



Guía docente

Guía docente				
Datos Identificativos				2022/23
Asignatura (*)	Fundamentos de Programación I	Código		614G02004
Titulación	Grao en Ciencia e Enxeñaría de Datos			
Descriptorios				
Ciclo	Periodo	Curso	Tipo	Créditos
Grado	1º cuatrimestre	Primero	Formación básica	6
Idioma	CastellanoGallego			
Modalidad docente	Híbrida			
Prerrequisitos				
Departamento	Ciencias da Computación e Tecnoloxías da Información			
Coordinador/a	Rabuñal Dopico, Juan Ramon	Correo electrónico	juan.rabunal@udc.es	
Profesorado	Alvarellos González, Alberto José	Correo electrónico	alberto.alvarellos@udc.es	
	Cedrón Santaefemia, Francisco Abel		francisco.cedron@udc.es	
	Rabuñal Dopico, Juan Ramon		juan.rabunal@udc.es	
	Rodríguez Tajés, Álvaro		a.tajes@udc.es	
Web	campusvirtual.udc.gal			
Descripción general	Esta materia es una introducción a la programación, en la que se ve como resolver problemas usando un lenguaje de programación. En esta asignatura se ayuda al alumno a comprender los tipos y estructuras de datos básico, al mismo tiempo que se sientan las bases para diseñar correctamente un algoritmo. Para asentar los conocimientos fundamentales de la programación de una manera más rápida y óptima es necesario emplear un lenguaje que permita la puesta en práctica de los conocimientos adquiridos y que sirva de base para el buen desarrollo de un programador informático; se utilizará el lenguaje Python, tanto para las prácticas como para los ejemplos teóricos.			

Competencias / Resultados del título

Código	Competencias / Resultados del título
A4	CE4 - Conocimiento y aplicación de los fundamentos de programación y técnicas algorítmicas básicas para diseñar soluciones a problemas, utilizando los lenguajes de programación más relevantes en el ámbito de la ciencia e ingeniería de datos.
B1	CB1 - Que los estudiantes hayan demostrado poseer y comprender conocimientos en un área de estudio que parte de la base de la educación secundaria general, y se suele encontrar a un nivel que, si bien se apoya en libros de texto avanzados, incluye también algunos aspectos que implican conocimientos procedentes de la vanguardia de su campo de estudio
B5	CB5 - Que los estudiantes hayan desarrollado aquellas habilidades de aprendizaje necesarias para emprender estudios posteriores con un alto grado de autonomía
B6	CG1 - Ser capaz de buscar y seleccionar la información útil necesaria para resolver problemas complejos, manejando con soltura las fuentes bibliográficas del campo.
C1	CT1 - Utilizar las herramientas básicas de las tecnologías de la información y las comunicaciones (TIC) necesarias para el ejercicio de su profesión y para el aprendizaje a lo largo de su vida.

Resultados de aprendizaje

Resultados de aprendizaje	Competencias / Resultados del título		
Saber analizar problemas y diseñar, programar y depurar algoritmos que los resuelvan utilizando un lenguaje de programación imperativo	A4	B5 B6	C1
Conocer los aspectos básicos que conducen a un buen diseño de programas	A4	B5 B6	C1
Saber elegir y utilizar las estrategias de resolución de problemas más relevantes	A4	B1 B5 B6	C1



