Introduction to Computer Science Overview

Content Area: **Computer Science & Business**
Course(s): **INTRODUCTION TO COMPUTER SCIENCE**
Time Period:
Length: **90 Days**
Status: **Published**

Cover

EAST BRUNSWICK PUBLIC SCHOOLS

East Brunswick New Jersey

Superintendent of Schools

Dr. Victor P. Valeski

BOARD OF EDUCATION

Todd Simmens, President

Vicki Becker, Vice President

Susanna Chiu

Robert Cancro

Liwu Hong

Laurie Lachs

Barbara Reiss

Chad Seyler

Meredith Shaw

Course Adoption: 4/21/1986

Curriculum Adoption: 4/21/1986

Date of Last Revision Adoption: 9/1/2017

Course Overview

COURSE DESCRIPTION

The Introduction to Computer Science course is a one semester introductory computer science class. Students taking the course are required to have taken a prerequisite math course of at least Algebra I. The course focuses on using a programming language to solve problems. Students will be introduced to an object oriented programming language. They will design and develop programs to be run and tested on the computer. The course introduces students to interactive input and output methods, file input and output, control structures for selection and iteration, assignment statements, math operators. Students are also introduced to Java data types including basic primitive types (int, long, double, char, boolean) as well as standard classes (String, and Math etc.). Students are required to take tests and write between 50-75 independent programming assignments in the semester.

COURSE SCOPE AND SEQUENCE

Sequential Unit Description	Associated CPI's to be Achieved	Marking Period Guide	Other Pacing Guide References	Proficiency (Summative) Assessments
Unit 1: Introduction to Computers and Programming • Describe the hardware and software components of a typical computer system • Understand the history of computing and computers • Understand the terminology of Computer Science • Use an IDE with an editor and a compiler to create programs to execute on a computer • Describe the steps in the program development process • Create simple programs with input and output • Use variables to store data • Document a program using comments	8.2.2.E.1 8.2.2.E.2 8.2.2.E.3 8.2.2.E.4 8.2.5.E.1 8.2.5.E.2 8.2.5.E.3 8.2.12.E.2 8.2.12.E.3	1	7 days	Formative • Class discussion Summative: • Vocabulary exercise • Unit 1 Programming assignments (1-3) • Unit test

Unit 2: Writing programs using String objects and String class methods • Define string and understand its implementation in Java as A String class • Input and output Strings • Use simple String class methods to process String data • Use a do..while loop to repeat program statements • Use String, char and int variable to store data related to strings • Debug simple errors in programs	8.2.12.E.1 8.2.12.E.2 8.2.12.E.3 8.2.12.E.4		1	7 days	Formative: Sample class programs Summative: Collected Exercises Unit 2 programs (1-3) Unit tests
Unit 3: Writing programs using primitive numeric data • Understand the difference between numeric data (int and double) and String objects • Perform mathematical calculations with numeric data using mathematical operators following the order of operations • Declare valid meaningful variables to reference data in a program • Understand how to write/identify valid numeric data values • Input and output numeric	8.2.12.E.1 8.2.12.E.2 8.2.12.E.3 8.2.12.E.4		1	17 days	Formative: Sample class programs Exercises Summative: Collected Exercises Unit 3 programs(1-5) Unit Tests

data • Format numeric data for output • Use methods of the Math class to perform numeric calculations					
Unit 4: Using conditional control structures to make decisions in a program • Introduce the if/else selection structures to make decisions • Write boolean expressions for conditions using reference operators and Boolean operators • Introduce nested ifs to create multilevel selection structures • Use String class methods to compare String objects • Understand how Strings objects are compared using the ascii code of their char data	A-APR 1 A-REI.1 A-SSE 1a A-SSE 1b A-SSE 2 F-BF 1a F-BF 2 F-LE b N-Q 1	N-Q 2 N-Q 3 8.2.12.E.1 8.2.12.E.2 8.2.12.E.3 8.2.12.E.4	1/2	18 days	Formative: Sample class programs Exercises Summative: Collected Exercises Unit 4 programs(1-7) Unit Tests
Unit 5: Using loop control structure to repeat statements • Introduce the while loop and the for to repeat program statements • Use a “counter” to control a loop • Use loops to generate data to create a “chart”	A-SSE 1 A-SSE 2 A-SSE 4 F-BF 1a F-BF 3 F-LE 1a F-LE 3	N-Q3 8.2.12.E.1 8.2.12.E.2 8.2.12.E.3 8.2.12.E.4	2	16	Formative: Sample class programs Exercises Summative: Collected Exercises Unit 5 programs (1-9) Unit Tests

• Discuss runtime errors associated with incorrectly constructed loops • Use a loop to “traverse” a string character by character • Process individual char data in a string • Use a loop to “factor” an int • Create and use “nested” loop structures	N-Q 1 N-Q 2				
Unit 6: Using randomness in programs • Discuss the concept of randomness in a program • Introduce the Random class to generate “pseudorandom” values in a program • Create a “formula” to generate random int values in a specified range	A-REI 1 A-REI 3	8.2.12.E.1 8.2.12.E.2 8.2.12.E.3 8.2.12.E.4	2	11 days	Formative: Sample class programs Exercises Summative: Collected Exercises Unit 6 programs (1-9) Unit Tests
Unit 7: Methods • Discuss the concept of reusing code • Introduce use of an API • Create methods for repeated use • Use methods to break down a difficult math idea • Create appropriate documentation	N-Q 1 N-Q 2 N-Q 3 A-APR 1 A-SSE 1a A-SSE 3 F-IF 1 I-IF 4	8.2.12.E.1 8.2.12.E.2 8.2.12.E.3 8.2.12.E.4	2	7 days	Formative: Sample class programs Summative: Unit 7 programs (1-6) Unit Tests
					Formative:

