or distance travelled) can also be calculated and displayed to give participants a more concrete impression of their accomplishments. Within the context of a high school, it is possible to take advantage of "school spirit" sentiments by providing data on how far the school as a whole has walked. In order to use these ideas to motivate participants, there must be a system in place to record and analyse the number of steps taken by each participant, each day, over the course of the program.

To this end, our group proposes to develop a web-accessible database to be used by 10000 Steps-A-Day participants to input the number of steps they take each day. We plan to allow participants to log in to this database and input their steps using a calendar-like interface. The participants will be able to enter their steps just for the current day or go back and enter steps for any days they might have missed. Each participant will also fill out a user profile containing personal information (date of birth, height, weight, sex, stride length, pedometer correction factor, etc.) This information will be used to calculate distance travelled and calories burned. In addition, we plan to implement graphical displays of the progress of both individuals and the school as a whole. (Please note that any graphical displays depend on the libraries available on the machine that will be hosting the database.) In the case that the appropriate libraries are not available, we will instead create a text output that can be inputted into a graphing program for easy graph creation.

Because the 10000 Steps A-Day program will be grant-funded by the Province of Nova Scotia, it is important to have data to show how the program affected the participants. To this end, our group will be implementing a survey in consultation with Mr. Blinn to assess the outcomes of the program. This information can be used to report back to the Province of Nova Scotia so that the endeavour can be evaluated.

In accordance with the requests of the client, Mr. Blinn, the web interface to the database will have the same general layout as the alumni section of the Millwood High School website, <http://
www.millwood.ednet.ns.ca/alumninew/index.htm>.

Proposed Work Plan

The objectives of this project are the development and implementation of a database to be used to record the progress of the 10000 Steps-A-Day participants at Millwood High School. We have constructed the following work plan, which highlights our individual skills and maximizes productivity within the available time frame. We are confident that our plan will produce a useful database for Millwood High School. Please see the Gantt charts attached for additional information regarding the work plan and how it fits in with the case study being developed based on this project.

Table 1: Work Plan

Task	**Responsible**	**Work Hours**		**Target Date**	**Deliverable**
		Min	**Max**		
1. Develop the basic website design and layout.	André	7	10	Oct. 21	Website interface (no scripts or databases)
2. Develop a rough draft of the survey that will be used to gauge the effectiveness of the program.	Andrew Mr. Blinn	4	6	Oct. 24	Survey questions
3. Document code while it is being written.	Andrew André	10	15	Nov. 2	Comments and information in code
4. Create log-in page, profile creation page, and profile tables.	André	15	20	Nov. 2	Functional log-in page, profile creation page, and profile tables
5. Implement survey on the website.	Andrew	5	7	Nov. 6	Functional web survey
6. Create data entry page and data tables.	André	10	15	Nov. 9	Functional data entry page and data tables
7. Document code for website more formally based off of comments written in the code by André, including code that was not finished by Nov. 2.	Andrew	3	5	Nov. 9	Completed documentation for website code
8. Create functions to graph data and results.	André	5	20	Nov. 16	Functional graphing functions, or tables of numbers that can be inputted into a graphing program
9. Write user manual for website.	Andrew	4	6	Nov. 18	Completed user manual
10. Code testing phase.	André Andrew	5	10	Nov. 25	Fully functional website
Total		77	145		

Project Resources

The staffing requirements for the proposed work plan will be met by students of Computer Science 3190 (Community Outreach) and will be provided free of charge to Millwood High School. Other required resources, such as website hosting, are already in place. The following budget table is meant only to communicate the scope and value of the project.

Table 4: Project Allocation Budget

Cost Category	**Cost Estimate**		**Comments**
	Max	Min	
Staff	$1540	$2900	Two four-year computer science students at $20/hour for 77–145 hours
Total	$1540	$2900	

Conclusion

Without motivation, it is very difficult for inactive people to commence and maintain a physically active lifestyle. The development and implementation of a database to support Millwood High School's 10000 Steps-A-Day program will provide motivation for the participants and, by doing so, encourage healthy living in our communities.

Mid-term Report Template

André Dion
Erin Mazerolle
Andrew Shand
Mark Zwicker

November 4, 2004

Mid-term Report: 10000 Steps-A-Day

Executive Summary

The following report details the progress to date on the Millwood High School 10000 Steps-A-Day program's website and database. Table 1 outlines our project proposal and how it has changed over the course of development. Table 2 describes the hours we have spent on the project and relates it to the project proposal to express the accuracy of our original plan. Reasons for the discrepancies between the actual hours and planned hours are discussed, and the problems encountered throughout the project are analysed. Finally, the steps required for completion of the project are outlined.

