

15-111, College-Level Programming and Computer Science Syllabus

1. Course Description

A technical introduction to the fundamentals of programming with an emphasis on producing clear, robust, and reasonably efficient code using top-down design, informal analysis, and effective testing and debugging. Starting from first principles, we will cover a large subset of the Python programming language, including its standard libraries and programming paradigms.

This is a rigorous and fast-paced course that requires at least one year of prior programming experience at the high school level. It is equivalent to Carnegie Mellon University's 15-112 course and should substitute for any first-semester college programming course as a result.

NOTE: students must achieve a C or better on the proctored final to satisfy the prerequisite for any subsequent Computer Science course at Carnegie Mellon such as 15-122 or 15-150.

2. Learning Objectives

At the end of the course, students should be able to:

1) Write clear, robust, and efficient code in Python using:

- sequential, conditional, and loop statements
- strings, lists, tuples, sets, and dictionaries
- objects and classes
- recursive approaches
- graphics and interaction

2) Develop programs to effectively solve medium-sized tasks by:

- employing modular, top-down design in program construction
- demonstrating an effective programming style based on established standards, practices, and guidelines
- proactively creating and writing test cases to test and debug code
- applying computational problem-solving skills to new problems
- explaining and analyzing the efficiency of algorithms, particularly by predicting the Big-O running time of small pieces of code

3) Design and write a substantial (500-1500 line) program in Python with minimal guidance

3. Topic List:

- **1 Basic Programming Constructs**
 - **1.1 Getting Started**
 - **1.2 Data, Expressions, and Variables**
 - 1.2.1 Types
 - 1.2.2 Constants
 - 1.2.3 Variables
 - 1.2.4 Operators
 - 1.2.5 More About Operators
 - 1.2.6 Code Tracing
 - **1.3 Functions**
 - 1.3.1 Intro
 - 1.3.2 Helpful Functions
 - 1.3.3 Writing Your Own Functions
 - 1.3.4 Variable Scope
 - 1.3.5 Print Versus Return
 - 1.3.6 Test Functions
 - 1.3.7 Helper Functions
 - 1.3.8 Console IO Functions
 - 1.3.9 Code Tracing with Functions
 - **1.4 Conditionals**
 - 1.4.1 if Statements
 - 1.4.2 if-else Statements
 - 1.4.3 Nested Conditionals
 - 1.4.4 if-elif-else
 - 1.4.5 if-else Expressions
 - 1.4.6 Short-Circuit Evaluation
 - 1.4.7 Improper Use of Conditionals
 - 1.4.8 Code Tracing with Conditionals
- **2 Loops And Strings**
 - **2.1 Getting Started**
 - **2.2 Loops**
 - 2.2.1 for loops and range
 - 2.2.2 Nested for loops

- 2.2.3 while loops
- 2.2.4 break and continue
- 2.2.5 Code Tracing With Loops
- **2.3 Style**
- **2.4 Strings**
 - 2.4.1 Intro
 - 2.4.2 String Literals
 - 2.4.3 String Operators
 - 2.4.4 String-related Builtin Functions
 - 2.4.5 String Methods
 - 2.4.6 Looping over Strings
 - 2.4.7 Strings are Immutable
 - 2.4.8 String Formatting with f Strings
 - 2.4.9 Code Tracing with Strings
- **2.5 Top-Down Design**
- **3 Animations**
 - **3.1 Getting Started**
 - **3.2 Our First Animations**
 - 3.2.1 Drawing Rectangles and Labels
 - 3.2.2 Mouse Press Events
 - 3.2.3 Model-View-Controller (MVC)
 - **3.3 Basic Shapes**
 - 3.3.1 Rectangles
 - 3.3.2 Ovals and Circles
 - 3.3.3 Lines
 - 3.3.4 Labels
 - 3.3.5 Intersections
 - **3.4 Mouse Events**
 - 3.4.1 Mouse Presses and Releases
 - 3.4.2 Mouse Moves and Drags
 - **3.5 Key Events**
 - 3.5.1 Key Presses and Releases
 - **3.6 Timer Events**
 - 3.6.1 Timer Events
 - **3.7 More Shapes, Colors, Images, and Sounds**
 - 3.7.1 Regular Polygons and Stars
 - 3.7.2 Arcs

