UAB
Universitat Autònoma de Barcelona

Metodología de la Programación

Código: 102764
Créditos ECTS: 6

2024/2025

Titulación	Tipo	Curso
2502441 Ingeniería Informática	FB	1

Contacto

Nombre: Ernest Valveny Llobet

Correo electrónico: ernest.valveny@uab.cat

Equipo docente

Miguel Hernández Cabronero

Miguel Eduardo Carpio Miranda

Francisco Javier Massó López

Carles Ros Visus

Marc Ortega Gil

Mireia Bellot Garcia

Idiomas de los grupos

Puede consultar esta información al final del documento.

Prerrequisitos

La asignatura no tiene ningún prerrequisito oficial. De todas formas, se asume que el estudiante ha cursado la as

Objetivos y contextualización

Esta asignatura forma parte de la materia Informática y debe verse como la continuación lógica de la parte de programación de la asignatura Fundamentos de Informática. El objetivo básico es profundizar en las nociones básicas de programación imperativa introducidas en Fundamentos de Informática e introducir los principios de la programación orientada a objetos.

De esta forma, los objetivos formativos que se proponen para la asignatura son los siguientes:

- Entender el ciclo de vida del software: análisis (entender lo que se nos pide), diseño (proponer una solución al problema), implementación (codificación en un lenguaje de programación de la solución

elegida), prueba (realización de un test de manera sistemática para asegurar la corrección de la solución implementada).

- Entender el concepto de algoritmo como herramienta de resolución de problemas con el ordenador y aprender los conceptos fundamentales de la algorítmica.
- Comprender las principales estructuras de la programación imperativa y utilizarlas correctamente para resolver problemas algorítmicos de cierta complejidad.
- Conocer las diferentes estructuras para representar información dentro de los algoritmos, tanto los tipos de datos estáticos (tablas, registros y cadenas de caracteres) como los tipos de datos dinámicos (pilas, colas, listas), para poder utilizar la estructura de datos más adecuada para representar la información asociada un problema algorítmico.
- Entender y aplicar correctamente los principios básicos de la programación orientada a objetos: concepto de clase y encapsulamiento de datos.
- Dotar al alumno de la capacidad de diseño de algoritmos para la resolución de problemas complejos, introduciendo demanera progresiva y sistemática una metodología rigurosa y estructurada de programación, basada fundamentalmente en la técnica del diseño descendente de algoritmos.
- Programar en un lenguaje de programación real y conocer las diferentes fases de desarrollo de un programa: escritura, compilación y montaje y ejecución y prueba.
- Desarrollar los programas siguiendo unas normas de estilo tendentes a conseguir programas de calidad. Dentro de estas normas de estilo engloban aquellas que facilitan la comprensión del código, como pueden ser el uso de comentarios, la indentación del código, la utilización de nombres adecuados para variables y funciones, etc.

Competencias

- Adquirir hábitos de pensamiento.
- Adquirir hábitos de trabajo personal.
- 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.
- 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.
- Conocimientos básicos sobre el uso y programación de los ordenadores, sistemas operativos, bases de datos y programas informáticos con aplicación en ingeniería.

Resultados de aprendizaje

1. Comprender y dominar la algorítmica, la complejidad computacional y su aplicación a la resolución de problemas.
2. Conocer los conceptos básicos de la estructura y programación de los computadores.
3. Conocer y saber utilizar los sistemas operativos, bases de datos y programas de uso común en la ingeniería.
4. Demostrar capacidad para la organizar la información en ficheros y bases de datos.
5. Demostrar capacidad para la representación de la información y la programación de ordenadores.
6. Demostrar la capacidad para diseñar algoritmos y analizar su complejidad computacional.
7. Desarrollar un pensamiento y un razonamiento crítico.
8. Gestionar el tiempo y los recursos disponibles. Trabajar de forma organizada.
9. Reconocer e identificar los métodos, sistemas y tecnologías propios de la ingeniería informática.
10. Tomar decisiones propias.