Project Description

The need to encourage physical activity, especially in youth, is becoming more and more apparent. With obesity rates at record levels, people must be educated about making healthy lifestyle choices and including physical activity in their daily lives. At Millwood High School, the administration is taking action against sedentary habits by introducing the 10000 Steps-A-Day program. This program, principally aimed at inactive individuals, encourages people to walk more by providing pedometers so participants can measure their progress. With a grant from the Province of Nova Scotia, Millwood High School plans to distribute pedometers to interested students and teachers in an effort to promote physically active lifestyles in their community. Community Outreach (CSCI 3190) is creating a website for participants to track their progress and stay motivated in consultation with the client, Millwood's vice-principal, Mr. Blinn. This website will also facilitate the administrative aspects of the program.

In addition to the project outlined above, the group is also undertaking the development of a case study outlining the steps required to successfully complete the website and database for Millwood High School. This case study will serve as a resource for Computer Programming 12 (CP 12) teachers throughout Nova Scotia. For more details on the progress of this aspect, please see "Mid-term Report: Case Study."

Progress to Date

Initially, a look and feel for the website was achieved that met the needs of the client. The client indicated he would like a similar design to Millwood High School's alumni website. Although this page had a pleasing exterior, it was inefficiently designed. Also, Mr. Blinn desired an aesthetic that would motivate the participants of the program to use the website regularly. Therefore, the website had to be visually appealing but also very quick to load and navigate. With these issues in mind, André created a clean and simple layout with a very attractive aesthetic.

Although the client indicated from the outset that the website and database were to be hosted on EDnet, the basic design was completed on a preliminary torch account due to the slow response from EDnet to our questions regarding their technical capabilities. Once we had received the required information, the website was transferred to the EDnet account, and work began on the underlying database and the user interface.

The tables that will store user profile information in the database have been drafted. Please see Figure 1 for an illustration.

The project also requires that the progress of the participants of the program be tracked. To this end, a survey will be administered at regular intervals such that the participant will be asked to complete the survey when they log in. Mr. Blinn has drafted the survey questions in consultation with the group, based on standard measures of physical well-being. All questions are multiple choice to facilitate its web-based nature, statistical tests, and a low time commitment from the participants. Currently we are in the process of implementing the web version of the survey, but we are keeping in mind that the questions might change as the project progresses based on client input.

The motivational aspects of the website require a creative flair. Mark has begun to accumulate facts and quotes that will be presented to users on the website for motivational purposes when they log in or input steps. A collection of facts will be available, and one will be randomly displayed to the participant. The facts will equate the number of steps taken by the participant (either in a single day or over the course of the program) to a real-life value, such as distance walked (e.g., as a fraction of marathon) or calories burned (e.g., as a fraction of a hamburger).

Table 5: Work Plan and Progress to Date

Task	**Responsible**	**Work Hours**			**Target Date**	**Actual Date**	**Deliverable**
		Min	**Max**	**Actual**			
1. Develop the basic website design and layout.	André	7	10	7	Oct. 21	Oct. 20 (+1)	Website interface (no scripts or databases)
2. Determine database/ website requirements and if all necessary components are installed on Ednet.	André Andrew Mark,Erin Mr. Blinn	8	21	9	Oct. 22	Oct. 26	Ability to access EDnet functions
3. Develop a rough draft of the survey that will be used to gauge the effectiveness of the program.	Andrew Mr. Blinn	4	6	4	Oct. 24	Nov. 2 (–9)	Survey questions
4. Document code while it is being written.	Andrew André	10	15	2 5	Nov. 2	X (–2)	Comments and information in code
5. Create log-in page, profile creation page, and profile tables	André	15	20	15	Nov. 2	X (–2)	Functional log-in page, profile creation page, and profile tables
6. Implement survey on the website.	Andrew	5	7	2	Nov. 6	X	Functional web survey
7. Create data entry page and data tables.	André	10	15	5	Nov. 9	X	Functional data entry page and data tables
8. Document code for website more formally based on comments written in the code by André, including code not finished by Nov. 2.	Andrew	3	5	0	Nov. 9	X	Completed documentation for website code
9. Create functions to graph data and results	André	5	20	0	Nov16	X	Functional graphing functions, or tables of numbers that can be inputted into a graphing program
10. Write user manual for website.	Andrew	4	6	0	Nov18	X	Completed user manual
11. Code testing phase	André Andrew	5	10	2	Nov. 25	X	Fully functional website
Total		**77**	**145**	**51**			

Please note that that an X in the Actual Date column indicates the task has not been completed. A negative number after the date indicates the task was completed before the Expected Date; whereas a negative number indicates that we are that many days behind schedule.

