Universidad de Alcalá

GUÍA DOCENTE

FUNDAMENTOS DE PROGRAMACIÓN

Grado en Ingeniería en Sistemas de Información
Grado en Ingeniería de Computadores
Grado en Ingeniería Informática
Grado en Sistemas de Información (G58)

Universidad de Alcalá

Curso Académico 2019/2020

Curso 1º – Cuatrimestre 1º

GUÍA DOCENTE

Nombre de la asignatura:	**Fundamentos de Programación**
Código:	**780003**
Titulación en la que se imparte:	**Grado en Ingeniería Informática** **Grado en Computadores** **Grado en Ingeniería en Sistemas de Información** **Grado en Sistemas de Información**
Departamento:	**Departamento Ciencias de la Computación**
Carácter:	**Básica**
Créditos ECTS:	**6**
Curso y cuatrimestre:	**Primer Curso / Primer Cuatrimestre**
Profesorado:	Salvador Sánchez Alonso Elena García Barriocanal Rosalía Peña Ros María Ángeles Fernández de Sevilla *La lista completa aparece publicada en la página web del departamento*
Horario de Tutoría:	Se indicará el primer día de clase
Idioma en el que se imparte:	Español/English Friendly

1a. PRESENTACIÓN

La asignatura pretende introducir a los alumnos en las técnicas de resolución de problemas de tipo general mediante el empleo de la programación, así como la comprensión de la teoría subyacente relacionada con el uso de lenguajes de programación de alto nivel.

Dado que la asignatura constituye el primer contacto con la programación, no se suponen conocimientos previos sobre el tema.

1b. COURSE SUMMARY

This subject aims to introduce students in the world of programming, providing the core knowledge to begin programming in any language, with no assumed previous knowledge. The “Programming Fundamentals” course is divided in 2 parts, the laboratory part provides basic skills in Python programming, while the theoretical part focuses on software design with an emphasis on understanding the theory underlying the use of programming languages.
The subject teaches and illustrates the software design process, and shows how to develop a correct, readable and reusable solution from a problem specification.

2. COMPETENCIAS

Competencias generales:

CG5 Capacidad para concebir, desarrollar y mantener sistemas, servicios y aplicaciones informáticas empleando los métodos de la ingeniería del software como instrumento para el aseguramiento de su calidad, de acuerdo con los conocimientos adquiridos según lo establecido en el apartado 5 de la resolución BOE-A-2009-12977.

CG8 Conocimiento de las materias básicas y tecnologías, que capaciten para el aprendizaje y desarrollo de nuevos métodos y tecnologías, así como las que les doten de una gran versatilidad para adaptarse a nuevas situaciones.

CG9 Capacidad para resolver problemas con iniciativa, toma de decisiones, autonomía y creatividad. Capacidad para saber comunicar y transmitir los conocimientos, habilidades y destrezas de la profesión de Ingeniero Técnico en Informática.

Competencias específicas:

CIB3 Capacidad para comprender y dominar los conceptos básicos de matemática discreta, lógica, algorítmica y complejidad computacional, y su aplicación para la resolución de problemas propios de la ingeniería.

CIB4 Conocimientos básicos sobre el uso y programación de los ordenadores, sistemas operativos, bases de datos y programas informáticos con aplicación en ingeniería.

CIB5 Conocimiento de la estructura, organización, funcionamiento e interconexión de los sistemas informáticos, los fundamentos de su programación, y su aplicación para la resolución de problemas propios de la ingeniería.

Resultados del aprendizaje

Al concluir con éxito esta asignatura, los estudiantes serán capaces de…

RA1. Desarrollar la habilidad de crear soluciones algorítmicas a problemas y ser capaz de representarlas en forma de programas de computadora.

RA2. Aplicar la estrategia de implementación descendente y el diseño modular a la construcción de programas, siguiendo los principios de máxima cohesión y mínimo acoplamiento.

RA3. Distinguir entre diferentes técnicas de desarrollo, diseño, prueba y depuración aplicadas a problemas, adquiriendo en particular una visión sistémica de la verificación y validación.

RA4. Experimentar con un lenguaje y entorno de programación de alto nivel, identificando sus capacidades y limitaciones en comparación con otros lenguajes y entornos.

RA5. Explicar los conceptos básicos de almacenamiento y representación de datos, y deducir tanto la estructura de un dato como su compatibilidad con otros en función de su tipo.