Contenido

Tema 0: Repaso de las estructuras de programación básicas

- Repaso de las estructuras de programación básicas: estructuras condicionales e iterativas, funciones y procedimientos, tablas, matrices, registros.

Tema 1: Introducción a la programación orientada a objetos

- Introducción al concepto de clase. Métodos y atributos. Parte privada y pública.
- Constructores y destructores.
- Encapsulación de datos.
- Composición.
- Persistencia y serialización de objetos. Ficheros.

Tema 2: Estructuras de datos dinámicas

- El concepto de apuntador. Operaciones con apuntadores.
- Objetos dinámicos.
- Arrays dinámicos
- Representación e implementación de estructuras de datos dinámicas: listas, pilas y colas.
- Utilización de estructuras de datos dinámicas.
- Análisis de costes y complejidad

Actividades formativas y Metodología

Título	Horas	ECTS	Resultados de aprendizaje
Tipo: Dirigidas			
Sesiones presenciales	50	2	1, 2, 4, 5, 6, 7, 9
Tipo: Supervisadas			
Seguimiento de la implementación del proyecto de programación	1	0,04	6, 7, 8, 10
Tipo: Autónomas			
Estudio individual	11	0,44	2, 9
Implementación del proyecto de programación	48	1,92	3, 4, 5, 6, 8, 10
Resolución de problemas	36	1,44	4, 5, 6, 8

La metodología docente de la asignatura parte del principio que "programar es la única forma de aprender a programar" y, por tanto, estará centrada principalmente en el trabajo práctico del estudiante. Las sesiones presenciales de clase se organizarán para introducir los contenidos teóricos de la asignatura, desde una perspectiva muy práctica a partir de ejemplos y de ejercicios y problemas de programación que se deberán resolver en clase directamente con el ordenador. Por otra parte, se realizará un proyecto de programación que se tendrá que ir desarrollando de forma principalmente autónoma durante todo el curso (con seguimiento y control por parte del profesor en sesiones puntuales) y que supondrá integrar de forma práctica casi todos los conceptos y herramientas de programación introducidos en las sesiones presenciales en la resolución de un problema real complejo. Además, se propondrá un conjunto de ejercicios que deben ser resueltos de forma individual a lo largo del curso (algunos de los cuales se evaluarán y discutirán en las sesiones presenciales) que deben servir para comprender, integrar y aplicar los conceptos desarrollados en las sesiones presenciales. En todas las actividades del curso (sesiones presenciales, problemas y prácticas) se utilizará el lenguaje de programación C ++.

A nivel presencial, la asignatura no distinguirá entre sesiones de teoría, problemas y prácticas. Las sesiones presenciales se organizarán en cuatro horas semanales en grupos de alrededor de 40 estudiantes. En las sesiones presenciales se irán trabajando los conceptos que se detallan en el temario de la asignatura. En algunos casos, se podrán poner a disposición del estudiante vídeos explicativos que el estudiante deberá visionar antes de la sesión de clase. Las sesiones de clase tendrán un enfoque bastante práctico con ejemplos y ejercicios que se plantearán a los alumnos para facilitar la comprensión y aprendizaje de los conceptos explicados. Estos ejercicios se realizarán y discutirán durante la sesión y servirán para ir introduciendo los contenidos de la asignatura y ver su aplicación práctica. Se recomienda que el alumno lleve su propio portátil a clase si dispone de uno, para poder realizar los ejercicios que se propongan en clase.

El estudiante deberá completar las clases presenciales con el trabajo personal autónomo en la realización de los ejercicios que se vayan proponiendo y que deben servir para acabar de entender los contenidos de la asignatura. Hay que tener presente que el temario de la asignatura tiene una continuidad lógica a lo largo del curso, por lo que para poder seguir correctamente una clase necesario haber asimilado lo explicado en las sesiones anteriores. Algunos de estos ejercicios se deberán entregar de forma individual como parte de la evaluación de la asignatura.

