Guia docente				
Datos Identificativos				2022/23
Asignatura (*)	Programación I		Código	614G03006
Titulación	Grao en Intelixencia Artificial			
Descriptores				
Ciclo	Periodo	Curso	Tipo	Créditos
Grado	1º cuatrimestre	Primero	Formación básica	6
Idioma	CastellanoGallego			
Modalidad docente	Presencial			
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 Martinez Perez, Maria Rabuñal Dopico, Juan Ramon	Correo electrónico	alberto.alvarellos@udc.es maria.martinez@udc.es juan.rabunal@udc.es	
Web	campusvirtual.udc.gal			
Descripción general	Esta materia es una introducción a la programación, en la que se ve cómo resolver problemas en un lenguaje estructurado. En ella se ayuda al alumno a comprender los tipos y estructuras de datos básicos, al mismo tiempo que se sientan las bases para diseñar correctamente un algoritmo. Y para asentar los conocimientos fundamentales de la programación de forma más rápida y óptima es necesario utilizar un lenguaje que permita la puesta en práctica de los conocimientos adquiridos y sirva de base para el buen desarrollo de un programador informático; se utilizará el Lenguaje de programación Phyton, tanto para las prácticas como para los ejemplos teóricos.			

Competencias del título	
Código	Competencias del título
A2	Capacidad para resolver problemas de inteligencia artificial que precisen algoritmos, aplicando correctamente metodologías de desarrollo software y diseño centrado en usuario/a.
A3	Capacidad para comprender y dominar los conceptos básicos de lógica, gramáticas y lenguajes formales para analizar y mejorar las soluciones basadas en inteligencia artificial.
B2	Que el alumnado sepa aplicar sus conocimientos a su trabajo o vocación de una forma profesional y posea las competencias que suelen demostrarse por medio de la elaboración y defensa de argumentos y la resolución de problemas dentro de su área de estudio.
B3	Que el alumnado tenga la capacidad de reunir e interpretar datos relevantes (normalmente dentro de su área de estudio) para emitir juicios que incluyan una reflexión sobre temas relevantes de índole social, científica o ética.
B4	Que el alumnado pueda transmitir información, ideas, problemas y soluciones a un público tanto especializado como no especializado.
B5	Que el alumnado haya desarrollado aquellas habilidades de aprendizaje necesarias para emprender estudios posteriores con un alto grado de autonomía.
B6	Capacidad para concebir, redactar, organizar, planificar, y desarrollar modelos, aplicaciones y servicios en el ámbito de la inteligencia artificial, identificando objetivos, prioridades, plazos recursos y riesgos, y controlando los procesos establecidos.
B7	Capacidad para resolver problemas con iniciativa, toma de decisiones, autonomía y creatividad.
B8	Capacidad para diseñar y crear modelos y soluciones de calidad basadas en Inteligencia Artificial que sean eficientes, robustas, transparentes y responsables.
B9	Capacidad para seleccionar y justificar los métodos y técnicas adecuadas para resolver un problema concreto, o para desarrollar y proponer nuevos métodos basados en inteligencia artificial.
C2	Capacidad de trabajo en equipo, en entornos interdisciplinares y gestionando conflictos.
C3	Capacidad para crear nuevos modelos y soluciones de forma autónoma y creativa, adaptándose a nuevas situaciones. Iniciativa y espíritu emprendedor.
C6	Capacidad para integrar aspectos jurídicos, sociales, ambientales y económicos inherentes a la inteligencia artificial, analizando sus impactos, y comprometiéndose con la búsqueda de soluciones compatibles con un desarrollo sostenible.

