Jerash University
Faculty of Computer Science and Information Technology
Computer Sciences Department

Semester: Fall Semester 2018/2019

Course symbol and number: 1001328	**Course Name:** البرمجة المتقدّمة
Teaching Language: English	**Prerequisites:** 1001131.
Credits: 3 hours**.**	**Course Level:** 300

Course Description

This course introduces the student to the Python language. The course provides insight into the features of Python such as lists, functions, working with files, dictionaries and sets, errors and exception handlings, using modules, GUI, that make it an excellent choice for developing projects of any size.

Course Objectives

The main objectives of this course are to:

1: Familiarization with Python's Idle programming environment, working with values, variables, expressions and statements, programs as a sequence of statements, input-process-output program style, solving very small problems, handling errors in programs.

2: Writing programs that make use of conditional execution, writing functions that return values.

3: Understanding and writing recursive functions.

4: Writing programs that make use of
conditional and iterative execution.

5: Practice with drawing shapes (computer graphics and animation).

6: Practice with string operations, solving problems using string manipulation.

7: Practice with creating and manipulating lists, solving problems requiring lists.

8: Working with nested lists (lists of lists).

9: Learning about and working with dictionaries.

10: Introduce event driven Graphical User Interface (GUI) programming.

11: Further work with file processing, dictionaries, and problem-solving.

12: Writing programs that make use of object oriented concepts (Classes and objects, Classes and functions, Classes and methods, Overloading, Overriding, and Inheritance).

Learning Outcomes

Upon completion of this course, students should be able to:

1. The students will be able to develop their own programs in Python, based on a simple problem description.
2. The students will in particular be trained in using the computer to solve problems from their own life and visualize the solutions (design GUI for their solutions).
3. Students' experience in this course will put them in a good position to use the computer to solve exercises in other university courses and to exploit this experience to develop their graduation projects.
4. The students will understand the built-in objects of Python.
5. The students will understand the GUI programming.
6. The students will be able to deal with drawing and animation.
7. The students will be able to deal with lists, dictionaries, tuples and files in python.

Text Book(s)

Title	How to Think Like a Computer Scientist (Learning with Python)
Author(s)	Allen Downey, Jeffrey Elkner and Chris Meyers
Publisher	Green Tea Press (Wellesley, Massachusetts)
Year	2008
Edition	Second Edition

References

Books	• *Core Python Programming* by Wesley Chun is a large book at about 750 pages. The first part of the book covers the basic Python language features. The second part provides an easy-paced introduction to more advanced topics including many of those mentioned above. • *Python Essential Reference* by David M. Beazley is a small book, but it is packed with information both on the language itself and the modules in the standard library. It is also very well indexed. • *Python Pocket Reference* by Mark Lutz really does fit in your pocket. Although not as extensive as Python Essential Reference it is a handy reference for the most commonly used functions and modules. Mark Lutz is also the author of Programming Python, one of the earliest (and largest) books on Python and not aimed at the beginning programmer. His later book Learning Python is smaller and more accessible.
Internet links	http://www.jpu.edu.jo/lms
Course link	Click here

Instructors

Instructor	Dr. Mohammed M. Abu Shquier
Office Location	720 – الطابق السابع
Office Phone	555
E-mail	Shquier@jpu.edu.jo

Topics Covered			
Topics	**Chapters in Text**	**Week number**	**Teaching hours**
• The way of the program. • Basic concepts: program, interpreter, compiler, programming languages, solving a problem. • What is debugging? • Program errors: syntax, semantic and runtime errors. • Experimental Debugging. • Formal and natural languages. • The first program	Chapter 1	1	
• Values and Types. • Variables. • Variable Names and Keywords. • Python Statements. • Evaluating Expressions. • Operators and Operands. • Order of Operations. • Operations on Strings. • Composition. • Comments.	Chapter 2	2	
Functions • Function Calls. • Type Conversion. • Type Coercion. • Math Functions. • Composition. • Adding New Functions. • Function Definitions and Use. • Flow of Execution. • Parameters and Arguments. • Variables and Parameters are Local. • Stack Diagrams. • Functions with Results..	Chapter 3	3	
Conditionals and Recursion • The Modulus Operator. • Boolean Expressions. • Logical Operators. • Conditional Execution. • Alternative Execution. • Chained Conditionals. • Nested Conditionals. • The return Statement. • Recursion. • Stack Diagrams for Recursive Functions. • Infinite Recursion. • Keyboard Input.	Chapter 4	4	
Fruitful Functions • Return Values. • Program Development. • Composition. • Boolean Functions. • More Recursion. • Checking Types.	Chapter 5	5	
Iteration • Multiple Assignments.	Chapter 6	6	