Además, los estudiantes deberán hacer en grupos de 2 un proyecto de programación que se desarrollará de forma autónoma durante todo el curso fuera de las sesiones presenciales. El proyecto de programación permitirá abordar un problema de programación de cierta complejidad que integre la mayoría de los conceptos explicados durante el curso. Durante el curso, se dedicarán algunas sesiones presenciales el control, seguimiento y evaluación del trabajo realizado por el alumno en el proyecto de programación.

La gestión de la docencia la asignatura se hará a través de la plataforma Caronte (http://caronte.uab.cat/), que servirá para poder ver los materiales, gestionar los grupos de prácticas, hacer las entregas correspondientes, ver las notas, comunicarse con los profesores, etc.

Competencias Transversales

• T01.01 Desarrollar un pensamiento y razonamiento crítico. Esta competencia se desarrollará durante las sesiones de teoría y problemas, a partir de la exposición y discusión de ejemplos y casos prácticos. Se tendrá en cuenta en la evaluación de la exposición de los estudiantes a las sesiones de problemas.

• T02.03 Gestionar el tiempo y los recursos disponibles. Trabajar de forma organizada. Esta competencia se desarrollará principalmente a partir del proyecto de programación. El estudiante tiene como objetivo desarrollar el proyecto de forma autónoma y debe ser capaz de organizarse el tiempo y los recursos para lograr este objetivo. Se evaluará en las sesiones de seguimiento del proyecto.

• T02.05 Tomar decisiones propias. Esta competencia se desarrollará durante el proyecto de programación en que el estudiante deberá elegir y tomar las mejores opciones para conseguir completar el proyecto. Se evaluará en las sesiones de seguimiento del proyecto.

Nota: se reservarán 15 minutos de una clase dentro del calendario establecido por el centro o por la titulación para que el alumnado rellene las encuestas de evaluación de la actuación del profesorado y de evaluación de la asignatura o módulo.

Evaluación

Actividades de evaluación continuada

Título	Peso	Horas	ECTS	Resultados de aprendizaje
Entrega de problemas	30%	0	0	1, 4, 5, 6, 7, 8
Evaluación individual	40%	4	0,16	2, 9

Proyecto de programación	30%	0	0	1, 3, 4, 5, 6, 8, 10

La evaluación de la asignatura tendrá en cuenta tres tipos de actividades de evaluación: entrega de problemas, evaluación individual y proyecto de programación. La nota final de la asignatura se obtiene combinando la evaluación de estas 3 actividades de la siguiente manera:

Nota Final = (0.3 * Evaluación continua) + (0.3 * Proyecto) + (0.4 * Evaluación Individual)

- Evaluación continua: en este apartado se incluye la entrega de los ejercicios que se propongan a lo largo del curso y otras actividades que se realicen en las sesiones presenciales o se propongan para entregar en la plataforma Caronte.
- Se deberá alcanzar una nota mínima de 4 en esta actividad para poder aprobar la asignatura.
- Los ejercicios que no se entreguen en el plazo fijado o que estén suspendidos se podrán recuperar con una penalización en la nota. La forma de recuperar estas actividades se explicará al principio del curso.

- Evaluación individual: en este apartado se incluye el resultado de las pruebas individuales que se harán a lo largo del curso. Habrá dos pruebas parciales que se realizarán durante el período lectivo del curso en horarios de clase y una prueba final durante el periodo oficial de exámenes. Esta prueba final será de recuperación y sólo la tendrán que hacer los estudiantes que no hayan superado alguno de los dos parciales. Si se ha superado uno de los dos parciales, pero el otro no, en esta prueba sólo se ha de recuperar la parte de la asignatura correspondiente al parcial que no se haya superado.
    - Deberá obternerse una nota mínima de 4,5 en cada uno de los dos parciales para poder aprobar la asignatura.
    - La nota final será la media de los dos parciales:

  Evaluación Individual = (0.5 * Parcial1) + (0.5 * Parcial 2)

- Deberá obtenerse una nota mínima de 5 en la nota de la evaluación individual para poder aprobar la asignatura.