Table 6: Schedule Variance Report

Resource Variance	This Period	Cumulative
Planned	25	36
Actual	26	41
Variance	-1	-5

Hours Variance Analysis

This period

- 1 hour over budget
- This week we put in extra time working on the draft of the survey, which we had planned to complete in a previous reporting period but could not due to incomplete information

Cumulative

- 5 hours over budget
- Between 4 people, this is not a significant deviation. It is mainly due to shifting the dates that tasks are due, rather than failing to estimate time requirements accurately. Thus, we foresee being under budget in the next reporting period.

Problems/Concerns

The problems experienced by the group were relatively manageable. However, due to the nature of our issues (many of them depended on client/third party response), our progress was hampered for some tasks. After the initial switch from our original project, we needed additional meetings with the new client to redefine the requirements and establish an acceptable course of action. The client had difficulty solidifying the needs of the project at first, so the group was forced to create a proposal with a flexible structure to accommodate changes in client requirements. Indeed, we have had to shift tasks in response to client demands, but we have found that it has not significantly increased our workload. While it is not ideal to plan a project without all the details, our proposal still provided a framework to gauge our progress against.

Specifically, the client was unsure about the survey questions. While it was known from the outset that the survey would be used to track the benefits of the 10000 Steps-A-Day program for the participants at Millwood High School, the client was unsure about the types of questions to be asked or how the responses would be analysed. Before getting a draft of the survey to implement on the website, we waited for the client to decide if outside expertise would be consulted. However, we have recently obtained a draft of the survey from the client and have begun implementation on the website.

Millwood High School depends primarily on EDnet (Nova Scotia's Department of Education network) for their Internet needs. For the group, this meant we had to deal with Ednet to obtain the information required to use their server to host the website and database. EDnet was slow to respond to provide the necessary information at the outset. Later, we were also delayed by EDnet when we needed support to connect the website to the back-end database. These types of problems are not at all uncommon when using a host without direct access to it and, because we obtained access to a temporary account at Dalhousie to begin work on the website, did not significantly hinder our progress.

Planned for Remainder of Term

Despite a productive first half of the term, it is clear from Table 1 that there is much to be done to complete the project. In order to allow the client to update the website and access the data, the code must be well documented. The documentation process will continue to occur while the code is being written and will also include a user manual to be written once the code is complete. The remaining work on the website includes completion of the log-in page, the calendar for users to enter their steps, and a way to allow users to graph their progress (either on the website or by outputting data for Excel). We also plan to implement a random presentation of motivational facts (discussed above). The database tables to store information on the number of steps must also be designed and implemented. However, the project is on track, and we do not foresee any delays in deployment or transferring administrative knowledge to Mr. Blinn.

Figure 1. Database tables to store user profile information.

Profiles

Field	Type	Null	Default	Links to	Comments	MIME
username	varchar(20)	No				
password	varchar(20)	No				
first_name	varchar(20)	No				
last_name	varchar(20)	No				
reg_date	date	No	0000-00-00			
birth_date	date	No	0000-00-00			
age	tinyint(3)	No	0			
gender	varchar(6)	No				
height	tinyint(3)	No	0		In centimetres	
weight	tinyint(3)	No	0		In kilograms	
error	tinyint (3)	No	0			
stride	tinyint(3)	No	0			

Indexes:

Keyname	Type	Cardinality	Field
PRIMARY	PRIMARY	0	username
username	INDEX	None	username
username_2	INDEX	None	username

Space usage:

Type	Usage	
Data	0	bytes
Index	1,024	bytes
Total	1,024	bytes

Row Statistics;

Statements	Value
Format	dynamic
Rows	0
Creation	Nov 03, 2004 at 08:13 PM
Last Update	Nov 03, 2004 at 08:13 PM

Template: Progress Report

Millwood Project
André Dion
Erin Mazerolle
Andrew Shand
Mark Zwicker

November 18, 2004

Progress Report #3

Completed In Reporting Period

- Changed password for website
- Determined that the "final" version of the survey (that we had already implemented) was not actually the final version
- Prompted the client for the final version of the survey
- Edited the survey for typographical errors and ensured that the answer options included all possibilities (quality control)
- Received and implemented the final version of the survey
- Implemented functionality to create profiles for new users
- Added a link on the website to the homepage of Millwood High School
- Added log-in box to root page
- Inserted background image into page explaining CSCI 3190
- Discussed with client what to do when a user loses their password (administrator must reset password)
- Continued progress on documentation materials
- Completed basic security functions (password encryption for users)
- Obtained an implemented calendar month generator (Dr. Chiasson)

