



Guía docente				
Datos Identificativos				2014/15
Asignatura (*)	Programación I	Código	614G01001	
Titulación	Grao en Enxeñaría Informática			
Descritores				
Ciclo	Periodo	Curso	Tipo	Créditos
Grado	1º cuatrimestre	Primero	Formación Básica	6
Idioma	Castellano			
Prerrequisitos				
Departamento	Tecnoloxías da Información e as Comunicaciós			
Coordinador/a	Garcia Martin, Esteban	Correo electrónico	esteban.garcia@udc.es	
Profesorado	Arcay Varela, Bernardino Boveda alvarez, María del Carmen Castro Martinez, Alfonso Garcia Martin, Esteban Rabüñal Dopico, Juan Ramon	Correo electrónico	bernardino.arcay@udc.es carmen.boveda@udc.es alfonso.castro@udc.es esteban.garcia@udc.es juan.rabunal@udc.es	
Web				
Descripción general				

Competencias de la titulación	
Código	Competencias de la titulación
A3	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.
A4	Conocimientos básicos sobre el uso y programación de los ordenadores, sistemas operativos, bases de datos y programas informáticos con aplicación en ingeniería.
A5	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.
A13	Conocimiento, diseño y utilización de forma eficiente de los tipos y estructuras de datos más adecuados a la resolución de un problema.
B1	Capacidad de resolución de problemas
B2	Trabajo en equipo
B3	Capacidad de análisis y síntesis
B4	Capacidad para organizar y planificar
B5	Habilidades de gestión de la información
B6	Toma de decisiones
B7	Preocupación por la calidad
C3	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.
C4	Desarrollarse para el ejercicio de una ciudadanía abierta, culta, crítica, comprometida, democrática y solidaria, capaz de analizar la realidad, diagnosticar problemas, formular e implantar soluciones basadas en el conocimiento y orientadas al bien común.
C6	Valorar críticamente el conocimiento, la tecnología y la información disponible para resolver los problemas con los que deben enfrentarse.
C7	Asumir como profesional y ciudadano la importancia del aprendizaje a lo largo de la vida.
C8	Valorar la importancia que tiene la investigación, la innovación y el desarrollo tecnológico en el avance socioeconómico y cultural de la sociedad.

Resultados de aprendizaje	
Competencias de materia (Resultados de aprendizaje)	Competencias de la titulación



<p>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 Pascal ISO10206 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 Pascal ISO_10206. 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.</p>	<p>A4 A5</p>		
<p>Ser capaz de realizar el seguimiento de un algoritmo (en pseudocódigo) o programa (en Pascal ISO-10206), 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. eligiendo y utilizando los tipos y estructuras de datos adecuadas. Saber elegir y utilizar las estructuras de control y estructuras de datos convenientes. Realizar la descomposición adecuada e implementar las funciones y procedimientos necesarios correctamente. Utilizar un estilo de programación apropiado.</p> <p>Saber hacer buen uso de identificadores, los comentarios adecuados, el establecimiento de precondiciones y postcondiciones el buen diseño de las interfaces de procedimientos y funciones y saber hacer buen conocimiento de la parte del lenguaje que se explique.</p>	<p>A3 A5 A13</p>	<p>B1 B2 B3 B4 B5 B6 B7</p>	<p>C3</p>
<p>Aprendizaje autónomo Planificación de las actividades a desarrollar Capacidad de abstracción Toma de decisiones Capacidad de iniciativa y participación</p>			<p>C3 C4 C6 C7 C8</p>

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
 - 1.5.1 Notación BNF
 - 1.5.2 Diagramas de Conway
- 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
- 1.8 Salida e 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
 - 1.9.1.1 Concepto
 - 1.9.1.2 Clases de tipos
 - 1.9.1.3 Tipo entero
 - 1.9.1.4 Tipo Real
 - 1.9.1.5 Tipo Char
 - 1.9.1.6 Tipo Boolean
 - 1.9.1.7 Definición de tipos de usuario
 - 1.9.1.8 Tipo enumerado
 - 1.9.1.9 Tipo subrango
 - 1.9.1.10 Tipos Anónimos
 - 1.9.1.11 Compatibilidad de tipos
 - 1.9.2 Operadores
 - 1.9.2.1 Aritméticos