RA6. Reconocer las tareas de comprobación que se llevan a cabo durante el procesamiento de un programa, y diseñar programas teniéndolas presentes.

3. CONTENIDOS

Bloques de contenido (se pueden especificar los temas si se considera necesario)	Total de clases, créditos u horas *
Fundamentos de la Programación Introducción, sintaxis y semántica de los lenguajes de programación de alto nivel. Conceptos básicos: variables, tipos, expresiones y asignaciones. Entrada/Salida básica. Estructuras de control selectivas e iterativas. Modularización: funciones y paso de parámetros, descomposición modular de los programas.	16 horas
Metodología del desarrollo del Software y Algoritmos y resolución de problemas Estrategias de resolución de problemas: el papel de los algoritmos en el proceso de resolución de problemas, implementación de estrategias para algoritmos, pruebas y depuración del código, concepto de algoritmo y propiedades de los mismos.	14 horas
Estructuras de datos fundamentales Representación interna de los datos; tipos primitivos; tipos estructurados; definición y uso de nuevos tipos.	18 horas
Recursión Concepto, clasificación de la recursión, transformación de algoritmos recursivos, ejemplos clásicos.	8 horas

* En las horas se incluyen las PEI (Pruebas de evaluación Intermedias)

4. METODOLOGÍAS DE ENSEÑANZA-APRENDIZAJE.- ACTIVIDADES FORMATIVAS

4.1. Distribución de créditos (especificar en horas)

Número de horas presenciales:	28 horas de clase 28 horas de trabajo en laboratorio 4 horas de evaluación final (Total 60 horas)
Número de horas del trabajo propio del estudiante:	90 horas
Total horas	150 horas

4.2. Estrategias metodológicas, materiales y recursos didácticos

La asignatura se organiza como una asignatura cuatrimestral de 6 ECTS, empleándose en el proceso de enseñanza-aprendizaje de sus contenidos las siguientes actividades formativas:

- Clases Teóricas presenciales.
- Clases Prácticas presenciales.
- Prácticas en Laboratorio presenciales.
- Tutorías: individuales y/o grupales.

Además, en función de la naturaleza de las distintas partes de la materia objeto de estudio, se podrán utilizar, entre otras, las siguientes actividades formativas:

- Elaboración de trabajos individuales o en equipo.
- Puesta en común de la información, problemas y dudas que aparezcan en la realización de los trabajos.
- Utilización de la Plataforma de Aula Virtual.

Actividades presenciales:

- En el aula: Exposición y discusión de los conocimientos básicos de la asignatura. Planteamiento y resolución teórica de ejercicios y supuestos relacionados. Actividades (lecturas, discusiones, casos, etc.) orientadas a la enseñanza de las competencias específicas de la asignatura.
- En el laboratorio: Planteamiento, desarrollo y solución de ejercicios prácticos utilizando herramientas, técnicas y métodos objeto de estudio de la asignatura, contribuyendo al desarrollo de la capacidad de análisis, razonamiento crítico y comprensión de las prácticas realizadas.

Actividades no presenciales:

- Análisis y asimilación de los contenidos de la materia, resolución de problemas, consultas bibliográficas, preparación de trabajos individuales y/o grupales, realización de autoevaluaciones, sesiones de revisión post-evaluación. Todas ellas orientadas al desarrollo de métodos para la organización y planificación del trabajo personal.
- Tutorías: asesoramiento individual y/o en grupos durante el proceso de enseñanza-aprendizaje, bien en forma presencial o bien a distancia.

Recursos y materiales:

- Bibliografía de referencia.
- Entornos hardware/ software de desarrollo.
- Plataforma de Aula Virtual y manuales de uso.
- Proyectores.

5. EVALUACIÓN: Procedimientos, criterios de evaluación y de calificación

En la convocatoria ordinaria el método de evaluación por defecto es la "evaluación continua", con características de evaluación formativa para servir de realimentación en el proceso de enseñanza-aprendizaje por parte del alumno.

Opcionalmente, y de manera justificada, el alumno podrá solicitar ante el director del centro la evaluación mediante prueba única, lo que deberá de ser solicitado por escrito y en los plazos reglamentados.