• The while Statement. • The while Statement (Drawing Iteratively). • The while Statement (Tables). • The while Statement (2D Tables). • Encapsulation and Generalization. • Local Variables. • More Generalization.			
Strings • A compound data type. • String Length. • Traversal and the for Loop. • String Slices. • String Comparison. Strings are Immutable. • find Function. • Looping and Counting. • The string Module. • Character Classification.	Chapter 7	7	
Lists • Lists. • List Values. • Accessing Elements. • List Length. • List Membership. • List and for Loops. • List Operations. • List Slices. • Lists are Mutable. • Lists Deletion. • Objects and Values. • Aliasing. • Cloning Lists. • List Parameters. • Nested Lists. • Matrices. • Strings and Lists. • Drawing shapes.	Chapter 8	8	
Tuples • Mutability and Tuples. • Tuple Assignment. • Tuples as return Values. • Random Numbers. • List of Random Numbers. • Counting. • Many Buckets.	Chapter 9	9	
Dictionaries • Dictionaries. • Dictionary Operations. • Dictionary Methods. • Aliasing and Copying. • Sparse Matrices. • Hints. • Long Integers. • Counting Letters. • Aside.	Chapter 10	10	
Files and Exceptions • Files and Exceptions. • Text Files. • Writing Variables.	Chapter 11	11	

• Directories. Pickling. • Exceptions**.**			
GUI Programming • Graphical User Interfaces. • The main ideas. • The simplest GUI program in Python • Event-driven programming. • Terminology. • Changing the layout. • Getting input from the user. • GUI Examples**:** • Designing a GUI. • Setting up the window and widgets. • A variable for the Entry widget. • A callback for the Check button. • Defining the check function. • Improving the output. • Stylistic points.	Notes	12	
Classes and Objects • User-defined Compound Types. • Attributes. • Instances as Parameters. • Sameness. • Rectangles. • Instances as return values. • Objects are Immutable. • Copying.	Chapter 12	13	
Classes and Methods • Object-Oriented Features. • Object Oriented Examples. • Optional Arguments. • The initialization method. • Points revisited. • Operator overloading. • Polymorphism.	Chapter 13	14	
Inheritance • A simple class def: student. • Creating and Deleting Instances. • Instantiating Objects. • Constructor: • Deleting instances: No Need to “free”. • Access to Attributes and Methods. • Definition of student. • Traditional Syntax for Access. • Accessing unknown members. • getattr(object_instance, string). • hasattr(object_instance,string). • Attributes: Two Kinds of Attributes Data Attributes. • Class Attributes. • Data vs. Class Attributes. • Subclasses. • Special Data Items – Example. • Private Data and Methods.	Chapter 14	15	

Evaluation

Assessment Tool	Expected Due Date	Weight
Programming assignments and LMS		20 %
First Exam		20 %
Second Exam		20 %
Final Exam	According to the University final examination schedule	40 %

Policy

Attendance	Attendance is very important for the course. In accordance with university policy, students missing more than the allowed absence rate of total classes are subject to failure. Penalties may be assessed without regard to the student's performance. Attendance will be recorded at the beginning or end of each class.
Exams	All exams will be CLOSE-BOOK; necessary algorithms/equations/relations will be supplied as convenient.

Class Schedule & Room

Office Hours

Sun: 8 - 9
Mon: 8 - 9:30
Tues: 11- 12
Wed: 11 – 12:30

* Or by an appointment through email

Teaching Assistant

To announced later on.

Prerequisites

Prerequisites by course	1001131