Resultados de aprendizaje

Resultados de aprendizaje	Competencias del título		
Conocer y comprender la importancia de los objetivos de la programación. Conocer los aspectos generales sobre los lenguajes y paradigmas de la programación. Conocer pseudocódigo y la sintaxis del lenguaje Phyton utilizado para describir algoritmos y programas. Conocer los pasos para la realización de un programa y sus principales componentes. Conocer los tipos de datos básicos usando el Lenguaje Phyton. Conocer las estructuras de control de la programación estructurada y las diferencias entre ellas. Conocer todos los aspectos relacionados con la realización de funciones y procedimientos. Llevar a cabo el proceso que permite, desde la abstracción, implementar código de alta calidad. Aplicar programación modular para resolver problemas específicos en el ámbito de IA. Comprender la sintaxis y semántica del lenguaje de programación.	A2 A3	B2 B3 B5 B7 B8 B9	C3 C6
Ser capaz de realizar el seguimiento de un algoritmo (en pseudocódigo) o programa (en Lenguaje Phyton), explicar qué realiza, y encontrar posibles errores. Ser capaz de resolver pequeños algoritmos y programas. A partir del planteamiento de un problema de pequeña-mediana envergadura saber realizar el programa para resolverlo: teniendo en cuenta los objetivos de la programación estructurada. Realizar la descomposición adecuada implementando las funciones y procedimientos necesarios correctamente. Emplear un estilo de programación apropiado: saber hacer buen uso de identificadores, comentarios justos, saber establecer precondiciones y postcondiciones, saber realizar un buen diseño de las interfaces de procedimientos y funciones, saber elegir y utilizar los tipos y estructuras de datos adecuados, saber elegir y utilizar las estructuras de control convenientes. Saber hacer buen conocimiento de la parte del lenguaje que se explique. Adquirir competencias para resolver problemas de forma metodológica y práctica. Identificar y tener la capacidad para seleccionar en un entorno práctico las principales librerías, coger experiencia en la utilización de librerías, que será fundamental en el desarrollo de la profesión donde se necesite programación para la IA y Ciencia de Datos. Analizar las alternativas para afrontar un problema e identificar qué aspectos pueden abordarse, habilidad muy importante en la IA. Manejar técnicas y herramientas de prueba para asegurar la calidad de los resultados	A2 A3	B2 B3 B4 B5 B6 B7 B8 B9	C2 C3

Contenidos	
Tema	Subtema

1 CONCEPTOS BÁSICOS. PARADIGMA IMPERATIVO	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 inicialización 1.8 Salida y Entrada 1.8.1 Sentencias de salida 1.8.2 Sentencias de entrada 1.9 Tipos de datos y operadores 1.9.1 Tipos de datos 1.9.2 Operadores 1.9.3 Expresiones 1.10 Importar y usar librerías 1.11 Depuración de programas

2 SENTENCIAS DE CONTROL	2.1 Secuencial 2.2 Alternativa 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 While 2.3.5 Bucle FOR 2.3.6 Equivalencia entre bucles 2.3.7 Errores en los bucles 2.3.8 Diseño de bucles
3 ARQUITECTURA DE UN PROGRAMA	3.1 Funciones 3.1.1 Tipos de funciones 3.1.2 Función como argumentos 3.2 Recursividad 3.2.1 Naturaleza de la recursividad 3.2.2 Recursión infinita
4 ESTRUCTURAS SIMPLES DE DATOS	4.1 Lista y Tuplas 4.1.1 Tipo de datos 4.1.2 Operaciones con Listas y Tuplas 4.2 Diccionarios 4.3 Cadenas de caracteres 4.3.1 Cadenas de longitud variable
5 ENTRADA / SALIDA	5.1 Ficheros 5.2 Tipos de ficheros
6 TESTEO Y PRUEBAS DE PROGRAMAS	6.1 Depuración de programas 6.2 Detección de errores 6.2.1 En tempo de compilación 6.2.2 En tempo de ejecución

