QFO-AP-FI-MO02	Course Syllabus :اسم النموذج	جامعة فيلادلفيا Philadelphia University
رقم الاصدار : 1 (Revision)	الجهة المصدرة: كلية تكنولوجيا المعلومات	
التاريخ :2017/11/05	الجهة المدققة: عمادة التطوير والجودة	
عدد صفحات النموذج:		

Course Syllabus

Course Title: Advanced Object Oriented Programming	**Course code:** 721324
Course Level: 3	**Course prerequisite (s) and/or corequisite (s):** 721220
Lecture Time: 9:10-10:00	**Credit hours:** 3

Academic Staff Specifics

Name	Rank	Office	Office Hours	E-mail Address

Course module description:

Increase students' knowledge of object-oriented concepts, teach the knowledge and skills needed to develop reusable, quality programs, instruct students on the use of object-oriented design solutions and complex systems, increase students' proficiency in programming in object-oriented and procedural environments. (Python for instance).

Course objectives:

The objectives of this course are to:

1. Demonstrate an understanding of basic programming concepts including data types, variables, modularity, parameters, conditional statements, iteration, and arrays.
2. Understand and manipulate several core data structures: Lists, Dictionaries, Tuples, and Strings.
3. Understand and employ objects, functions and modularity.
4. Demonstrate an understanding of basic and some advanced issues related to writing classes and methods understand the basic ideas behind class hierarchies, polymorphism, and programming to various interfaces
5. Demonstrate methods of error handling
6. Demonstrate an understanding of basic programming of paradigms including object-oriented, imperative, and functional programming

7. Use an existing library to implement a graphical user interface and develop multithreading programming
8. Develop Python scripts for publishing on the Web.

Course/ module components

• Books (title , author (s), publisher, year of publication)

Guttag, John V. (Author)
New Delhi: PHI Learning Private Limited, 2014
ISBN : 978-81-203-4866-0
Introduction to computation and programming using python

Support material (s) (vcs, acs, etc).

Homework and laboratory guide.

Teaching methods:

Lectures, discussion groups, tutorials, problem solving, debates, etc.

Duration: 15 weeks, 45 hours in total
Lectures: 30 hours, 2 per week
Laboratories: 15 hours, 1 per week

Learning outcomes:

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

- **Knowledge and understanding**
  1. Identify object-oriented design concepts, teach the knowledge and skills (A2)
  2. Recognize and use several core data structures: Lists, Dictionaries, Tuples, and Strings. (A2)
  3. Extends students' proficiency in programming in object-oriented and procedural environment (A2)
  4. Recognize basic programming paradigms including object-oriented, imperative, and functional programming (A2)
- **Cognitive skills (thinking and analysis).**
  5. Acquire a full Object Oriented Thinking (B4)
  6. Be able to design small object oriented programs which meet requirements expressed in English. (B3)
  7. Be able to develop programs in python programming language. (B4)
- **Practical skills - able to**
  8. Use object-oriented programming techniques to design and implement a clear, well-structured Python program (C1)
  9. Apply Python Exception Handling model to develop robust programs and learn how to use it. (C5)
  10. Write and debug Python programs which make use of the fundamental control (C5)
  11. Use API libraries for python (C4)
  12. Recognize basic programming paradigms including object-oriented, imperative, and functional programming (C3)
- **Transferable skills - able to**
  13. Solve problems that have origins in a variety of paradigms including object-oriented, imperative, and functional programming. (D2)
  14. Work as part of a team (D6)

15. Be able to demonstrate communication skills (personal and academic). (D4)
16. Transfer practical and subject specific skills (Transferable Skills). (D2)

Assessment instruments

Learning outcomes (1-7) are assessed by examinations; Learning outcomes (8, 9, 10, 11, 12, 13, and 16) are assessed by examinations, tutorials, projects/assignments, by laboratory works; Learning outcomes (14, 15) are assessed by projects.

Allocation of Marks	
Assessment Instruments	**Mark**
First examination	**20**
Second examination	**20**
Final examination: 40 marks	**40**
Reports, research projects, Quizzes, Home works, Projects	**20**
Total	**100**

Assignments All assignments will be announced or handed out in class. Many assignments will require programming in Python. All individual assignments, whether programming or not, are to be done individually. While you may discuss the assignment in general terms with others, your solutions should be composed, designed, written and tested by you alone. If you need help, consult the TA or the instructor.

Documentation and academic honesty

- Documentation style (with illustrative examples)
- Protection by copyright
- Avoiding plagiarism.

Course/module academic calendar

week	**Basic and support material to be covered**	**Homework/reports and their due dates**	**Status**
(1)	**Cornerstones of Computing**		
(2)	**Getting Started in Python**		
(3)	**Elementary Control Structures**	**Assignment 1**	
(4)	**Additional Control Structures**	**Assignment 2**	
(5)	**Defining Our Own Classes**	**Assignment 3**	
(6) First examination	**Good Software Practices**	**Assignment 4**	
(7)	**Input, Output, and Files**	**Assignment 5**	
(8)	**Inheritance**	**Assignment 6**	
(9)	**Deeper Understanding of the Management of Objects**	**Assignment 7**	
(10)	**More Python Containers (**Dictionaries,…)	**Assignment 8**	
(11)	**Implementing Data Structures**	**Assignment 9**	

Second examination			
(12)	**Event-Driven Programming and API**	**Assignment 10**	
(13)	**Scripts for publishing on the Web**	**Assignment 11**	
(14)	**Functional programming**	**Assignment 12**	
(15) Specimen examination (Optional)	**A case study/Project**		
(16) Final Examination			

Expected workload:
On average students need to spend 2 hours of study and preparation for each 50-minute lecture/tutorial.

Attendance policy:
Absence from lectures and/or tutorials shall not exceed 15%. Students who exceed the 15% limit without a medical or emergency excuse acceptable to and approved by the Dean of the relevant college/faculty shall not be allowed to take the final examination and shall receive a mark of zero for the course. If the excuse is approved by the Dean, the student shall be considered to have withdrawn from the course.

Module references

Books
Students will be expected to give the same attention to these references as given to the Module textbook(s)

1. Mark Lutz and David Ascher, "Learning Python", Beijing: O'Reilly, 2004, 2nd ed..
2. Deitel, H. M, " Python : how to program ", Upper Saddle River, New Jersey: Prentice Hall, 2002
3. Dusty Phillip , "Python 3 Object Oriented Programming", Packt, July 2010. (ISBN : 1849511268 ISBN 13 : 978-1-849511-26-1)
4. Allen B. Downey, " Python for Software Design How to Think Like a Computer Scientist", Olin College of Engineering, Massachusetts, May 2009. (ISBN-13: 9780521898119)
5. Michael H Goldwasser,*Saint Louis University* , David Letscher, *Saint Louis University* , “Object-Oriented Programming in Python “, ISBN-10: 013615031, Publisher: Prentice Hall Copyright: 2008

Websites

www.python.org
http://www.learningpython.com/