Problems/Concerns

- Major problem with the survey as we wasted time implementing the wrong version
- Difficulties extracting correct version from client
- Difficulties uploading calendar PHP files (due to EDnet)

Planned for Next Reporting Period

- Implement registration authorization so that administrator is notified when a new user registers and is able to authorize the new user
- Complete documentation materials
- Complete implementation of motivational materials (statistics, quotes, colour-coded calendar)
- Implement functionality for users to input steps on the calendar interface
- Implement automatic prompting of users to fill out survey
- Remove temporary warnings from website text
- Careful proofreading of website content for correctness and clarity
- Careful testing of the website functionality
- Deployment and transfer of website and documentation

Schedule Variance Report

Resource Variance	This Period	Cumulative
Planned	27	63
Actual	30	71
Variance	-3	-8

Hours Variance Analysis

This period

- 3 hours over budget
- It is difficult to assess this variance in meaningful terms due to the shuffling of tasks. The individual task that we were most over budget on was development of the survey questions, which took much more time than expected due to confusion regarding whether the survey questions were final and lack of client expertise in survey design. However, we compensated for this increase by shifting deadlines (for instance, we have not and are unlikely to complete the graph functions as originally planned). In addition, we contracted out the development of a script to generate calendar months, saving additional time.

Cumulative

- 8 hours over budget
- While not ideal, this is not a significant deviation for a group of four people over many weeks. It is not reflective of the status of the project, because some tasks have been pushed aside to make room for tasks that were more time consuming than expected.

Current Schedule Status and Schedule Variance Discussion

Overall the project is going well in that we foresee a functional product being transferred to the client by the date due. As mentioned above, we may have to rethink inclusion of graph functions due to time constraints. However, the client was not particularly enthusiastic about the implementation of the graph functions, and data will still be easily exportable to Excel where graphs can be created. In addition, we are confident that the other features (statistics, quotes, and colour-coded days on the calendar) will provide sufficient motivation to website users.

Appendix E: Rubrics

Course Outcomes Rubric

This chart represents suggested modifications of the outcomes to create a scale.

Module	Exceeds all expectations	Exceeds some expectations	Meets minimum expectations	Does not meet minimum expectations
1.0 Students will be expected to understand and apply the basic skills and processes of problem solving using computer programming	Demonstrate an understanding of the role of number systems in data storage (1.1)	Demonstrates knowledge of the role of number systems in data storage including web page colours and index sort	Demonstrates basic knowledge of the role of binary and decimal number systems in data storage	Has minimal knowledge of the role of number systems in data storage
	Demonstrates mastery of mathematical concepts such as operators and number systems (1.2)	Uses Boolean operators and interprets Venn diagrams	Uses Boolean operators when developing Internet search strategies	Has difficulty applying mathematical concepts
	Defines a problem using objects that can be generalized to many problems (1.3)	Defines a problem in explicit terms using an object-oriented analysis	Defines a problem with minimal use of objects	Does not define a problem in explicit terms
	Can identify and fully outline strategies to solve a range of problems (1.4)	Can identify and outline strategies to solve a range of problems	Can identify but not fully outline strategies to solve problems	Cannot identify and outline strategies to solve problems
	Demonstrates broad range of problem-solving skills and a deep understanding (1.5)	Demonstrates many problem-solving skills	Demonstrates limited range of problem-solving skills	Does not demonstrate a range of skills for solving problems
	Demonstrates and applies understanding of ethical, moral, and legal issues to new situations (1.6)	Demonstrates and applies understanding of ethical, moral, and legal issues to routine situations	Demonstrates minimal understanding of ethical, moral, and legal issues	Disregards ethical, moral, and legal issues
	Investigates a broad range of related career opportunities and prerequisites for entry (1.7)	Investigates a range of related career opportunities	Investigates a limited number of related career opportunities	Does not investigate related career opportunities
2.0 Students will be expected to identify problems, select effective strategies, and plan solutions	Demonstrates knowledge of advanced syntax and features of a programming language (2.1)	Makes occasional errors in general syntax and features of a programming language	Makes few errors in basic syntax and features of a programming language	Makes frequent errors in syntax and features of a programming language
	Demonstrates a mastery of framing problems (2.2)	Is able to outline most aspects of a problem	Is able to outline some aspects of a problem	Is unable to frame and outline problem requirements

