Guía docente
804227 - PROG2VJ - Programación II

Última modificación: 18/05/2026

Unidad responsable: Centro de la Imagen y la Tecnología Multimedia
Unidad que imparte: 804 - CITM - Centro de la Imagen y la Tecnología Multimedia.

Titulación: GRADO EN DISEÑO Y DESARROLLO DE VIDEOJUEGOS (Plan 2014). (Asignatura obligatoria).

Curso: 2026 **Créditos ECTS:** 6.0 **Idiomas:** Catalán, Inglés

PROFESORADO

Profesorado responsable: Mateo, Ramon

Otros: Mateo, Ramon

METODOLOGÍAS DOCENTES

Las clases de aprendizaje dirigido estructuran en sesiones de dos horas. Durante parte de las sesiones, el profesor expone los conceptos teóricos y lo ejemplifica mediante ejemplos que se resuelven, en lo posible, de forma participativa por parte de los estudiantes. Otra parte de la sesión se dedica a que los estudiantes practiquen los conceptos introducidos resolviendo una serie de ejercicios propuestos por el profesorado y, cuando proceda, también se dedicará tiempo para la resolución de dudas y problemas con los que se hayan encontrado durante la realización de los ejercicios.

Se hará un uso intensivo del campus virtual, tanto para publicar el material de la asignatura (apuntes, enunciados de problemas, soluciones propuestas, recopilación de links, etc.) como mecanismo de comunicación para publicar avisos, pedir las revisiones de las diferentes pruebas, etc.

OBJETIVOS DE APRENDIZAJE DE LA ASIGNATURA

- Distinguir los conceptos relacionados con la programación en el ámbito del diseño y desarrollo de videojuegos: algoritmo, tipo, valor, variable, recursividad, interpretación y compilación.

- Aplicar funciones y estructuras de datos cumpliendo ciertas especificaciones: pila, cola, lista enlazada.

- Aplicar soluciones recursivas a problemas sencillos.

- Implementar objetos sencillos con clases a partir de una especificación.

- Desarrollar programas capaces de: realizar acciones de entrada/salida línea a línea sobre ficheros de texto, usar estructuras de control, de repetición y de selección, manejar cadenas, vectores, registros y matrices, y usar flujos de datos.

HORAS TOTALES DE DEDICACIÓN DEL ESTUDIANTADO

Tipo	Horas	Porcentaje
Horas grupo mediano	16,0	10.67
Horas aprendizaje autónomo	90,0	60.00
Horas grupo grande	24,0	16.00
Horas actividades dirigidas	20,0	13.33

Dedicación total: 150 h

CONTENIDOS

1. Punteros y gestión de memoria

Descripción:
Descripción:
· Punteros
· Paso de parámetros (por valor y referencia)
· Gestión de la memoria dinámica: reservar y liberar memoria

Dedicación: 20h
Grupo grande/Teoría: 4h
Actividades dirigidas: 4h
Aprendizaje autónomo: 12h

2. Programación orientada a objetos

Descripción:
Descripción:
· Clases, encapsulación y diseño modular
· Sobrecarga de operadores
· Herencia
· Polimorfismo
· Friendship y templates

Dedicación: 45h
Grupo grande/Teoría: 8h
Actividades dirigidas: 10h
Aprendizaje autónomo: 27h

3. Estructuras de datos

Descripción:
Descripción:
· Strings
· Pilas y colas
· Listas
· Vectores dinámicos
· Árboles

Dedicación: 40h
Grupo grande/Teoría: 8h
Actividades dirigidas: 8h
Aprendizaje autónomo: 24h

4. Recursividad y Fractales

Descripción:
Estructura de los algoritmos recursivos
Recursividad en series numéricas
Las Torres de Hanoi
Fractales: El conjunto de Cantor y el triángulo de Sierpinski

Dedicación: 20h
Grupo grande/Teoría: 4h
Actividades dirigidas: 4h
Aprendizaje autónomo: 12h