1.9.2.2 Relacionales

1.9.2.3 Lógicos

1.9.2.4 De conjunto

1.9.2.5 De cadena

1.9.2.6 Prioridad de operadores

1.9.2.7 Expresiones



2 Sentencias de control	2.1 Secuencial 2.2 Alternativa 2.2.1 La sentencia IF 2.2.2 La sentencia case 2.3 Repetitiva 2.3.1 Introducción 2.3.2 Variables asociadas a los bucles 2.3.3 Bucle WHILE 2.3.4 Ejemplos a realizar en clase 2.3.5 Bucle FOR 2.3.6 Bucle REPEAT 2.3.7 Equivalencia entre bucles 2.3.8 Ejemplos 2.3.9 Errores en los bucles 2.3.10 Diseño de bucles
3 Arquitectura de un programa	3.1 Procedimientos 3.1.1 Concepto 3.1.2 Tipos de procedimientos 3.1.3 Parámetros por valor y referencia 3.1.4 Parámetros protegidos 3.1.5 La pila de activación de procedimientos 3.1.6 Variables globales y locales: Alcance 3.1.7 Efectos laterales 3.2 Funciones 3.2.1 Concepto 3.2.2 Funciones predefinidas 3.2.3 Funciones de usuario 3.3 Recursividad 3.3.1 Naturaleza de la recursividad 3.3.2 Recursividad directa e indirecta. La directiva del lenguaje FORWARD. 3.3.3 Recursión infinita 3.3.4 Ejemplos



4 Estructuras simples de datos	<ul style="list-style-type: none">4.1 Arrays<ul style="list-style-type: none">4.1.1 Tipo de dato ARRAY4.1.2 Declaración de un Array4.1.3 Arrays de más de una dimensión4.1.4 Operaciones con Arrays4.1.5 Arrays como parámetros4.1.6 Funciones de tipo Array4.1.7 Constantes de tipo Array 4.2 Registros<ul style="list-style-type: none">4.2.1 Tipo de dato registro4.2.2 La sentencia with4.2.3 Operaciones con registros4.2.4 Registros variantes4.2.5 Registros como parámetros4.2.6 Constantes de tipo registro 4.3 Cadenas<ul style="list-style-type: none">4.3.1 Cadenas de longitud fija4.3.2 Cadenas de longitud variable 4.4 Conjuntos<ul style="list-style-type: none">4.4.1 Operaciones y relaciones entre conjuntos4.4.2 Procesamiento de conjuntos 4.5 Operaciones básicas sobre Arrays 4.6 Entrada/Salida
--------------------------------	--

Planificación

Metodologías / pruebas	Horas presenciales	Horas no presenciales / trabajo autónomo	Horas totales
Sesión magistral	30	30	60
Prácticas de laboratorio	20	50	70
Seminario	10	10	20
Atención personalizada	0		0

(*) Los datos que aparecen en la tabla de planificación són de carácter orientativo, considerando la heterogeneidad de los alumnos

Metodologías

Metodologías	Descripción
--------------	-------------



Sesión magistral	<p>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</p> <p>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.</p> <p>El objetivo es que el alumno aprenda a algorimizar, utilizar las estructuras básicas de datos y resolver sencillos problemas de programación. Se utilizará como lenguaje de codificación Pascal estándar Extendido ISO 10206</p>
Prácticas de laboratorio	<p>En las sesiones de prácticas el alumno realizará programas en papel para después codificarlo en Pascal Estandar Extendido ISO 10206, compilarlo, ejecutarlo y comprobar su nivel de corrección.</p> <p>Los enunciados de los programas se proporcionará con la suficiente antelación para que los alumnos puedan aprovechar mejor su tiempo.</p> <p>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.</p>
Seminario	<p>Las sesiones de Seminario/Tutoría se utilizarán para acercarse aún más al alumno y detectar las lagunas que presenta en los temas vistos hasta la fecha. Se fomentará la participación activa de los alumnos, la realización de pequeños ejercicios para que estas sesiones sirvan de sonda y hacer ver al alumno a que conceptos, destrezas o conocimientos debe dedicarle más tiempo de estudio y/o práctica.</p>

