Faculty of Computing and Information Technology
Department of Information Technology
Fall 2019

CPIT-110 Syllabus

CPIT-110 Problem-Solving and Programming
Credit: 3 (Theory: 3, Lab: 1.5)
Prerequisite: None
Classification: University Required

Course Objectives

The main objective of this course is to teach the students the basics of constructing algorithms and programming languages. The student at the end of this course are expected to learn the basic skills of algorithmic problem solving, the systematic approach to define the problem and creating number of solutions, and the basic programing skills which include syntax, commands, variables, selection statements, loops, functions, etc.

Class Schedule

Meet 50 minutes 3 times/week or 80 minutes 2 times/week
Lab/Tutorial 80 minutes 1 times/week

Textbook

Dr. Liang, "Introduction to Programming Using Python"
ISBN 13: 978-0-13-274718-9
ISBN 10: 0-13-274718-9

Grade Distribution

Week	Assessment	Grade %
8	Midterm Exam 1 – Part 1	15
9	Midterm Exam 1 – Part 2	15
12	Midterm Exam 2 – Part 1	15
13	Midterm Exam 2 – Part 2	15
16	Final Comprehensive Exam – Part 1	20
15	Final Comprehensive Exam – Part 2	20

* Part 1: Multiple Choice Questions, Part 2: Writing Code

Topics Coverage Durations

Topics	Weeks
Chapter 0: Introduction to Problem-Solving	1
Chapter 1: Introduction to Python	1
Chapter 2: Elementary Programming	2
Chapter 3: Mathematical Functions and String	1
Chapter 4: Selections	2
Chapter 5: Loops	2
Chapter 6: Functions	2

Course Learning Outcomes (CLO)

By completion of the course the students should be able to

1. Construct algorithms for solving simple problems.
2. Write a programing code that implements algorithms for solving simple problems.
3. Analyze and explain the behavior of simple programs involving the fundamental programming constructs.
4. Identify and describe uses of Python built-in data types and functions.
5. Write programs that use Python built-in data types and functions.
6. Apply appropriate conditional and iteration constructs for a given programming task.
7. Write and/or modify short programs that use standard conditional structures.
8. Write programs that use standard iterative control structure.
9. Write programs that use functions.
10. Trace the execution of a variety of code segments and write summaries of their computations.
11. Identify common coding errors and apply strategies for avoiding such errors.
12. Apply a variety of strategies to the testing and debugging of simple programs.
13. Use of an appropriate IDE (Integrated Development Environment) to create, compile and run a program developed by the selected programing language.