Introduction to Programming Nanodegree Syllabus

Learn to Code

Before You Start

Prerequisites: No prior experience with programming is required. You will need to be comfortable with basic computer skills, such as managing files, running programs, and using a web browser to navigate the Internet.

You will need to be self-driven and genuinely interested in the subject. No matter how well structured the program is, any attempt to learn programming will involve many hours of studying, practice, and experimentation. Success in this program requires meeting the deadlines set for your cohort and devoting at least 10 hours per week to your work. This requires some tenacity, and it is especially difficult to do if you don't find the subject interesting or aren't willing to play around and tinker with your code—so drive, curiosity, and an adventurous attitude are highly recommended!

You will need to be able to communicate fluently and professionally in written and spoken English.

Educational Objectives: This introductory Nanodegree program teaches the foundational skills all programmers use, whether they program mobile apps, create web pages, or analyze data. It is ideal for beginners who want to learn new skills, make informed choices about career goals, and set themselves up for success in career-track Nanodegree programs.

Length of Program*: 120 Hours
Textbooks required: None
Instructional Tools Available: Video lectures, Mentors, Forums

*The length is an estimation of total hours the average student may take to complete all required coursework, including lecture and project time. If you spend about 10 hours per week working through the program, you should finish in 12 weeks, so approximately 3 months. Actual hours may vary.

Part: Intro to HTML *Estimated Time: 10 hours*

For this section, you will submit your very first programming file containing HTML code. HTML is the coding language for building websites. We recommend taking notes from this section and using your notes as the content for your HTML file. This project is a lab that is auto-graded in the classroom.

Supporting Lesson Content: Intro to the Web and HTML Basics

Lesson Title	Learning Outcomes
Welcome to the Nanodegree	➔ Understanding on how to set up for the program on your personal device ➔ Introduction to the "Programmer Mindset" ➔ Successfully writing and rendering your first lines of HTML code with a text editor and browser
Nanodegree Orientation	➔ Understanding on how to submit projects ➔ Understanding on student support services offered for students ➔ Habits of Successful Students
The World Wide Web	➔ High level overview on how the web works ➔ Components of the web: browsers, HTTP requests, Servers, the Internet
HTML Basics	➔ HTML tags ➔ Adding Images ➔ HTML Syntax ➔ Whitespace ➔ Inline vs Block elements ➔ HTML Document Structure

Part: Intro to CSS *Estimated Time: 10 hours*

In this section, you'll learn both HTML and CSS - both languages for developing websites. For the project, you'll use HTML and CSS to make Animal Trading Cards. You will apply your knowledge of HTML Document Structure to your html file and then create custom CSS styling based on your preferences. This project will demonstrate your understanding of linking CSS files in HTML files, implementing CSS classes to avoid repetition, as well create semantically organized HTML code.

Supporting Lesson Content: HTML Syntax & CSS

Lesson Title	Learning Outcomes
Adding CSS to HTML	➔ Understanding CSS basics ➔ Divs, Spans, and Classes ➔ Semantic Tags ➔ Using DevTools in the Browser ➔ Verifying HTML and CSS files ➔ Debugging HTML and CSS code ➔ Page Structure ➔ Visual Styling ➔ Designing with Boxes
Animal Trading Cards	➔ Apply what you've learned to create your first code-reviewed project.

Part: Intro to Python *Estimated Time: 60 hours*

In this section, you will learn the Python programming language. You will finish by building your own interactive game using Python that can be shared with your friends.