Module	Exceeds all expectations	Exceeds some expectations	Meets minimum expectations	Does not meet minimum expectations
	Demonstrates a complete understanding of data (2.3)	Demonstrates an understanding of data	Demonstrates an understanding of some data	Does not demonstrate an understanding of data
	Uses many methods and terms to develop comprehensive plans (2.4)	Uses some methods and terms to develop plans	Uses few methods and terms to develop plans	Does not develop appropriate plans using methods and terms
	Solves many problems using a programming language (2.5)	Solves most problems using a programming language	Solves few problems using a programming language	Is unable to solve problems using a programming language
	Demonstrates a deep understanding of the effectiveness of other people's code and documentation (2.6)	Demonstrates an understanding of the effectiveness of other people's code and documentation	Demonstrates some understanding of the effectiveness of other people's code and documentation	Does not demonstrate an understanding of the effectiveness of other people's code and documentation
3.0 Students will be expected to apply programming techniques to develop solutions to a range of problems	Develops acceptable solutions both individually and collaboratively with a variety of participants (3.1)	Develops acceptable solutions both individually and collaboratively	Develops solutions either individually or collaboratively	Fails to develop solutions either individually or collaboratively
	Creates user interface that demonstrates design principles and understanding of needs of a variety of users including those with special needs (3.2)	Create user interface that demonstrates design principles and understanding of needs of typical users	Applies basic design principles to the design of the user interface	Creates poorly designed user interface
	Demonstrates a mastery of input/output operations (3.3)	Applies many input/output operations	Applies few input/output operations	Is unable to apply input/output operations
	Demonstrates a mastery of data manipulation (3.4)	Applies many data-manipulation techniques	Applies few data-manipulation techniques	Is unable to apply data-manipulation techniques
	Demonstrates a mastery of data formatting (3.5)	Applies many data-formatting techniques	Applies few data-formatting techniques	Is unable to apply data-formatting techniques
	Develops advanced error-handling techniques that anticipate a broad range of errors (3.6)	Develops error-handling techniques that anticipate typical errors	Develops techniques to respond to a limited range of errors	Fails to consider appropriate error-handling techniques

Module	Exceeds all expectations	Exceeds some expectations	Meets minimum expectations	Does not meet minimum expectations
4.0 Students will be expected to work collaboratively to define and solve a complex problem by creating a software application	Performs thorough analysis of problem, which suggests a variety of possible solutions (4.1)	Analyses some components of problems more completely than others	Performs minimal analysis before beginning to solve problem	Does not analyse problems
	Develops a thorough and realistic plan that includes time lines and resources required (4.2)	Develops a realistic plan that includes time lines and most resources	Develops a plan that includes time lines	Plan does not include realistic time lines or resources
	Assumes leadership role in group when appropriate; demonstrates behaviours that encourage other group members to contribute (4.3)	Contributes actively to the group; supports group leader	Contributes as member of a group	Does not participate in team
	Identifies and selects and evaluates a variety of relevant information resources (4.4)	Selects and evaluates a minimal set of relevant resources	Identifies minimal resources appropriate to project	Fails to identify appropriate resources
	Is a lead contributor to the building and deployment of the solution (4.5)	Is a strong contributor to the building and deployment of the solution	Is a minor contributor to the building and deployment of the solution	Does not contribute to the building and deployment of the solution
	Creates documentation that is complete and clearly formatted (4.6)	Creates documentation that completely defines/ explains solution	Creates minimal documentation with some error	Creates little or no documentation
	Tests solution under a variety of conditions and modifies appropriately (4.7)	Modifies solution to run under most test conditions	Makes minimal modification of solution to pass basic tests of functionality	Fails to modify solution as a result of test of solution
	Demonstrates thorough knowledge of subject and presents with confidence and without reference to notes; fully engages audience (4.8)	Demonstrates knowledge of subject and presents with confidence with some reference to notes; audience is engaged	Demonstrates knowledge of subject and presents with some confidence and reference to notes; minimal use of visuals and supplementary materials; audience is interested	Demonstrates minimal knowledge of subject and presents without confidence; poor, minimal, no, or inappropriate use of visuals and supplementary materials; audience not engaged
	Identifies possible improvements as a result of reflection on solution (4.9)	Identifies some improvements as a result of reflection	Demonstrates minimal impact of reflection	Demonstrates little evidence of reflection

Group Project Rubric

Name: ____

Project: ____

Part	Score (circle one)	Total
Part A: Team Building		
• Team Player	0 1 2 3	
• Took the activity seriously and worked appropriately	0 1 2 3	
• Team Contribution	0 1 2 3 4	/10
Part B: Team Deliverable		
• Web Page Design	0 1 2 3	
• Functionality	0 1 2 3	
• Logo	0 1 2	
• Mission Statement	0 1 2	
• Ground Rules	0 1 2	
• Roles	0 1 2	
• Content	0 1 2	
• Performance/Effort	0 1 2 3 4	/20
Part C: Project Construction		
• Use of Time	0 1 2 3 4	
• Teamwork	0 1 2 3	
• Use Cases	0 1 2 3	
• Consultations	0 1 2 3	
• Professionalism	0 1 2	
Part D: Final Product		
• Client Acceptance Factors	10 15 20 25 30	
• Team Presentation	1 2 3 4 5	
• Use Cases Followed	1 2 3	
• Design	1 2 3 4 5 6	
• Content	1 2 3 4 5 6	
• Individual Performance during Presentation	1 2 3 4 5	/55

