Global Learning
ON THE EDGE

Course Plan **Burnaby Online Program – School District #41 Burnaby**

Course: **Information Communications Technology 12 Programming Strand**

Course Description: ICT12 emphasizes is object-oriented programming methodology with an emphasis on problem solving, algorithm development and data structures and abstraction.
It is meant to be an introductory course to programming in Java.

Pre-requisites: As such there are no prerequisites but a guideline below may be helpful when choosing students for this course.
• Math 11 with a C+ or greater
• Information and Communications Technologies 11 with a B or greater

Overview of Course Goals:

Following is an overview of the six major topics covered by ICT12.

T1. Object-Oriented Program Design

The overall goal for designing a piece of software (a computer program) is to correctly solve the given problem. At the same time, this goal should encompass specifying and designing a program that is understandable, can be adapted to changing circumstances, and has the potential to be reused in whole or in part. The design process needs to be based on a thorough understanding of the problem to be solved.

T2. Program Implementation

The overall goals of program implementation parallel those of program design. Classes that fill common needs should be built so that they can be reused easily in other programs. Object-oriented design is an important part of program implementation.

T3. Program Analysis

The analysis of programs includes examining and testing programs to determine whether they correctly meet their specifications. It also includes the analysis of programs or algorithms in order to understand their time and space requirements when applied to different data sets.

T4. Standard Data Structures

Data structures are used to represent information within a program. Abstraction is an important theme in the development and application of data structures.

T5. Standard Algorithms

Standard algorithms serve as examples of good solutions to standard problems. Many are intertwined with standard data structures. These algorithms provide examples for analysis of program efficiency.

T6. Computing in Context

A working knowledge of the major hardware and software components of computer systems is necessary for the study of computer science, as is the awareness of the ethical and social implications of computing systems.

Curricular Requirements: The course schedule references where specific topics will be covered.

[C1] This course teaches students to code fluently in an object oriented paradigm using the programming language Java. The course teaches students to use many of the standard Java library classes.
[C2] The course teaches students to design and implement computer-based solutions to problems in a variety of application areas.
[C3] The course teaches students to use and implement commonly used algorithms and data structures.
[C4] The course teaches students to develop and select appropriate algorithms and data structures to solve problems.
[C5] The course teaches students to identify the major hardware and software components of a computer system, their relationship to one another, and the roles of these components within the system.
[C6] The course teaches students to read and understand a large program consisting of several classes and interacting objects, and enables students to read and understand the program and be able to modify or add to it.
[C7] The course teaches students to recognize the ethical and social implications of computer use.

Texts:
West, Tom, and Christine Stephenson. *Java for AP* Computer Science*. Toronto, ON, Canada: Holt Software Associates Inc., 2003.

Course Schedule:
Weeks 1 - 2
Topic **Computer Hardware, Software and Computer Programming**
Topics [C2] [C3] [C5] [C7]
• Computer Systems
• Computer basics
• Java basics
• Using the READY IDE
• Basic output
Objectives:
• Identify types of computers
• Identify hardware design
• Understand computer software
• Explore computer systems, networks and their usage
• Understand social issues and responsible computer usage
• Use the READY IDE to enter, compile and execute programs
• Understand program errors and the Program Life Cycle

• Distinguish between several program languages, design techniques and object oriented programming
• Types of Java Programs
Resources: Chapters 1 & 2
Assessments:
• Perform an analysis and create a specification for the automation of a task performed at school.
• Explain why it is necessary for a Java applet to have security limitations.
• Specify some of the other useful operations that could be added to the L*ibraryCollection* class.
Strategies:
• Discuss types of computers, basic hardware design, computer software, system usage, and networks. Explore social issues involved in computing, namely privacy, piracy, intellectual property, and the social impact of computers.
Outline proper personal use of computers.
• Introduce the fundamentals of writing and running programs on computers (i.e., the differences between compilers, interpreters, and compiler interpreters
and an overview of different programming languages.)
• Introduce the concept of object oriented programming

Weeks 3 - 6
Topic **Java Programming (Data Types & Expressions)**
Topics [C2] [C3] [C4] [C1]
• Variables and Data Types
• Assignment statements
• Error types
• Using expressions

Objectives:
• Identify the different parts of Java programs
• Use appropriate declarations and naming conventions
• Use appropriate indentation and nesting with curly braces
• Understand terminology, comments, variables, constants, reserved words, literals
• Declare and assign variables
• Understand the eight simple data types and their range
• Understand the String type, declaration, concatenation, and type conversion
• Use casting to convert data types
• Understand expressions and operator precedence
• Use logical and comparison operators
• Understand short circuit evaluation
• Understand private and reference data types

Resources: Chapters 3 & 4
Labs:
• The *MinAndMax* and *RollDice* programs.
Assessments:
• Complete the 3.1.10 Exercises (Familiarizing you with Java and the Java development environment.)
• The *Square* and *Initials* program
• Complete Exercise #6-#8 (expressions, declarations, evaluations & simplifications)

• Identify the short circuit evaluations in Exercise #19.
• Unit Quiz (Chapters 1-4 inclusive)
Strategies:
• The students should predict the output for several small programs without entering them in to the READY IDE.
• Students will need to create several simple programs to illustrate appropriate variable declarations, assignment and formatted output.
• Students should write programs that incorporate Boolean, expressions, comparison and logical operators, mathematical functions and simplifying
various statements by using compound statements.