Unit 7: Civics • Discuss backing up computers • Introduce the need for security • Create password policies • Discuss how to prevent online attacks • Discuss file sharing legal issues • Describe methods for continuing education	8.1.8.D.1 8.1.12.D.2 8.1.12.D.3 8.1.12.D.4 8.1.12.D.5	8.2.12.E.1 8.2.12.E.2 8.2.12.E.3 8.2.12.E.4	Ongoing 7 days	Class discussions Summative:	

CONTENT FOCUS AREA AND COURSE NAME

Course Name: Introduction to Computer Science - #1452

Course Number	School Numbers	Course Level	Grads(s)	Credits	Min. Per Week	Elective/Required	Initial Course Adopted
1452	050	10-12	10-12	2.50	210	E	04/21/86

Textbooks and Other Resources

Java an Introduction to Computer Science and Programming(Walter Savitch) Second Edition Prentice Hall

Teacher prepared samples and exercises

Standards

MA.A-APR.A	Perform arithmetic operations on polynomials

MA.A-APR.C	Use polynomial identities to solve problems
MA.A-APR.D	Rewrite rational expressions
MA.A-CED	Creating Equations
MA.A-SSE.A	Interpret the structure of expressions
MA.A-SSE.A.2	Use the structure of an expression to identify ways to rewrite it. For example, see $x^4 - y^4$ as $(x^2)^2 - (y^2)^2$, thus recognizing it as a difference of squares that can be factored as $(x^2 - y^2)(x^2 + y^2)$.
TECH.8.1.8.D.1	Understand and model appropriate online behaviors related to cyber safety, cyber bullying, cyber security, and cyber ethics including appropriate use of social media.
TECH.8.1.8.E.1	Effectively use a variety of search tools and filters in professional public databases to find information to solve a real world problem.
TECH.8.1.8.E.CS1	Plan strategies to guide inquiry.
TECH.8.1.8.E.CS2	Locate, organize, analyze, evaluate, synthesize, and ethically use information from a variety of sources and media.
TECH.8.1.12.D.2	Evaluate consequences of unauthorized electronic access (e.g., hacking) and disclosure, and on dissemination of personal information.
TECH.8.1.12.D.3	Compare and contrast policies on filtering and censorship both locally and globally.
TECH.8.1.12.D.4	Research and understand the positive and negative impact of one's digital footprint.
TECH.8.1.12.D.5	Analyze the capabilities and limitations of current and emerging technology resources and assess their potential to address personal, social, lifelong learning, and career needs.
TECH.8.2.2.E.1	List and demonstrate the steps to an everyday task.
TECH.8.2.2.E.2	Demonstrate an understanding of how a computer takes input through a series of written commands and then interprets and displays information as output.
TECH.8.2.5.E.1	Identify how computer programming impacts our everyday lives.
TECH.8.2.5.E.2	Demonstrate an understanding of how a computer takes input of data, processes and stores the data through a series of commands, and outputs information.
TECH.8.2.5.E.3	Using a simple, visual programming language, create a program using loops, events and procedures to generate specific output.
TECH.8.2.12.E.1	Demonstrate an understanding of the problem-solving capacity of computers in our world.
TECH.8.2.12.E.2	Analyze the relationships between internal and external computer components.
TECH.8.2.12.E.3	Use a programming language to solve problems or accomplish a task (e.g., robotic functions, website designs, applications, and games).
TECH.8.2.12.E.4	Use appropriate terms in conversation (e.g., troubleshooting, peripherals, diagnostic software, GUI, abstraction, variables, data types and conditional statements).

Grading and Evaluation Guidelines

GRADING PROCEDURES

In terms of proficiency level the East Brunswick grades equate to:

A Excellent Advanced Proficient

B Good Above Average Proficient

C Fair Proficient

D Poor Minimally proficient

F Failing Partially Proficient

COURSE EVALUATION

Each quarter students will be evaluated with tests and programming assignments using a total point basis to determine the quarter average. The semester/course average will be a weighted average of the 2 quarter averages (40% each) and a final exam (20%)

Course achievement will be evaluated based on the percent of all pupils who achieve the minimum level of proficiency (final average grade) in the course. Student achievement levels above minimum proficiency will also be reported. Final grades, and where relevant mid-term and final exams, will be analyzed by staff for the total cohort and for sub-groups of students to determine course areas requiring greater support or modification.)

Other Details

10152 Computer Programming

Computer Programming courses provide students with the knowledge and skills necessary to construct computer programs in one or more languages. Computer coding and program structure are often introduced with the BASIC language, but other computer languages, such as Visual Basic (VB), Java, Pascal, C++, and COBOL, may be used instead. Initially, students learn to structure, create, document, and debug computer programs, and as they progress, more emphasis is placed on design, style, clarity, and efficiency. Students may apply the skills they learn to relevant applications such as modeling, data management, graphics, and text-processing.