Comments:

Final Mark _____%

Team Project Self-Evaluation

This evaluation is to be taken seriously and is strictly confidential. Results will be viewed only by the teacher. This is your opportunity to relate how your group experience went.

Rank each statement from 0 to 5, with 5 being the highest.

Self-Evaluation						
Name:						
I was always on task	0	1	2	3	4	5
I was a strong team member	0	1	2	3	4	5
I feel I did my share of group work	0	1	2	3	4	5
I rate my overall effort as	0	1	2	3	4	5
Comments:						

Self-Evaluation						
Name:						
I was always on task	0	1	2	3	4	5
I was a strong team member	0	1	2	3	4	5
I feel I did my share of group work	0	1	2	3	4	5
I rate my overall effort as	0	1	2	3	4	5
Comments:						

Self-Evaluation						
Name:						
I was always on task	0	1	2	3	4	5
I was a strong team member	0	1	2	3	4	5
I feel I did my share of group work	0	1	2	3	4	5
I rate my overall effort as	0	1	2	3	4	5
Comments:						

Self-Evaluation						
Name:						
I was always on task	0	1	2	3	4	5
I was a strong team member	0	1	2	3	4	5
I feel I did my share of group work	0	1	2	3	4	5
I rate my overall effort as	0	1	2	3	4	5
Comments:						

Appendix F: Moral, Ethical, and Legal Issues

The moral, legal, and ethical issues associated with computer programming and information technology are an important part of this course. Many issues, past and present, have faced those who program and those who use information technology. Below is a list of some potential topics that students should investigate and consider when creating their own projects. By examining these issues, students may increasingly become critical thinkers, aware of the ethical and moral issues associated with computer programming and information technology use.

- e-mail spam
- spyware
- hackers/crackers
- copyright
- viruses
- Microsoft antitrust
- hate sites
- pop-ups
- file sharing
- site blocking
- open source code

Copyright Issues

Students must respect the work of others and act in an ethically and legally responsibly manner. Just as students would not think it appropriate to steal a book from a store to support their studies or interests, students must not steal the information or works of others in the electronic environment. We are in an age where the technology to do things such as download images, data, or music files from websites is readily available. The legal right to use the technology for these purposes is, as students who pay attention to the news media understand, not always available. Students using the schools equipment and Internet connection must follow Canadian copyright and other laws. Students must respect the intellectual property of others and properly credit materials that are not their own original work.

Teachers and students will find many wonderful Internet resources to support learning in Computer Programming 12. To ensure that these resources are used in an ethical and legal fashion, teachers should direct students to the following excerpt from Copyright Matters! 2nd edition, Council of Ministers of Education, Canada, 2005.

Can teachers and students copy from the Internet?

From a copyright point of view, you should be aware of the following four rules:

1. Most material available on the Internet is protected by copyright. This includes text (e.g., postings to newsgroups, e-mail messages), images, photographs, music, video clips, and computer software. Under the Copyright Act, reproduction and unauthorized use of a protected work are currently infringements. Therefore, reproduction of any work or a substantial part of any work on the Internet would infringe copyright unless you have the permission of the owner. Many Internet users are questioning the

appropriateness of the rules in copyright law. Canada and other countries around the world are currently studying uses of copyright materials from the Internet. Many Internet users and service providers are asking for changes in copyright law that would allow defined uses of works on the Internet without infringing copyright. The ministers of Boards Association (CSBA), the Canadian Teachers' Federation (CTF), and others in the education community are active participants in this ongoing work.

2. Copyright protects the way in which information is expressed. The information itself is not protected by copyright. Restating ideas, facts, or information in your own words is not copyright infringement.
3. Where a work has been placed on the Internet with the message that it can be freely copied, there is an actual licence to copy the education in Canada (except in Quebec), the Canadian School work. Sometimes the terms of the licence are subject to conditions. Common conditions are that the posting cannot be used for commercial purposes, must be circulated in its entirety, cannot be used out of context, and cannot be edited or reformatted. If you abide by the conditions, you may copy the work without infringing copyright.
4. Any works protected by copyright that are on your school's or district's Web site require copyright clearance, unless the school or district already owns the copyright in them. If the school or district does not own the copyright, permission must be obtained from the copyright owner. The permission must be in writing. The same would apply for students accessing student Web sites.