Para ello se establecen los siguientes

Procedimientos de Evaluación

Criterios de Evaluación

Los Criterios de Evaluación deben atender al grado de adquisición de las competencias por parte del estudiante. Para ello se definen los siguientes criterios:

CE1: El alumno demuestra el dominio de los rudimentos básicos de la programación mediante la escritura de código simple, completo, robusto y eficiente.

CE2: El alumno demuestra aptitud en la resolución de problemas mediante programas de computadora

CE3: El alumno distingue las diferentes estructuras de control de la lógica de un programa y es capaz de utilizar las más adecuadas en cada circunstancia

CE4: A partir de la descripción de un problema complejo, el alumno es capaz de diseñar programas modulares cuya estructura viene determinada por los principios de cohesión y acoplamiento según las buenas prácticas de la programación descendente

CE5: El alumno es capaz de diseñar soluciones iterativas y recursivas para un mismo problema y distingue las ventajas e inconvenientes de cada una

CE6: El alumno es capaz de realizar un correcto análisis de las estructuras y tipos de datos necesario, así como de proponer alternativas de diseño en función de las necesidades específicas del problema

CE7: El alumno entiende un código escrito por otra persona y es capaz de modificarlo para corregirlo o para ampliar su funcionalidad.

Instrumentos de Evaluación

Esta sección indica los instrumentos de evaluación que serán aplicados a cada uno de los criterios de evaluación.

1. Prueba de Evaluación Intermedia (PEI-1): Resolución de cuestiones teórico-prácticas sobre estructuras de control, rudimentos de la programación modular y conceptos básicos.
2. Pruebas de Laboratorio (PL-1): Elaboración de programas sencillos en un lenguaje de programación.
3. Prueba de Evaluación Intermedia (PEI-2): Resolución de cuestiones teórico-prácticas sobre datos estructurados, recursividad, manejo de estructuras de datos y algoritmos de tratamiento de datos.
4. Prueba de Laboratorio (PL-2): Elaboración de programas de complejidad intermedia en un lenguaje de programación.
5. Trabajo de aula de laboratorio (TAL): Elaboración de programas propuestos en cada sesión de laboratorio, cuyo objetivo es fomentar el trabajo constante, evolutivo e incremental e incorporar paulatinamente los conceptos y competencias prácticas de la asignatura.
6. Prueba de Evaluación Final (PEF): consistente en la resolución de supuestos teórico-prácticos, así como en la solución de problemas complejos mediante el uso de programas en un lenguaje de programación.

Criterios de Calificación

Esta sección cuantifica los criterios de evaluación para la superación de la asignatura.

a) Convocatoria Ordinaria: Evaluación Continua

En la convocatoria ordinaria – evaluación continua la relación entre los criterios, instrumentos y calificación es la siguiente.

Competencia	Resultado Aprendizaje	Criterio de Evaluación	Instrumento de Evaluación	Peso en la calificación
CG8, CG9, CIB4	RA1, RA2, RA5	C1, C3, C7	PEI-1	10%
CG8, CG9, CIB4	RA1, RA2, RA4, RA6	C1, C2, C3, C7	PL-1	10%
CG5, CG8, CG9, CIB3, CIB4	RA1, RA2, RA3, RA4, RA5, RA6	C1, C2, C3, C5, C7	PEI-2	40%
CG5, CG8, CG9, CIB3, CIB4	RA1, RA2, RA3, RA4, RA6	C1, C2, C3, C4, C5, C6	PL-2	20%
CG5, CG8, CG9, CIB5, CIB4	RA1 a RA6	C1 a C7	TAL	20%

Como criterio general, aquellos alumnos en convocatoria ordinaria que se presenten a la evaluación de menos del 50% de todas las prácticas se considerarán No Presentados.

b) Convocatoria Ordinaria: Evaluación Final

Competencia	Resultado Aprendizaje	Criterio de Evaluación	Instrumento de Evaluación	Peso en la calificación
CIB3-5, CG5, CG8, CG9	RA1-RA6	C1-C7	PEF	100%

c) Convocatoria Extraordinaria

Competencia	Resultado Aprendizaje	Criterio de Evaluación	Instrumento de Evaluación	Peso en la calificación
CIB3-5, CG5, CG8, CG9	RA1-RA6	C1-C7	PEF	100%

Todas las pruebas podrán realizarse en las aulas de teoría o laboratorio, o a través del Aula Virtual, siendo la superación de las prácticas y de las actividades de la asignatura requisito necesario para la superación de la asignatura.