- Proyecto: incluye la evaluación todo el trabajo del proyecto de programación: las dos entregas del proyecto (una entrega parcial a mitad de curso y la entrega final) y la evaluación del seguimiento del proyecto que se hará a lo largo del curso.
- Deberá obtenerse una nota mínima de 5 en la entrega final del proyecto para poder aprobar el proyecto.
- La nota de la entrega final del proyecto se podrá recuperar si la nota del proyecto es >= 3. En la recuperación del proyecto, la nota máxima que se puede obtener es un 7.

Evaluación única

Los estudiantes que se quieran acoger a la evaluación única tendrán que entregar en la fecha inidicada para la evaluación única los ejercicios evaluables se indiquen al principio de curso y el proyecto de programación y deberán realizar una prueba de evaluación individual que incluirá los contenidos de todo el curso. Esta prueba de evaluación individual constará de dos partes que se corresponderán a los dos parciales que se realizarán a lo largo del curso y se tendrá que conseguir un mínimo de 4,5 en cada una de las dos partes para poder aprobar la asignatura.

La nota final en el caso de evaluación única se calculará deesta forma:

Nota Final = (0.2 * Entrega problemas) +(0.3 * Proyecto) + (0.5 * Evaluación Individual)

Las notas mínimas de cada parte para poder aprobar la asignatura son las mismas que para la evaluación continua.

Se aplicará el mismo sistema de recuperación que por la evaluación continua

No evaluable: Un alumno se considerará no evaluable (NA) si no hace como mínimo el 50% de las entregas de ejercicios y no hace ninguna de las pruebas de evaluación: parcial 1, parcial 2, prueba final de recuperación, entrega final de la práctica.

Suspendidos: Si el cálculo de la nota final es igual o superior a 5 pero no se llega al mínimo exigido en alguna de las actividades de evaluación, la nota final será suspenso y se pondrá un 4.5 en la nota del expediente del alumno.

Matrículas de honor: Otorgar una calificación de matrícula de honor es decisión del profesoradoresponsable de la asignatura. La normativa de la UAB indica que las MH sólo se podrán conceder a estudiantes que hayan obtenido una calificación final igual o superior a 9.00. Se puede otorgar hasta un 5% de MH del total de estudiantes matriculados. El criterio para atorgar las MH será la nota final de la asignatura.

Convalidaciones: Para los alumnos repetidores se podrá convalidar la nota del proyecto del año anterior si se cumplen estas condiciones:

- La nota final del proyecto del curso anterior es mayor o igual a 7
- La nota de la evaluación individual del curso anterior es mayor o igual a 3

Si se convalida, la nota del proyecto será un 5, independientement de la nota del curso anterior.

Revisiones de notas: Para cada actividad de evaluación, se indicará un lugar, fecha y hora de revisión en la que el estudiante podrá revisar la actividad con el profesor. En este contexto, se podrán hacer reclamaciones sobre la nota de la actividad, que serán evaluadas por el profesorado responsable de la asignatura. Si el estudiante no se presenta a esta revisión, no podrá se revisarposteriormente esta actividad.