Planificación				
Metodologías / pruebas	Competéncias	Horas presenciales	Horas no presenciales / trabajo autónomo	Horas totales
Prácticas de laboratorio	A2 A3 B2 B3 B4 B5 B6 B7 B8 B9 C2 C3 C6	28	60	88
Prueba objetiva	A2 A3 B4 B5 B6 B7 B8 B9	2	2	4
Sesión magistral	A2 A3 B2 B3 B5 B7 B8 B9 C3 C6	28	28	56
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**
Prácticas de laboratorio	En las sesiones de prácticas el alumno diseñará el pseudocódigo del problema a resolver (papel o ordenador) para después codificarlo en Lenguaje Phyton, compilarlo, ejecutarlo y comprobar su nivel de corrección. Los enunciados de los programas se proporcionará con la 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, contando con que el profesor no es un compilador que busca errores.
Prueba objetiva	Para evaluar a aprendizaxe, realizarase unha proba escrita que constará de varios exercicios a realizar en linguaxe de programación Python. Realizaraxe nas fechas fixadas pola Xunta de Facultade.
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 relacionarlo con otros dentro de la asignatura Después se desarrolla el tema correspondiente en la 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 como lenguaje de codificación Phyton Las sesión magistrales pueden ser presenciales o a través de plataformas informáticas como TEAMS en casos excepcionales. También se pueden incluir vídeos explicativos de diferentes partes de los contenidos teóricos

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

Evaluación			
Metodologías	**Competéncias**	**Descripción**	**Calificación**
Prácticas de laboratorio	A2 A3 B2 B3 B4 B5 B6 B7 B8 B9 C2 C3 C6	Durante las últimas semanas con 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. Será necesario que el programa a realizar por el alumno en el laboratorio compile y ejecute de forma correcta y completa.	40
Prueba objetiva	A2 A3 B4 B5 B6 B7 B8 B9	El EXAMEN FINAL, tanto en la primera convocatoria (enero) como en la segunda oportunidad (junio/julio) constará de varias preguntas o ejercicios que el alumno tendrá que desarrollar en código Phyton, y tendrá un valor de 6 puntos.	60

Observaciones evaluación

La nota final vendrá dada por
la nota obtenida por EVALUACIÓN CONTINUA y la obtenida en el EXAMEN FINAL. El
Examen Final constará de varias preguntas y problemas a codificar en el
lenguaje de programación empleada en las sesiones prácticas
La realización fraudulenta de las pruebas o actividades de evaluación, una vez comprobada, implicará directamente la cualificación de suspenso "0" en la materia en la convocatoria correspondiente, invalidando así cualquiera cualificación obtenida en todas las actividades de evaluación de cara a la convocatoria extraordinaria.
Para los alumnos que tengan matrícula a tiempo parcial la asistencia a las clases no es obligatoria, pero sí la asistencia a la prueba en el laboratorio en la fecha establecida para la evaluación de la parte práctica de laboratorio.

Fuentes de información	
Básica	- Charles Russell Severance (Autor), Fernando Tardio Muniz (Traductor) (2015). Python para informaticos: Explorando la informacion. - Charles Russell Severance (2016). Python for Everybody: Exploring Data in Python 3. Open Textbook Library - Alberto Cuevas Alvarez (2016). Python 3. Curso Práctico. Editorial RA-MA - Mark Lutz (2013). Learning Python. Quinta edición . O?Reilly Media Inc - Arturo Montejo Ráez, Salud María Jiménez Zafra (2019). Curso de Programación Python. Editorial Anaya
Complementária	- 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
Introducción a los Computadores /614G03012
Asignaturas que continúan el temario
Programación II/614G03007
Otros comentarios
El alumno debe tener en cuenta que debe realizar una labor autodidacta muy importante, siguiendo el siguiente esquema: Leer, atender, comprender, preguntar, estudiar y practicar:- Leer: Lea el tema a tratar antes de asistir a las sesiones teóricas. ¡ES MUY IMPORTANTE!- Atender: Atienda en clase, no sólo esté de cuerpo presente.- Comprender: Comprenda lo que se le dice en las sesiones de teoría, y si no pregunte.- Preguntar: Pregunte todo lo que no comprenda, no quede con dudas.- Estudiar: Estudie después de las sesiones, para retener lo comprendido. - Practicar: Haga muchos programas, los que se le pidan, sugieran, y otros por su cuenta, tanto en papel como en el ordenador. Programación es una asignatura que no se puede aprender estudiando en dos días. El alumno debe ir madurando los conceptos, hacer sobre el papel y en la máquina muchos programas, aprendiendo también de los errores al realizarlos. Es una asignatura que, por medio del sistema de evaluación continua, se puede aprobar sin más que 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