
1866 AUB American University of Beirut الجامعة الأميركية في بيروت

Maroun Semaan Faculty of Engineering and Architecture

# INTRODUCTION TO COMPUTATION AND PROGRAMMING USING PYTHON

EECE 230X | 3 Credit Hours | Online Course

## COURSE DESCRIPTION

This is an introductory programming course with an emphasis on abstractions and elementary algorithmic ideas. It uses the Python programming language. Topics include: data types, selection, repetition, lists, tuples, strings, functions, files, exception handling, program efficiency, recursion, divide and conquer algorithms, recurrence relations, sorting and searching algorithms, binary search, merge sort, randomized quicksort, dictionaries, memorization, classes and object-oriented programming, stacks and queues, graphs, and graph traversal algorithms. The course has a weekly lab.

+961 1 374374 ext.4478 Registration Link fswd@aub.edu.lb

# Introduction to Computation and Programming Using Python

**Pre-requisites**

Elementary high school math

**Course Instructor**

Prof. Louay Bazzi,
*ECE department, MSFEA, AUB*

**Reference book**

Guttag, John.
Introduction to Computation and Programming Using Python: With Application to Understanding Data, Second Edition. MIT Press, 2016.

## COURSE LEARNING GOALS

The objectives of this course are to give students:

- An understanding of the principles of programming using Python
- An understanding of elementary algorithms concepts including searching, sorting, recursion, and time analysis
- An ability to write programs to solve new problems

## COURSE SPECIFIC LEARNING OBJECTIVES

After completing the course, the students are expected to have learned to:

- Apply the principles of functional programming
- Implement searching and sorting algorithms
- Solve computational problems using searching and sorting
- Analyze the efficiency of elementary algorithms
- Solve computational problems using recursion
- Solve computational problems using elementary data structures such as two-dimensional lists, dictionaries, stacks, and queues
- Apply the principles of object-oriented programming

# TOPICS

## MODULE I Foundations

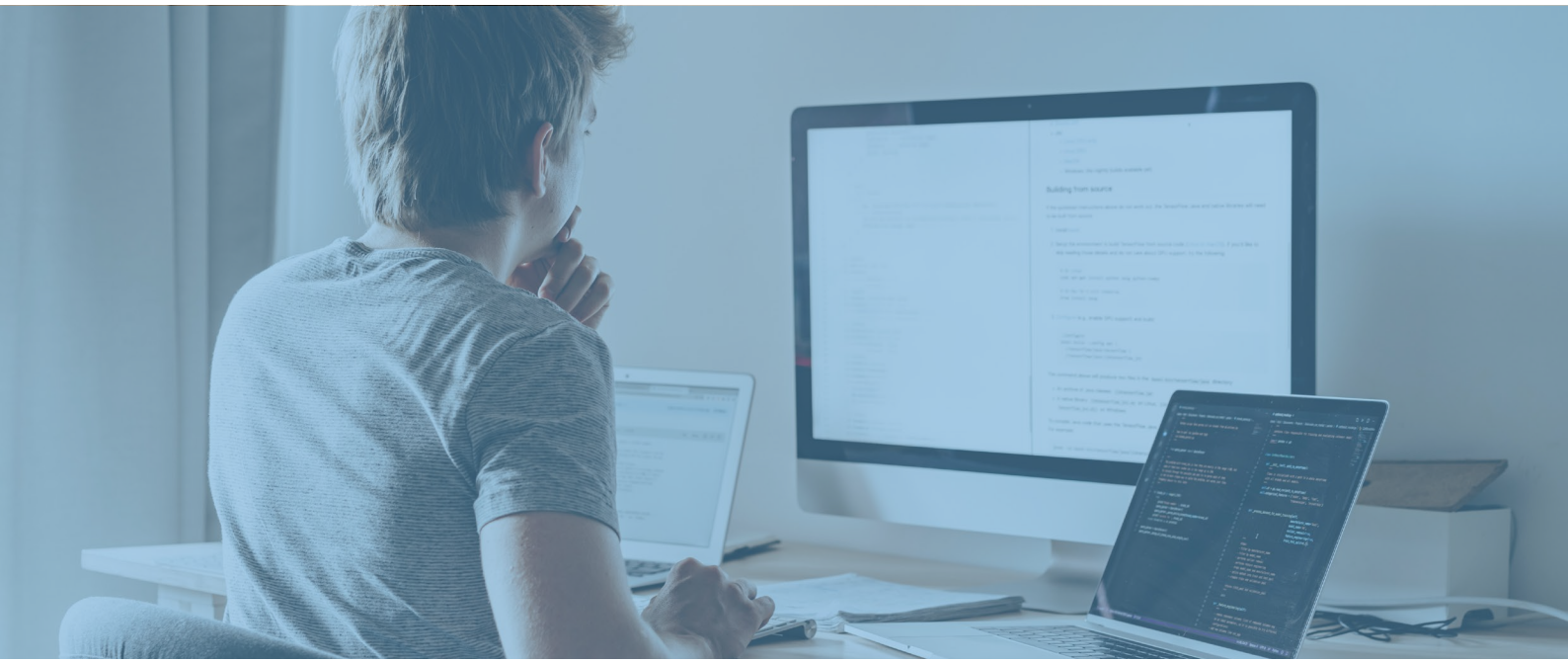
| Topic | Description | Week |
|---|---|---|
| Getting Started | Algorithms and computational problem solving; Stored program Computers; Binary representation; Programming languages; Sample python code | 1 |
| Introduction to Python | Data types in Python; Operators; Expressions; Variables and the assignment operator; Miscellaneous | 1 & 2 |
| Selection and Repetition | Motivation; Selection: if, if-else; Multi-way selection and nested structures; Repetition: while loops and counters; Miscellaneous; Examples; For loops; Boolean variables in loops; Bisection method | 2 & 3 |
| Lists, tuples, and strings | List type; Manipulating lists; Element distinctness problem; List copy: alias versus clone; Tuples; Lists and tuples with mutable objects; Sequences; List comprehension | 3 & 4 |
| Functions | Introduction; Scope of variables and execution stack; Functions handling lists, tuples, and strings; Miscellaneous; Higher-order functions; Some methods associated with list and str types | 4 & 5 |
| Files and exceptions | Files; Exceptions and assertions | 5 |
| Miscellaneous: plotting, randomness, and Monte Carlo simulation | Plotting; Generating random numbers; Monte Carlo Simulation: approximating π | 5 |
| Program Efficiency, Binary Search, and Insertion Sort | Asymptotic Analysis: Theta notation; Working with Theta; Time analysis of the element distinctness problem; Other asymptotic notations; Time analysis of programs from previous assignments; Binary search; Insertion Sort; Time analysis of some list operations and methods | 6 & 7 |