Nota importante: copias y plagios

Sin perjuicio de otras medidas disciplinarias que se estimen oportunas, y de acuerdo con la normativa académica vigente, las irregularidades cometidas por un estudiante que puedan conducir a una variación de la calificación se calificarán con un cero (0). Las actividades de evaluación calificadas de esta forma y por este procedimiento no serán recuperables. Si es necesario superar cualquiera de estas actividades de evaluación para aprobar la asignatura, esta asignatura quedará suspendida directamente, sin oportunidad de recuperarla en el mismo curso. Estas irregularidades incluyen, entre otros:

- la copia total o parcial de una práctica, informe, o cualquier otra actividad de evaluación;
- dejar copiar;
- presentar un trabajo de grupo no hecho íntegramente por los miembros del grupo;
- presentar como propios materiales elaborados por un tercero, aunque sean traducciones o adaptaciones, y en general trabajos con elementos no originales y exclusivos del estudiante;
- tener dispositivos de comunicación (como teléfonos móviles, smart watches, etc.) accesibles durante las pruebas de evaluación teórico-prácticas individuales (exámenes);
- hablar con compañeros durante las pruebas de evaluación teórico-prácticas individuales (exámenes);
- copiar o intentar copiar de otros alumnos durante las pruebas de evaluación teórico-prácticas (exámenes);
- usar o intentar usar escritos relacionados con la materia durante la realización de las pruebas de evaluación teórico-prácticas (exámenes), cuando éstos no hayan sido explícitamente permitidos.

En estos casos, la nota numérica del expediente será el valor menor entre 3.0 y la media ponderada de las notas (y por tanto no será posible el aprobado por compensación).

En la evaluación de las entregas de problemas y prácticas se utilizarán herramientas de detección de copia del código del programa.

Nota sobre la planificación de las actividades de evaluación: Las fechas de evaluación continua y entrega de trabajos se publicarán a principio de curso y pueden estar sujetos a cambios de programación por motivos de adaptación a posibles incidencias. Siempre se informará en Caronte sobre estos cambios ya que se entiende que esta es la plataforma habitualde intercambio de información entre profesores y estudiantes.

Bibliografía

- http://www.cplusplus.com/ : The C++ Resources Network
- https://es.wikibooks.org/wiki/Programaci%C3%B3n_en_C%2B%2B: Programación en C++ - Wikilibros
- https://www.sololearn.com/: SoloLearn
- L. Joyanes, I. Zahonero: Programación en C: metodología, estructura de datos y objetos, Mc Graw-Hill, 2001.
- B. Eckel. Thinking in C++, Volume 1: Introduction to Standard C++, Prentice-Hall, 1999.
- B. Eckel. Thinking in C++, Volume 2: Standard Libraries and Advanced Topics, Prentice-Hall, 1999.
- E. Valveny, R. Benavente, A. Lapedriza, M. Ferrer, J. García: Programació en Llenguatge C. Amb 56 problemes resolts i comentats. Servei publicacions UAB, 2009.
- L. Joyanes, A. Castillo, L. Sánchez, I. Zahonero: Programación en C: libro de problemas, Mc Graw-Hill, 2002.
- B.W. Kernighan, D.M. Ritchie: El lenguaje de programación C. 2ª Edición, Prentice Hall, 1986.
- B.W. Kernighan, R. Pike: La Práctica de la Programación. Pearson Educación, 2000.
- L. Joyanes Aguilar : Fundamentos de Programación: Algoritmos, Estructuras de Datos y Objetos. 3ª Edición, Mc. Graw-Hill, 2003

Software

Cualquier entorno de desarrollo en C++.

Lista de idiomas

Nombre	Grupo	Idioma	Semestre	Turno
(PAUL) Prácticas de aula	411	Catalán	segundo cuatrimestre	manaña-mixto
(PAUL) Prácticas de aula	412	Catalán	segundo cuatrimestre	manaña-mixto
(PAUL) Prácticas de aula	431	Catalán	segundo cuatrimestre	manaña-mixto
(PAUL) Prácticas de aula	432	Catalán	segundo cuatrimestre	manaña-mixto
(PAUL) Prácticas de aula	451	Catalán	segundo cuatrimestre	tarde
(PAUL) Prácticas de aula	452	Catalán/Español	segundo cuatrimestre	tarde
(PAUL) Prácticas de aula	471	Catalán	segundo cuatrimestre	tarde
(PAUL) Prácticas de aula	472	Catalán	segundo cuatrimestre	tarde