Excerpted from *Copyright Matters!* 2nd Edition, Council of Ministers of Education, Canada, 2005 <www.cmec.calelse/copyright/matters/index.stm>.

Appendix G: Computer Programming Topics Checklist

Module 1

- ☐ Convert binary to decimal, decimal to binary, hex to decimal, decimal to hex, octal to decimal, and decimal to octal.
- ☐ Set Theory
- ☐ Set operators (union, intersection)
- ☐ Truth tables
- ☐ Boolean logic and operators (AND, OR, NOT)
- ☐ Problem Analysis charts
- ☐ IPO charts, IPO worksheets
- ☐ Flow Charts
- ☐ Algorithms
- ☐ Data Dictionary
- ☐ Data storage

Module 2

- ☐ Program Comments
- ☐ Variables and constants
- ☐ Data conversion
- ☐ Basic input/output
- ☐ Create methods with arguments, and return values
- ☐ IF statements and IF/ELSE statements
- ☐ Nested IF statements
- ☐ While and Do While loops
- ☐ For loops
- ☐ Nested loops

Module 3

- ☐ Create and organize classes
- ☐ Arrays
- ☐ Two-dimensional arrays
- ☐ Passing arrays to methods
- ☐ Searching strings and arrays
- ☐ Sorting strings and arrays
- ☐ File input/output
- ☐ GUI design principles
- ☐ Graphic methods
- ☐ Format data (strings, integers, etc)

- ☐ Error handling code (in Java try and catch)
- ☐ Applets
- ☐ Classes
- ☐ Inheritance
- ☐ Encapsulation
- ☐ Polymorphism

Module 4

- ☐ Team building skills
- ☐ Reflections (project, group, and self)
- ☐ Presentation skills
- ☐ Gantt Charts (time lines)
- ☐ Project plan
- ☐ Evaluation methods (peer, self, and project)
- ☐ Test plan
- ☐ External documentation
  - ○ User guide
  - ○ Help documentation
  - ○ Troubleshooting tips

Appendix H: Glossary of Terms

Computer Programming 12 Glossary

Binary-base two	A number representation consisting of zeros and ones used by practically all computers because of its ease of implementation using digital electronics and Boolean logic.
Bit	The smallest conceivable unit of information, equivalent to a single binary digit.
Boolean logic	Logic in which elements have one of two values. The algebraic operations defined on the set logical OR: a type of addition; and logical AND: a type of multiplication.
Byte	A unit of memory capacity equal to 8 bits.
Data dictionary	A design tool used to account for all objects required in a program.
Flow chart	A schematic representation of a sequence of operations, as in a manufacturing process or computer program.
Hexadecimal	A numeric system based on 16, used in creating web colours.
Information technology	Applied computer systems—both hardware and software—often including networking and telecommunications, usually in the context of a business or other enterprise.
Object-oriented programming	Is a method for reducing the complexity of the design problem by dividing it into a number of self-contained classes of objects. Each class corresponds to a well-defined real-world entity and interacts with other classes in limited, well-defined ways.
Principles of design	The elements and principles of design are the building blocks used to create visual representations.
Pseudocode	An outline of a program, written in English, that can easily be converted into a programming language.
Structured programming	The use of a limited set of computer language features to improve the readability, reliability, and ease of maintenance of a computer program.
Syntax	A set of rules for a language that determines whether a particular statement is correctly formulated.

Appendix I: Resources

Teachers should note that this list of resources is current as of the publication of this guide and that some resources will change over time. Teachers should consult Authorized Learning Resources, which can be accessed through the Department of Education website. As new print and web resources are approved for Computer Programming 12, they will be added to this list.

In addition to the print resources listed in this appendix, Internet resources can also be valuable when planning learning, teaching, and assessment activities. Teachers are responsible for checking websites before students access them to ensure that they are appropriate for student use. Teachers should be fully advised of provincial, board, and school policies pertaining to Internet use. In particular, teachers should familiarize themselves with the Internet Access and Use Policy for Nova Scotia Schools available through the Department of Education home page.