## MODULE II Recursion and elementary data structures

| Topic | Description | Week |
|---|---|---|
| Recursion I: Foundations | Introduction: recursive factorial and recursion stack; Tracking recursion, indirect recursion, infinite recursion; Two-way recursion: Fibonacci numbers; Two-way recursion: tower of Hanoi; Recursive binary search | 8 & 7 |
| Recursion II: Merge Sort, divide-and-Conquer, and solving recurrences | Merge Sort; Merge Sort time: recurrence and recursion tree method; Comparison of Merge Sort with Insertion Sort and list.sort method; Additional recursion tree examples; Master Theorem; Ternary Search | 8 & 9 |
| Data structures digression: lists of lists, -2 dimensional lists, dictionaries, and stacks | Lists of lists; Applications of 2- dimensional lists; Dictionaries; Applications of dictionaries; Stacks | 9 & 10 |
| Recursion III: Applications: Enumeration, Randomized Quick Sort, Maze DFS traversal | Application of recursion: enumeration: generate all binary strings; Another enumeration problem: generate all permutations; Quick Sort; Randomized Quick Sort; Maze DFS traversal | 10 & 11 |

**MODULE III** Object Oriented Programming and Applications

| Topic | Description | Week |
|---|---|---|
| Classes and Object Oriented Programming | Object oriented programming; User defined classes; OOP machinery; OOP concepts; Inheritance; Miscellaneous | 11 & 12 |
| Stacks and Queues | Stack class derived from list; Queue class: circular implementation | 12 & 13 |
| Graphs | Classes for dynamic directed and undirected graphs; Graph Depth First Search (DFS); Graph Breadth First Search (BFS); Other graph problems | 13 |



## GRADING*

- Assignments (10%)
- Quiz 1 (25%)
- Quiz 2 (25%)
- Final (40%)

* In the experimental offering of EECE 230X in Spring 2021-22, grading is not available to students auditing the course asynchronously from outside AUB

## SIGN UP

Register now and become a Python wizard!

CLICK TO REGISTER

+961 1 374374 ext.4478 Registration Link fswd@aub.edu.lb