Weeks 7 - 12
Topic **Java Programming (Input & Output, Objects & Methods, Selection & Loops)**
Topics [C2] [C3] [C4] [C1]
• Program design.
• Input and output
• Programming style.
• Creating objects methods use
• Selection constructs
• Loops

Objectives:
• Understand how to use System.in and System.out as well as the use of the hsa.Stdin class (for simple input).
• Students will demonstrate appropriate programming style.
• Demonstrate the creation of objects and the use of methods.
• Students will manipulate class methods.
• Understand how to format text output.
• Demonstrate appropriate use of selection constructs including if statements, ifelse structures, multi-way if structures, and nesting of selection constructs using different indenting styles.
• Write programs using looping constructs (i.e., counted loops (for loops) and
conditional loops (while and do-while loops)

Resources: Chapters 5, 6, 7 & 8
Labs:
• Develop the QuadraticInput and TermDeposit programs
Assessments:
• Write out an executionflow diagram from a nested if-else construct
• Implement several classes including the AddressBook class and the PaintBug class.
Strategies:
• Allow students to practice writing programs using different types of loops and conditionals.
• Students should complete different classes, choosing appropriate representations.

Weeks 13 -15
Topic Top Down Design, Strings & Methods
Topics [C2] [C3] [C4] [C1]
• Introducing top-down design

• Program testing
• The String class (methods & objects)
• Writing methods.
• Using class methods
Objectives:
• Understand step-wise refinement and full path testing
• Demonstrate advanced string manipulations using string
methods and objects (including StringBuffer and StringTokenizer classes).
• Understand how to write methods (including function type methods, procedure type methods and parameter passing).
• Understand how to use class variables.

Resources: Chapters 9, 10 & 11

Labs:
• Phone book and a Mortgage programs.
Assessments:
• Using step-wise refinement, modify the PhoneBookExample program so
that it will notify the user if he or she attempts to add a name that already exists.
• BetweenWords & IsUpper programs
• PalindromeLine program
• TestInflate & TestMax programs
• Correct the DrawBoxes so that it completes a grid of four squares
Strategies:
• Give students classes to complete in which they must choose appropriate
representation for the given class.
• Work through examples of the string methods, objects and substrings

Weeks 16 -19
Topic Modifying & Creating Classes, Arrays, Inheritance & Interfaces
Resources: Chapters 12 – 14
Labs: BankArrayTest & DayOfYearList programs
Labs & Assessments:
• Use an array to control a series of bank accounts and a class that uses arrays to implement a phone book.
Topics [C2] [C3] [C4] [C1]
• Create objects and classes.
• Constructors and helper methods
• Creating and manipulating arrays
• Inheritance, polymorphism, abstract classes, and class hierarchies
Objectives:
• Understand the syntactical detail necessary to write a class with fields and
methods.
• Declare, create, and use of arrays.
• Understand how to work with related arrays and
• Perform operations on arrays such as inserting and removing an element.
• Using classes that implement inheritance, polymorphism, abstract classes,and class hierarchies.
Strategies:

• Use an animation program written in Java, allowing students to learn the concepts of inheritance, polymorphism, abstract classes, and class hierarchies while producing graphically interesting programs.

Week 20 **Exam Week Assessments**:
Midterm Exam 20% (Chapters 1 – 14)
Weeks 21 - 23 **Recursion, Searching and Sorting**
Topics [C2] [C3] [C4] [C1]
• Recursion
• Searching (binary, linear)
• Generosity and the Object class
• Multiple Inhertiance
• ArrayList class
• Selection sort, insertion sort, merge sort, and quicksort.
Resources: Chapters 15- 17
Labs & Assessments:
• Towers of Hanoi, the Dragon Curve, theo programs
Strategies:
• Provide practice programs for manipulating sorting and searching classes and programs
Objectives:
• Use recursive programming techniques, including mutual recursion.
• Understand and manipulate programs with linear search, and binary search routines.
• Use the ArrayList class in a program.
• Understand how to use insertion sort, merge sort, and quicksort.
• Distinguish amongst the specific characteristics of each sort.

Teaching Strategies
This course is designed for both students with some programming background and novice programmers. Students work individually on weekly hands-on programming assignments and labs about four to six hours per week. Students are encouraged to assist one another with learning the material through dialogue in the form of forums online.

How will your mark be calculated?

Assignments	45%
Quizzes	20%
Midterm	10%
Final	25%

Students are expected to:
- contact the teacher by instant messaging, email or phone when help is needed or questions arise
- be actively engaged and submitting work on a regular basis
- inform the teacher when they will be inactive for two or more weeks.
- be aware that if they are inactive in a course for four or more weeks they may be removed from that course

- check their email at least twice a week
- create and submit completed solutions for all activities in the unit/chapter before requesting a test.
- cite all sources properly
- answer in their own words
- check that their work and tests have been marked.
- make time available to come in to Burnaby Online to write tests.
- make appointments to write tests at least 2 school days in advance.