Atención personalizada

Metodologías	Descripción
Sesión magistral Prácticas de laboratorio Seminario	<p>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 según sea el tipo de clase, detectando el nivel de asimilación y comprensión de los temas explicados y las prácticas requeridas a implantar.</p> <p>En las sesiones de seminario en donde se puede llegar más alumno para conocer las lagunas que presente e indicarle el camino para cubrirlas.</p> <p>El tratamiento diferenciado para los alumnos que tengan matrícula a tiempo parcial será la que marque el departamento o en su defecto la Facultad.</p>

Evaluación

Metodologías	Descripción	Calificación
--------------	-------------	--------------



Sesión magistral	<p>La nota de la asignatura será la suma de lo obtenido en la Evaluación Continua y lo obtenido en el Examen Final.</p> <p>La nota de evaluación continua se divide en dos partes:</p> <p>1.- A la mitad del curso se realizará una prueba que valdrá 2 puntos.</p> <p>2.- En la última semana del curso se realiza una prueba en el laboratorio utilizando ordenadores que valdrá 3 puntos.</p> <p>El examen final constará de tres ejercicios que el alumno tendrá que desarrollar en código y tendrá un valor de 5 puntos.</p> <p>En examen extraordinario de julio constará de tres problemas a desarrollar en código con un valor de 5 puntos, que se sumarán a lo obtenido en la evaluación continua.</p> <p>Para la evaluación en la oportunidad adelantada de diciembre se realizará un único examen de tres problemas a resolver en papel y puntuando sobre 10.</p>	70
Prácticas de laboratorio	Como se indica en la parte de sesión magistral en la última semana del curso se realizará una prueba en el laboratorio usando ordenadores que tendrá un valor de 3 puntos sobre la nota total del curso.	30

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 tres problemas a codificar.

Fuentes de información

Básica	<ul style="list-style-type: none"> - ISO (1990). Extended Pascal ISO 10206. ISO - Leestma, S e Nyhoff, L.. (1999). Programación en Pascal. Madrid Prentice Hall - Carmen Bóveda, Esteban García, Alejandra Martínez (2014). Programación estructurada en un lenguaje didáctico y estándar. La Coruña , Reprografía del Noroeste - Valls, J. e Camacho, D. (2004). Programación estructurado y algoritmos en Pascal. Madrid Prentice Hall
Complementaria	- Grogono, P (). Programación en Pascal. Addison-Wesley I

Recomendaciones

Asignaturas que se recomienda haber cursado previamente

Asignaturas que se recomienda cursar simultáneamente

Asignaturas que continúan el temario

Otros comentarios



El alumno debe tener en cuenta que debe realizar una labor autodidacta muy importante. Siga el siguiente esquema: Leer, atender, comprender, preguntar, estudiar y practicar.

Leer: Lea el tema a tratar antes de asistir a las sesiones teóricas. Aunque le parezca raro ES MUY IMPORTANTE. Atender: Atienda en clase, no se duerma, no se dedique sólo a tomar apuntes.

Comprender: Comprenda lo que se le dice en las sesiones de teoría, y si no

pregunte. Preguntar: Pregunte todo lo que no comprenda, tiene ese derecho,

Estudiar: para retener lo comprendido.

Practicar: Haga muchos programas, los que se le pidan y otros por su cuenta, tanto en papel como en el ordenador.

Programación es una asignatura que no se puede aprender con un empacho de estudio en dos días. El alumno debe ir madurando los conceptos, y hacer sobre el papel y en la máquina muchos programas.

Es una asignatura que, por medio del sistema de evaluación continua, se puede aprobar sin más que seguir el ritmo de las distintas clases de sesiones teóricas y 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