Como resultado del proceso de evaluación el alumno obtendrá una calificación que dependerá de su actividad en las distintas pruebas de la asignatura. El resultado de cada prueba arrojará información bien mediante indicadores cuantitativos de adquisición de competencias, bien mediante una calificación cualitativa, que a modo de orientación podrá determinarse en función del grado de dominio mostrado en las tareas propuestas por los profesores responsables de la asignatura:

Sobresaliente	Notable	Aprobado	Suspenso
Excelente dominio de los conocimientos básicos. Elaboración de ideas a partir de la reflexión y aplicación de los conocimientos adquiridos. Cumplimiento de todas las tareas programadas.	Dominio de los conocimientos básicos. Alto nivel de reflexión. Cumplimiento adecuado de la mayoría de las tareas programadas.	Domina los conocimientos básicos. Nivel medio de reflexión. Cumplimiento de un número suficiente de las tareas programadas.	Bajo nivel de comprensión y aplicación de ideas. Nivel bajo de reflexión. Falta de implicación en las tareas propuestas por el profesor.

6. BIBLIOGRAFÍA

Bibliografía Básica

- FERNÁNDEZ DE SEVILLA, M.A. Introducción práctica a la programación con Python. Ed. Servicio de publicaciones de la UAH.
- MARZAL, A. y GRACIA, I. Introducción a la programación con Python - UJI. Ed. Publicacions de la Universitat Jaume I.
- PEÑA, R. Resolución de problemas para ingenieros con Python estructurado. Ed. Garceta.
- SEVERANCE, C. Python for informatics: http://www.pythonlearn.com/book.php

Bibliografía Complementaria

a) Python

- Learn Python the hard way - http://learnpythonthehardway.org/book/
- A bite of Python - http://www.swaroopch.com/notes/python/
- Think Python - http://www.greenteapress.com/thinkpython/
- John M. Zelle (2010) Python Programming: An Introduction to Computer Science. Editorial Franklin, Beedle & Associates. 2ª ed.

b) Teoría de la programación

- GARCÍA MOLINA, F., MONTOYA DATO, J. y otros. Una Introducción a la Programación: Un enfoque algorítmico. Ed. Thomson Paraninfo. 2005.
- JOYANES, L. Fundamentos de la programación, 1ª Ed. Ed. McGraw-Hill. 1992.

Universidad de Alcalá

FUNDAMENTALS OF PROGRAMMING

Bachelor degree(s) in
Computer Engineering
Computer Science
Information Systems Engineering
Information Systems

Universidad de Alcalá

Academic Year 2019/2020

1st year – 1st semester

TEACHING GUIDE

Subject:	**Fundamentals of programming**
Code:	**780003**
Degree:	**Degree in Computer Engineering** **Degree in Informatics Engineering** **Degree in Information Systems Engineering** **Degree in Information Systems**
Department:	**Computer Science Department**
Character:	**Basic**
ECTS:	**6**
Course and semester:	**1st year – 1st semester**
Professors:	Salvador Sánchez Alonso Rosalía Peña Ros María Ángeles Fernández de Sevilla *Full list in the Department Website*
Office hours:	Each lecturer will inform on the first class
Classes offered in:	Spanish / English Friendly

1. INTRODUCTION

This subject aims to introduce students in the world of programming, providing the core knowledge to begin programming in any language, with no assumed previous knowledge. The “Programming Fundamentals” course is divided in 2 parts, the laboratory part provides basic skills in Python programming, while the theoretical part focuses on software design with an emphasis on understanding the theory underlying the use of programming languages.
The subject teaches and illustrates the software design process, and shows how to develop a correct, readable and reusable solution from a problem specification.

2. SKILLS

General skills:

CG5 Ability to conceive, develop and maintain systems, services and applications using the methods of software engineering as a tool for quality assurance, according to the knowledge acquired as provided in paragraph 5 of resolution BOE-A- 2009-12977.
CG8 Knowledge of basic materials and technologies that enable learning and development of new methods and technologies, as well as to equip them with great versatility to adapt to new situations.

CG9 Ability to solve problems with initiative, decision making, autonomy and creativity. Ability to communicate and transmit knowledge and skills of the profession of Technical Engineer.

Specific skills:

CIB3 Ability to understand and master the basics of discrete, logic, algorithmic mathematical and computational complexity, and its application for solving problems of engineering.