Conocer y saber usar lenguajes de programación de relevancia actual

A4

B1

C1

B5

B6

Contenidos

Tema

Subtema

1 CONCEPTOS BÁSICOS

1.1 Algoritmos

1.1.1 Representación de algoritmos

1.2 Programas

1.2.1 Tipos de programas

1.3 Lenguajes de programación

1.3.1 Una visión histórica

1.3.2 Clasificación de los lenguajes

1.3.3 Instrucciones más importantes

1.3.4 Propiedades de los lenguajes

1.4 Traductores

1.5. Descripción de los lenguajes

1.6 Estructura de un programa

1.7 Elementos de un programa

1.7.1 Símbolos predefinidos

1.7.2 Símbolos especiales

1.7.3 Identificadores

1.7.4 Etiquetas

1.7.5 Comentarios

1.7.6 Directivas

1.7.7 Constantes

1.7.8 Números

1.7.9 Cadenas de caracteres

1.7.10 Variables: Declaración e iniciación

1.7.11 Variables: Dirección de Memoria

1.8 Salida e Entrada

1.8.1 Sentencias de salida

1.8.2 Sentencias de entrada

1.9 Tipos de datos e operadores

1.9.1 Tipos de datos

1.9.2 Operadores

1.9.3 Expresiones

1.10 Depuración de programas



2 SENTENCIAS DE CONTROL	2.1 Secuencial 2.2 Condicional 2.2.1 La sentencia condicional simple 2.2.2 La sentencia condicional múltiple 2.3 Repetitiva 2.3.1 Introducción 2.3.2 Variables asociadas a los bucles 2.3.3 Funcionamiento de los diferentes tipos de bucles 2.3.4 Bucle FOR 2.3.5 Equivalencia entre bucles 2.3.6 Errores en los bucles 2.3.7 Diseño de bucles
3 ARQUITECTURA DE UN PROGRAMA	3.1 Funciones 3.1.1 Tipos de funciones 3.2 Función como argumentos 3.3 Corrutinas 3.4 Recursividad 3.4.1 Naturaleza de la recursividad 3.4.2 Recursión infinita
4 ESTRUCTURAS SIMPLES DE DATOS	4.1 Vectores 4.1.1 Tipo de datos 4.1.2 Declaración de un vector 4.2 Tuplas 4.2.1 Tipo de datos 4.2.2 Operaciones con tuplas 4.3 Cadenas de caracteres 4.3.1 Cadenas de longitud variable 4.4 Listas y Diccionarios
5 ALMACENAMIENTO PERSISTENTE	5.1 Ficheros 5.2 Tipos de ficheros

Planificación				
Metodologías / pruebas	Competencias / Resultados	Horas lectivas (presenciales y virtuales)	Horas trabajo autónomo	Horas totales
Sesión magistral	A4 B1 B5 B6 C1	30	30	60
Prácticas de laboratorio	A4 B1 B5 B6 C1	20	50	70
Seminario	B6 C1	8	10	18
Atención personalizada		2	0	2
(*) Los datos que aparecen en la tabla de planificación són de carácter orientativo, considerando la heterogeneidad de los alumnos				

Metodologías	
Metodologías	Descripción



Sesión magistral	En las sesiones de teoría, el profesor describe los objetivos y los contenidos de la materia, para dar una visión particular del tema a tratar y relacionado con otros dentro de la asignatura. Después se desarrolla el tema correspondiente en forma de sesión magistral, ayudándose de las herramientas técnicas disponibles, haciendo hincapié en ciertas cuestiones en las que el alumno debe profundizar en su autoaprendizaje. El objetivo es que el alumno aprenda a algoritmizar, utilizar las estructuras básicas de datos y resolver sencillos problemas de programación. Se utilizará Python como lenguaje de codificación. Las sesión magistrales pueden ser presenciales o a través de plataformas informáticas como TEAMS. También se pueden incluir vídeos explicativos de diferentes partes de los contenidos teóricos
Prácticas de laboratorio	En las sesiones de prácticas el alumno realizará programas en papel para después codificarlo en el lenguaje Python, ejecutarlo y comprobar su nivel de corrección. Los enunciados de los programas se proporcionan con suficiente antelación para que los alumnos puedan aprovechar mejor su tiempo. Es misión del profesor supervisar el código generado por el alumno para resolver dudas, corregir malos estilos de programación y corregir errores.
Seminario	En las sesiones de seminario se realizarán ejercicios y las prácticas con la finalidad de detectar en los alumnos las lagunas de conocimiento en la materia impartida hasta ese momento, y dar las explicaciones y/o referencias necesarias para enmendarlas. Las sesiones de seminario y para resolver dudas pueden ser a través de plataformas informáticas como TEAMS

Atención personalizada

Metodologías	Descripción
Sesión magistral Prácticas de laboratorio Seminario	Tanto en las sesiones magistrales como en los laboratorios de prácticas y en las sesiones de seminario se llevará una atención personalizada del alumno, en distintos niveles cuando sea posible el tipo de clase, detectando el nivel de asimilación y la comprensión de los temas explicados y las prácticas requeridas a implantar. En las sesiones de seminario y donde se puede llegar más al alumno para conocer las lagunas que presente e indicarle el camino para cubrirlas. Los alumnos que tengan matrícula a tiempo parcial deben hablar, a comienzo del curso, con los profesores encargados de su grupo.