- 3.7.3 More Colors (rgb and gradient)
- 3.7.4 Images
- 3.7.5 Sounds
- **3.8 Motion**
 - 3.8.1 Bounded Motion
 - 3.8.2 Wraparound Motion
 - 3.8.3 Bouncing Motion
 - 3.8.4 Circular Motion
- **3.9 Calling runApp with Optional Arguments**
- **4 Lists**
 - **4.1 Getting Started**
 - **4.2 1d Lists**
 - 4.2.1 Introduction
 - 4.2.2 Creating Lists
 - 4.2.3 List and String Similarities
 - 4.2.4 Visualizing Code Execution
 - 4.2.5 List Mutability
 - 4.2.6 List Operators
 - 4.2.7 List-Related Builtin Functions
 - 4.2.8 List Methods
 - 4.2.9 List-Related String Methods
 - 4.2.10 Looping over Lists
 - 4.2.11 List Comprehensions
 - 4.2.12 Writing Mutating and Non-Mutating Functions
 - 4.2.13 Code Tracing with Lists
 - **4.3 Tuples**
 - 4.3.1 Tuples
 - **4.4 2d Lists**
 - 4.4.1 Introduction
 - 4.4.2 2d List Literals
 - 4.4.3 Variable-Length 2d Lists
 - 4.4.4 2d List Dimensions
 - 4.4.5 Looping over 2d Lists
 - 4.4.6 Accessing Rows And Columns
 - 4.4.7 Copying 2d Lists
 - 4.4.8 Non-Rectangular 2d Lists
 - 4.4.9 3d Lists

- 4.4.10 Variable Names
- 4.4.11 Code Tracing with 2d Lists
- 4.4.14 Word Search Case Study
- **5 Animations With Lists**
 - **5.1 Getting Started**
 - **5.2 Animations With 1d Lists**
 - 5.2.1 Drawing Multiple Rectangles
 - 5.2.2 Key Holds
 - 5.2.3 Polygons
 - **5.3 Animations With 2d Lists**
 - 5.3.1 Getting Started
 - 5.3.2 Drawing a 2d Board
 - 5.3.3 Cell Selection
 - 5.3.4 Tic-Tac-Toe and Connect4 Case Studies
 - 5.3.5 Snake Case Study
 - 5.3.6 Tetris Case Study
- **6 Sets, Dictionaries, and Efficiency**
 - **6.1 Getting Started**
 - **6.2 Sets**
 - 6.2.1 Introduction
 - 6.2.2 Creating Sets
 - 6.2.3 Using Sets
 - 6.2.4 Properties of Sets
 - 6.2.5 Code Tracing with Sets
 - **6.3 Dictionaries**
 - 6.3.1 Introduction
 - 6.3.2 Creating Dictionaries
 - 6.3.3 Using Dictionaries
 - 6.3.4 Properties of Dictionaries
 - 6.3.5 Code Tracing with Dictionaries
 - **6.4 Efficiency**
 - 6.4.1 Introduction
 - 6.4.2 Measuring Efficiency
 - 6.4.3 Big O
 - 6.4.4 Searching and Sorting
 - 6.4.5 Hash Tables
 - 6.4.6 More Examples

- 6.4.7 Efficiency Practice
- **7 Recursion**
 - **7.1 Getting Started**
 - **7.2 Code Tracing with Recursion**
 - **7.3 Recursion Practice**
 - **7.4 More Recursion Practice**
 - **7.5 Some Recursion Issues**
 - **7.6 Some Recursion Classics**
 - 7.6.2 Greatest Common Divisor (gcd)
 - 7.6.3 Towers of Hanoi
 - 7.6.4 Flood Fill
 - 7.6.5 Quick Sort
 - 7.6.6 Merge Sort
- **8 Object Oriented Programming (OOP)**
 - 8.1 Getting Started
 - 8.2 Writing Classes
 - 8.3 Writing Methods
 - 8.4 Equality Testing
 - 8.5 Using Instances in Sets and Dictionaries
 - 8.6 Class Attributes
 - 8.7 Additional Topics
 - 8.8 Wrapping Up
 - 8.9 Code Tracing with OOP