CIB4 Basic knowledge of the use and programming of computers, operating systems, databases and software with applications in engineering.

CIB5 Knowledge of the structure, organization, operation and interconnection of computer systems, the fundamentals of its programming and its application for solving problems of engineering.

Learning outcomes

Upon successful completion of this course, students will be able to:

- LO1. Develop the ability to create algorithmic solutions to problems and to represent them in the form of computer programs.
- LO2. Apply the top-down implementation strategy and the modular design to the construction of programs, following the principles of maximum cohesion and minimum coupling.
- LO3. Distinguish between different techniques of development, design, testing and debugging applied to problems, acquiring in particular a systemic view of verification and validation.
- LO4. Experiment with a high level programming language and environment, identifying its capabilities and limitations compared to other languages and environments.
- LO5. Explain the basic concepts of storage and representation of data, and deduce both the structure of a data and its compatibility with other data elements depending on their type.
- LO6. Recognize the verification tasks that are carried out during the processing of a program, and design programs with them in mind.

3. CONTENTS

Content modules	Total (*)

Fundamentals of programming Introduction, syntax and semantics of high level programming languages. Basic concepts: variables, types, expressions and assignments. Basic input / output. Selective and iterative control structures. Modularization: functions and step of parameters, modular decomposition of the programs.	16 hours
Software development methodology. Algorithms and problem solving strategies Problem solving strategies: the role of algorithms in the problem solving process, implementation of strategies for algorithms, code testing and debugging, the concept of algorithm and its properties.	14 hours
Fundamental data structures Internal representation of the data; primitive types; structured types; definition and use of new datatypes.	18 hours
Recursion Concept, classification of recursion, transformation of recursive algorithms, classic examples.	8 hours

* Assessment hours included

4. TEACHING AND LEARNING METHODOLOGIES – LEARNING ACTIVITIES

4.1. Credits distribution

Number of face to face hours	28 class hours 28 laboratory hours 4 hours final assessment (Total 60 hours)
Student working hours	90 hours
Total hours	150 hours

4.2. Methodological strategies, materials and educational resources

The subject is organized as a four-month subject of 6 ECTS, following these teaching activities during the teaching-learning process:

- Face-to-face theoretical classes.
- Face-to-face practical classes.

- Laboratory practices.
- Tutorials: individual and / or group.

In addition, depending on the nature of the different parts of the subject, the following training activities may be included in the process, among others:

- Preparation of individual or team work.
- Put in common the information, problems and doubts that might appear along the process.
- Use of the Virtual Classroom Platform.

Class activities:

- In the classroom: Presentation and discussion of the basic knowledge of the subject. Approach and theoretical resolution of exercises and related assumptions. Activities (readings, discussions, cases, etc.) oriented to the teaching of the specific competences of the subject.
- In the laboratory: Approach, development and solution of practical exercises using tools, techniques and methods object of study of the subject, contributing to the development of the capacity of analysis, critical reasoning and understanding of the practices carried out.

Non face to face activities:

- Analysis and assimilation of the contents of the subject, resolution of problems, bibliographical queries, preparation of individual and / or group work, self-evaluations, post-evaluation and review sessions (all of them oriented to the development of methods for the organization and planning of personal work).
- Tutorials: individual and / or group counseling during the teaching-learning process, either in person or remotely.

Resources and materials:

- Reference bibliography.
- Hardware / software development environments.
- Virtual Classroom platform and user manuals.
- Projectors.

5. ASSESSMENT: Procedures and criteria

In the ordinary call the default evaluation method is the "continuous evaluation", with characteristics of formative evaluation to serve as feedback in the teaching-learning process by the student.

Optionally, and in a justified manner, the student may opt out the default continuous evaluation by means of a formal application to the director of the Faculty, in writing form and within the prescribed deadlines.

Assessment criteria

The assessment criteria will measure the degree of acquisition of competences by the student. For this, the following criteria are defined:

- AC1: The student demonstrates the mastery of the basic rudiments of programming by writing simple, complete, robust and efficient code.
- AC2: The student demonstrates aptitude in solving problems creating computer programs
- AC3: The student distinguishes the different control structures of the logic of a program and is able to use the most appropriate in each circumstance
- AC4: From the description of a complex problem, the student is able to design modular programs whose structure is determined by the principles of cohesion and coupling according to the good practices of the top-down programming
- AC5: The student is able to design iterative and recursive solutions for the same problem and distinguishes the advantages and disadvantages of each
- AC6: The student is able to perform a correct analysis of the structures and types of data needed, as well as to propose design alternatives according to the specific needs of the problem
- AC7: The student understands a code written by another person and is able to modify it to correct it or to expand its functionality.