Print Resources

Student and Teacher Resources

- Hutchison, Don, and Mark Yannotta. *Mathematics for New Technologies*. Pearson, 2004.
  ISBN: 0-201-77137-3
  Description: This text clearly explains many of the concepts associated with Module 1. It includes information, examples, exercises, and tests on topics such as binary math, hexadecimal numbers, set theory, logic operators, Boolean logic, and logic circuits.
- Sprankle, Maureen. *Problem Solving and Programming Concepts.* Prentice-Hall, 2003.
  ISBN: 0-13-048268-4
  Description: This text provides an in-depth look at the problem-solving concepts required to program in any computer language. It encompasses all fundamental areas of problem solving from basic mathematical functions and operators to the design and use of code. Many examples and problems are included
- Morelli, Ralph. Java, Java, Java: Object Oriented Programming. Prentice-Hall, 2003.
  ISBN:0-13-033370-0
  Description: This text offers an object-orientated approach to problem solving in Java. The book focusses on problem decomposition and program design. The multiple exercises found in the text introduce the use of Java syntax and many programming concepts.
- Nova Scotia Department of Education and Culture. *Special Education Policy Manual*. Halifax, NS: Province of Nova Scotia, 1997.
- Fain, Yavok. *Java Programming for Kids, Parents and Grandparents*. <smartdataprocessing.com/java4kids.htm> [E-book].
- Association for Computing Machinery. *A Model Curriculum for K–12 Task Force Curriculum Committee*. New York, NY: Computer Science Teachers Association, 2004.
- Noel, Wanda. *Copyright Matters!: some Key Questions and Answers for Teachers*, 2nd Edition. Ottawa, ON: Council of Ministers of Education Canada, 2005.
- Baecker, R., J. Grudin, W. Buxton, and S. Greenberg. *Readings in Human Computer Interaction: Towards the Year 2000*. 2nd Edition. San Francisco, CA: Morgan Kaufmann Publishers, 1995.
- Norman, D. A. *The Psychology (Design) of Everyday Things*. New York: Basic Books, 1989.
- Williams, Robin. *The Non-designers Design Book*. Berkeley, CA: Peachpit Press/Addison Wesley, 1994.
- Nielsen, Jakob. *Usability Engineering*. Cambridge, MA: AP Professional, 1993.

- Raskin, Jeff. *The Humane Interface: An Engineering Guide for Product and Software Developers*, Addison Wesley, 2000.
- Lewis, C. and J. Rieman. *Task-Centered User Interface Design: A Practical Introduction*. Shareware book. 1994. <www.hcibib.org/tcuid/>
- Diamond Bullet Design. <www.diamondbullet.com>
- Thom, James. The Usability Methods Toolbox <jthom.best.vwh.net/usability/>

Web Resources

Website addresses are accurate as of the printing of this publication.

Content	**Search for**	**URL**
Java Tutorials	Freewarejava	freewarejava.com/tutorials/index.shtml
Designing Programs with flow charts, this site provides a tutorial on flow chart use, including an example	techtutorials flowcharts	techtutorials.net/
Logic Circuit Software, free ware that allows users to create logic gates and practise the use of operators and Boolean logic	MultiMedia Logic Download	schoolnet.gov.mt/micallef/donloads.htm
Sun's Java home page	Java sun	java.sun.com/
Sun Java's tutorial for new java programmers. An excellent resource and starting point.	Java sun tutorial	java.sun.com/docs/books/tutorial/information/download.html
Good java programming conventions	Javaranch style	javaranch.com/style.jsp
Unified Modeling Language. Used to represent objects, their properties and methods, and describe how objects interact.	UML or Unified Modeling language	www.uml.org/
An e-book written by Yakov Fain who contributed the article "Teaching Kids Programming," found in Appendix A	Java Programming for Kids, Parents and Grandparents	smartdataprocessing.com/java4kids.htm
ISTE Professional Development Site, contains links to presentations and papers on many relevant issues related to teaching computer programming	ISTE CS/IT Symposium	iste.org/Content/NavigationMenu/Professional_Development/Symposia/Computer_Science/2005/February_2005_CS_IT_Symposium/Speaker_Presentations/Speaker_Presentations.htm
Computer Science Teachers Association website, contains a model curriculum designed in the United States and many other useful resources	Computer Science Teachers Association	csta.acm.org/Curriculum/sub/ACMK12CSModel.html *(Note: This URL is case sensitive)*

Content	Search for	URL
Computer programming online dictionaries	Kevinboone, kzone,	www.kevinboone.com/compdict/compdict_index.html *(Note: This URL is case sensitive)*
Intel Museum: an excellent resource, this site contains a history of computer memory video in Shockwave format.	intel museum	intel.com/museum/online/memory_tech/index.htm
Math of computing notes and activities: this site covers number systems, binary math, operators, and logic gates.	Fanshawe college	gs.fanshawec.ca/tlc/math270/
This site contains many free resources including a time chart and project design resources.	technology student	www.technologystudent.com/designpro/despro1.htm