Evaluación

Metodologías	Competencias / Resultados	Descripción	Calificación
Sesión magistral	A4 B1 B5 B6 C1	La nota de la asignatura será la suma obtenida en la Evaluación Continua (durante las 15 semanas del periodo lectivo correspondiente a la asignatura) y lo obtenido en el Examen Final. El EXAMEN FINAL constará de preguntas, cuestiones o ejercicios que el alumno tendrá que responder. Tendrá un valor de 6 puntos. El examen oficial, tanto en la primera (enero) como en la segunda (julio) oportunidad constará de preguntas, cuestiones o ejercicios que el alumno tendrá que responder. Dicho Examen Final tendrá un valor máximo de 6 puntos, que se sumarán al obtenido en la Evaluación Continua (prácticas de laboratorio).	60



Prácticas de laboratorio	A4 B1 B5 B6 C1	Durante las últimas semanas de prácticas del curso se realizará una prueba en el laboratorio usando ordenadores que tendrá un valor máximo de 4 puntos sobre la nota total del curso (puntuada sobre 10). Será necesario que el programa a realizar por el alumno en el laboratorio se ejecute de forma correcta y completa. Como posibilidad de recuperación de la Evaluación Continua (prácticas), el alumno podrá decidir ser examinado de un contenido adicional al examen final, que constará de varios problemas a desarrollar en código Python. De hacer esta recuperación se elimina la puntuación anterior automáticamente.	40
--------------------------	----------------	---	----

Observaciones evaluación

La nota final vendrá dada por la nota obtenida por la EVALUACIÓN CONTINUA y la obtenida en el EXAMEN FINAL.

Fuentes de información

Básica	- Charles Russell Severance (Autor), Fernando Tardío Muniz (Traductor) (2015). Python para informáticos: Explorando la información. . - Charles Russell Severance (2016). Python for Everybody: Exploring Data in Python 3.. Open Textbook Library - Alberto Cuevas Álvarez (2016). Python 3. Curso Práctico. Editorial RA-MA. - Mark Lutz (2013). Learning Python. Quinta edición . O'Reilly Media Inc. - Arturo Montejó Ráez, Salud María Jiménez Zafra (2019). Curso de Programación Python. Editorial Anaya
Complementaria	- Mark Summerfield (2010). Python 3.. Editorial Anaya - Sébastien Chazallet (2016). Python 3. Los fundamentos del lenguaje - 2ª edición. Ediciones-ENI - Raúl González Duque (2008). Python para todos. - John V. Guttag (2013). Introduction to Computation and Programming Using Python. The MIT Press

Recomendaciones

Asignaturas que se recomienda haber cursado previamente

Asignaturas que se recomienda cursar simultáneamente

Asignaturas que continúan el temario

Fundamentos de Programación II/614G02009

Otros comentarios



El alumno debe tener en cuenta que se debe realizar una labor autodidacta muy importante, siguiendo el siguiente esquema: Leer, atender, comprender, preguntar, estudiar y practicar. Leer: Leer el tema a tratar antes de asistir a las sesiones teóricas. ES MUY IMPORTANTE! Atender: Atender en clase, no estar solo de cuerpo presente. Comprender: Comprender lo que se les dice en las sesiones de teoría, y si no que pregunte. Preguntar: Pregunte todo lo que no comprenda, no quedar con dudas. Estudiar: Estudiar luego de las sesiones, para retener lo comprendido.

Practicar:

Haga muchos programas, los que se le pida, surjan, y otros por su cuenta, tanto en papel como en ordenador. Fundamentos de Programación I es una asignatura que no se puede aprender estudiando en dos días. El alumno debe ir madurando los conceptos, hacer sobre el papel y el ordenador muchos programas, aprendiendo también de los errores al realizarlos. Esta es una asignatura que, por medio del sistema de evaluación continua, se puede seguir de forma activa, el ritmo de las distintas sesiones teóricas y prácticas. Debe hacer caso a las indicaciones particulares de refuerzo de estudio que le señale el profesor.

(*) La Guía Docente es el documento donde se visualiza la propuesta académica de la UDC. Este documento es público y no se puede modificar, salvo cosas excepcionales bajo la revisión del órgano competente de acuerdo a la normativa vigente que establece el proceso de elaboración de guías