Assessment instruments

This section indicates the assessment instruments that will be applied to each of the evaluation criteria.

1. Intermediate Assessment Test (PEI-1): Resolution of theoretical-practical questions about control structures, rudiments of modular programming and basic concepts.
2. Laboratory Test (PL-1): Programming simple programs in a programming language.
3. Intermediate Evaluation Test (PEI-2): Resolution of theoretical-practical questions on structured data, recursion, data structures management and data processing algorithms.
4. Laboratory Test (PL-2): Elaboration of programs of intermediate complexity in a programming language.
5. Laboratory classroom work (TAL): Elaboration of proposed programs in each laboratory session, whose objective is to promote constant, evolutionary and incremental work and gradually incorporate the concepts and practical competences of the subject.

6. Final Evaluation Test (PEF): consisting of the resolution of theoretical-practical assumptions, as well as the solution of complex problems through the use of programs in a programming language.

Evaluation and Grading criteria

This section quantifies the evaluation criteria for passing the subject.

a) Ordinary Call: Continuous Evaluation

In the ordinary call - continuous assessment the relationship between the criteria, instruments and qualification is as follows.

Competency	Learning outcomes	Assessment criteria	Assessment instrument	% over total
CG8, CG9, CIB4	LO1, LO2, LO5	C1, C3, C7	PEI-1	10%
CG8, CG9, CIB4	LO1, LO2, LO4, LO6	C1, C2, C3, C7	PL-1	10%
CG5, CG8, CG9, CIB3, CIB4	LO1, LO2, LO3, LO4, LO5, LO6	C1, C2, C3, C5, C7	PEI-2	40%
CG5, CG8, CG9, CIB3, CIB4	LO1, LO2, LO3, LO4, LO6	C1, C2, C3, C4, C5, C6	PL-2	20%
CG5, CG8, CG9, CIB5, CIB4	LO1 a LO6	C1 a C7	TAL	20%

As a general criterion, those students in ordinary call who will submit to the evaluation less than 50% of all practices will be considered as Not Assessed (*No presentado*).

b) Ordinary call: Final assessment

Competency	Learning outcomes	Assessment criteria	Assessment instrument	% over total
CIB3-5, CG5, CG8, CG9	RA1-RA6	C1-C7	PEF	100%

c) Extraordinary call

Competency	Learning outcomes	Assessment criteria	Assessment instrument	% over total
CIB3-5, CG5, CG8, CG9	RA1-RA6	C1-C7	PEF	100%

All the tests will take place either in the theory or laboratory classrooms, or through the Virtual Classroom. It is mandatory to pass the laboratory practices part to pass the subject. As a result of the evaluation process the student will obtain a grade that will depend on their activity in the different tests of the subject. The result of each test will yield information either through quantitative indicators of competence acquisition, or through a qualitative qualification, which can be determined as a guide based on the

degree of proficiency shown in the tasks proposed by the professor(s) responsible for the subject:

Honours (Sobresaliente)	Remarkable (Notable)	Pass (Aprobado)	Fail (Suspenso)
Excellent domain of the basic knowledge. Elaboration of ideas based on the reflection and application of the acquired knowledge. Fulfilment of all scheduled tasks.	Mastery of basic knowledge. High level of reflection. Adequate compliance with most scheduled tasks.	Master the basic knowledge. Average level of reflection. Fulfilment of a sufficient number of scheduled tasks.	Low level of understanding and application of ideas. Low level of reflection. Lack of involvement in the tasks proposed.

6. BIBLIOGRAPHY

SEVERANCE, C. Python for informatics: http://www.pythonlearn.com/book.php

Learn Python the hard way - http://learnpythonthehardway.org/book/

A bite of Python - http://www.swaroopch.com/notes/python/

Think Python - http://www.greenteapress.com/thinkpython/

Zelle, JM. (2010) Python Programming: An Introduction to Computer Science. Editorial Franklin, Beedle & Associates. 2ª ed.