Saint Patrick High School

Curriculum Guide

Department	Mathematics
Class	Computer Science 1
Grade Level	Sophomore, Junior, Senior
Term(s)	All Year

Required Text(s)	**Mozilla Developer Network** - Online Documentation (https://developer.mozilla.org/en-US/) **W3Schools** - Online Tutorials (https://www.w3schools.com) **Site Point E-Books** ● JavaScript from novice to Ninja 2nd Ed ● CSS Master **Eloquent JavaScript** - Online Textbook (http://eloquentjavascript.net)
Addition Resources	Hardware: ● School Laptops ● Binder for handouts, notes, assignments Software: ● VS Code (https://code.visualstudio.com/) ● NodeJS (https://nodejs.org/en/) ● Google Chrome (https://www.google.com/chrome/)

Course Description

This course is designed to teach students the fundamentals of computer science. No previous background in programming is necessary. Students will learn how to design algorithms to complete tasks by writing code with proper syntax and grammar in JavaScript. A short list of concepts covered includes: data types, variables & values, operations, relations, conditions, functions, objects, classes, graphs, recursion, and events. Students will tie all of these idea together by learning the basics of static web development. Teacher approval is required to be admitted successfully.

Table of Contents

Unit	Time
Unit 1: Imperative Programming	5 weeks
Unit 2: Control Flow & Functions	3 weeks
Unit 3: Types	4 weeks
Unit 4: Objects	4 weeks
Unit 5: Standard Algorithms	6 weeks
Unit 6: Graphs & Recursion	4 weeks
Unit 7: Web Development	8 weeks

Standards

This class draws upon standards from both the Common Core Standards for Mathematical Practice and the Next Generation Science Standards. This field of study cannot claim to be the sole heir to either body of knowledge; therefore both are set or standards are highlighted. You may view the corresponding descriptions of all standards at the websites reference here.

http://www.corestandards.org/Math/Practice/

http://www.nextgenscience.org/three-dimensions

Unit 1: Imperative Programming

Essential Questions

- What are the key elements of a programmable task?
- What are the rules of correct syntax and proper programming grammar?
- How does a computer execute a program?
- What are the basic operations and relations on primitive data?

Learning Targets

- Students will be able to identify basic components of computer architecture and describe its responsibility in executing a program.
- Students will be able to reason deductively and create algorithms with various levels of abstraction using pseudo code.
- Students will be able to identify an expression vs a statement
- Students will be able to create and update a stack frame as they read line by line of the code.
- Students will be able to assign values to variables: let, var, and const.
- Students will be able to use basic operations for numbers and booleans.
- Students will be able to convert an integer to binary and/or hexadecimal.
- Students will become proficient with Atom / VS Code.
- Students will Sign up and complete assignments for REPL.it
- Students will be able to download and upload a repository to GitHub

Unit 2: Control Flow & Functions

Essential Questions

- How can processes be encapsulated in functions?
- How do conditionals and loops extend imperative programming?
- How does scope affect the execution of a program?
- How can functions give rise to other functions?

Learning Targets

- Students will be able to describe a function's signature.
- Students will be able to differentiate the invocation of a function and its declaration.
- Students will be able to write conditions with numbers, booleans, and strings with their basic relations.
- Students will be able to judge and implement inclusive and mutually exclusive conditional statements by demonstrating the range of outputs.
- Students will be able to design loops to simplify computation/completion of a programmable task.
- Students will be able to judge whether a programmable task is better implemented with a for-loop or a while-loop.
- Students will be able to look at pre-written code and describe the accessibility of all of its variables in terms of scope.
- Students will be able to read a function and argue whether it satisfies its intended goal by deciphering its internal algorithm.
- Students will be able to write es6 arrow functions and pass them as arguments to other functions.
- Students will be able to chain functions with return statements to produce complex outputs.

Unit 3: Types

Essential Questions

- How does the abstraction of a data type differ from a variable?
- In what practical and syntactical ways are string and array related?
- Which data types come pre-made with JavaScript and what can be done with them?

Learning Targets

- Students will be able to change a variables type by calling a constructor function.
- Students will be able to compare data types with double and triple equals articulating whether the statement is truthy or falsey.
- Students will be able to modify the contents of an array using a for-loop.
- Students will apply the three essential functional programming operators on arrays: map, filter, and reduce.
- Students will be able to build a string out of ASCII characters.
- Students will be able to log template strings with custom messages.
- Students will be able to implement loops to produce string expressions
- Students will apply the three essential functional programming operators on strings: map, filter, and reduce.

Unit 4: Objects

Essential Questions

- How does the abstraction of an object compare to data types and variables?
- What do objects allow us to model in a program?
- How might different types of objects work together to complete a programmabled task?
- What is meant by the phrase “separation of concerns?

Learning Targets

- Students will be able to define an object literal with instance variables and instance functions.
- Students will be able to write an object constructor function that returns an objects.
- Students will be able to create an new object via the prototypal chain.
- Students will be able to get and set instance variables in an object to change its state.
- Students will be able to determine the context and use cases of the keyword “this” as defined in the JavaScript language.
- Students will implement object to model real world scenarios and manage their state.

Unit 5: Standard Algorithms

Essential Questions

- What are the dimensions of an algorithm we must consider when choosing one?
- How do we use assumptions, case-analysis, and contradiction to prove the validity of our algorithms?
- What questions and algorithms are at the heart of computer science?

Learning Targets

- Students describe the impact on data-types on the space efficiency of a given algorithm.
- Students articulate the impact of loops on the time efficiency of a given algorithm.
- Students will be able to rank operations by their payload on the CPU.
- Students will be able to reproduce famous numeric algorithms (find max/min, square-root, center of mass)
- Students will be able to defend the validity and efficiencies of bubble-sort, selection-sort, and insertion-sort.

Unit 6: Graphs & Recursion

Essential Questions

- What does a graph describe in the discrete sense?
- What are the relevant characteristics which underlie a graph?
- What characteristics about a function make it recursive?
- How could we write recursive functions to work with arrays and graphs?

Learning Targets

- Students will be able to label or draw a graph from vertex and edge specifications.
- Students will be able to classify if two graph are isomorphic.
- Students will be able to represent a graph as an n by n associative matrix.
- Students will be able to calculate degree measures and other measures of centrality/reachability.
- Students will be able to perform matrix operations which correspond to graph manipulations.
- Students will be able to construct tree structured graphs, particularly binary trees provided a sequence of inputs.
- Students will be able to write a single condition recursive algorithm which completes a programmable task.
- Students will be able to write a multi condition recursive algorithm which completes a programmable task.
- Students will be able to create tail-recursive algorithms which and prove that they are faster for computation algorithms.

Unit 7: Web Development

Essential Questions

- What are the components of a website and how are they interdependent?
- How can we understand a websites by thinking of them as graphs?
- How can we understand a websites by thinking of them as objects?

Learning Targets

- Students will be able to describe a web page as a tree of HTML elements, objects, with various parents, children and siblings. The Document Object Model.
- Students will be able to use css selectors and operations to navigate the DOM.
- Students will be able to apply the css box model to create consistent html layouts.
- Students will be able to apply flexbox to create flexible content.
- Students will be able to add behavior to their website through the use of transitions and animations.
- Students will be able to use media queries to create layouts for multiple devices.
- Students will be able to add/remove event listeners to elements on a website to alter the DOM.
- Students will experiment with web frameworks, import their requirements, and implement their methods.