Supporting Lesson Content: Python Programming

Lesson Title	Learning Outcomes
Turtles & Python	➔ Write your first lines of Python code using turtles - a visual module that displays colorful rendition of programming
Functions - Part 1	➔ Learn how to use functions to take an input and transform it into some output ➔ Explore syntax error messages and troubleshoot basic Python code
Functions - Part 2	➔ Understand the difference between print and return statements ➔ Learn how to manage the flow of a computer program using Boolean values, if statements, and While loops
Shell Workshop	➔ Understand and practice working with the Command Line Interface (CLI)
Python at Home	➔ Installing Python and learning Command Line Interface (CLI) basics ➔ Learn how to store values in **Variables** and work with text as **Strings** ➔ Selecting substrings with String Indexing
Strings & Lists	➔ Learn how to store values in Variables and work with text as Strings ➔ Selecting substrings with String Indexing ➔ Use String methods: **slicing**, **concatenation**, **find,** and **replace** ➔ Use **Lists** to store more complex data ➔ Use **for** loops to programmatically access each item within a List
Version Control with Git	➔ Learn about the benefits of version control and install version control
Working with Files	➔ Understand how files are created and stored in computer memory ➔ Learn how to list files in a directory, work with filenames, and move and organize files ➔ Learn how to read text from a text file, process that text using string operations, and write text to another text file ➔ Work through some common bugs in text processing
Objects & Classes	➔ Understand the difference between Objects and Classes in programming ➔ Define a new Class, understand the “self”, and defining special methods like initializers ➔ Creating instance variables ➔ Learn about inheritance, super Classes, and Class variables

Part: Intro to JavaScript *Estimated Time: 40 hours*

Learn the history of JavaScript and how it compares to Python programming. Understand how the DOM is formed, what Nodes and Elements are, and how to select items from the DOM. By the end, you'll write JavaScript code that allows the user to create a grid of squares representing their design, and apply colors to those squares to create a digital masterpiece.

Supporting Lesson Content: Introduction to JavaScript

Lesson Title	Learning Outcomes
What is JavaScript?	➔ Understand the history of JavaScript and start writing your code immediately using the JavaScript console
Data Types & Variables	➔ Learn to represent real-world data using JavaScript variables, and distinguish between the different data types in the language
Conditionals	➔ Learn how to add logic to your JavaScript programs using conditional statements
Loops	➔ Harness the power of JavaScript loops to reduce code duplication and automate repetitive tasks
Functions	➔ Dive into the world of JavaScript functions. Learn to harness their power to streamline and organize your programs
Arrays	➔ Learn how to use Arrays to store complex data in your JavaScript programs
Objects	➔ Meet the next JavaScript data structure: the Object. Learn to use it to store complex data alongside Arrays
ES6 Syntax	➔ Learn how to update your code to comply with this major JavaScript update
The Document Object Model	➔ Understand how the DOM is formed, what Nodes and Elements are, and how to select items from the DOM
Creating Content with Javascript	➔ Use JavaScript and DOM methods to create new page content, update existing content, and delete content
Working with Browser Events	➔ Learn what an event is, how to listen for an event and respond to it, what data is included with an event, and the phases of an event

Discover Your Path *Extracurricular*

In this section, there is no project submission. Instead, you will explore a quick overview of the vast world of programming. After this section, you'll have a better understanding of different options you have as a programmer. This will help guide to in your final project for this program.

Supporting Lesson Content: Exploration of Five Programming Career Tracks

Lesson Title	Learning Outcomes
Front-End Programming	➔ Learn about front-end web developers who create intuitive and responsive websites
Back-End Programming	➔ Learn about back-end web programmers who write server-side code to build web apps that serve millions of people worldwide
Mobile Programming	➔ Learn about mobile programming and the differences between iOS and Android programming
Data Analysis Programming	➔ Learn about data analysts who analyze data to direct growth and make informed decisions
Reverse Engineer Project Compass	➔ Dissect a fully-functioning web app and identify the roles of various programmers in its creation

A Taste of... *Extracurricular*

Explore five paths of development: Front-End, Back-End, Android, Data Analyst, and iOS.

Supporting Lesson Content: Choice of content from five career-track programs.

Elective Title	Learning Outcomes
Front-End Developer	➔ Learn all about Responsive Web Design with this interactive lesson.
Back-End Developer	➔ Get a taste of databases and SQL with this introduction on Back-End.
Android Developer	➔ Check out Android Studio and XML with this short lesson.
Data Analyst	➔ Explore Python Notebooks and two new Python modules used for Data.
iOS Developer	➔ Try out Swift, the main programming language for iOS.