5. Algoritmos de ordenación

Descripción:
Descripción:
· Bubblesort
· Quicksort
· Binary Tree
· Heap Sort
· Merge Sort

Dedicación: 25h
Grupo grande/Teoría: 6h
Actividades dirigidas: 4h
Aprendizaje autónomo: 15h

ACTIVIDADES

Actividades sobre punteros y memoria

Descripción:
Dos conjuntos de preguntas y pequeños ejercicios permitirán practicar con los conceptos de apuntadores y gestión de la memoria.
El primer conjunto de ejercicios corresponde a la implementación y manejo de variables básicas y apuntadores hacia ellas, modificando tanto su dirección como el contenido de la memoria donde apuntan.
El segundo conjunto de ejercicios incorporará estructuras, definiendo variables que sean instancias de las estructuras o apuntadores a ellas, permitiendo así empezar a trabajar con operadores de acceso a los miembros de la estructura, que después seran utilizados per los objetos y sus miembros

Objetivos específicos:
Practicar con los contenidos teóricos expuestos y coger práctica con la implementación de programas en C++

Material:
Enunciado de los ejercicios y MS Visual C++

Entregable:
El resultado de la actividad corresponde a un conjunto de ficheros de código C++ que se tendrán que enviar al espacio de entregas del campus y eventualmente presentalos delante del grupo del aula. Estas entregas y presentaciones serviran para evaluar el 2% de la nota correspondiente a la participación del estudiante.

Dedicación: 8h
Grupo mediano/Prácticas: 4h
Aprendizaje autónomo: 4h

Actividades de Fundamentos de Programación Orientada a Objetos

Descripción:
Conjunto de ejercicios de programación para practicar los conceptos de encapsulamiento, herencia y polimorfismo. Entre otros, se trabajarán ejercicios para representar y manipular burbujas, cápsulas, vehículos, Pokemons o vectores.

Objetivos específicos:
Practicar con los contenidos teóricos expuestos y coger práctica con la implementación de programas en C++

Material:
Enunciado de los ejercicios y MS Visual C++

Entregable:
El resultado de la actividad corresponde a un conjunto de ficheros de código C++ que se tendrán que enviar al espacio de entregas del campus y eventualmente presentalos delante del grupo del aula. Estas entregas y presentaciones serviran para evaluar el 2% de la nota correspondiente a la participación del estudiante.

Dedicación: 18h
Grupo mediano/Prácticas: 8h
Aprendizaje autónomo: 10h

Actividades de Estructuras de Datos

Descripción:
Cuestionario tipo concurso en el aula con preguntas para repasar los conceptos expuestos
Conjunto de ejercicios de programación para implementar los objetos asociados a las estructuras básicas y practicar su uso con algunos ejemplos. Se tratarán: i) Pilas, ii) Colas, iii) Listas, iv) Arrays dinámicos, v) Matrices y vi) Árboles.
Estos ejercicios servirán para reforzar el aprendizaje de programación orientada a objetos.

Objetivos específicos:
Practicar con los contenidos teóricos expuestos y coger práctica con la implementación de programas en C++
Ser capaz de identificar las estructuras de datos adecuadas para resolver diferentes problemas y situaciones

Material:
Enunciado de los ejercicios y MS Visual C++

Entregable:
El resultado de la actividad corresponde a un conjunto de ficheros de código C++ que se tendrán que enviar al espacio de entregas del campus y eventualmente presentalos delante del grupo del aula. Estas entregas y presentaciones serviran para evaluar el 2% de la nota correspondiente a la participación del estudiante.

Dedicación: 18h
Grupo mediano/Prácticas: 10h
Aprendizaje autónomo: 8h

Actividades de recursividad y fractales

Descripción:
Practicar y resolver algunos problemas y juegos intrínsecamente recursivos
Conjunto de ejercicios de programación para implementar las soluciones a los problemas y juegos practicados, incluyendo series numéricas, las torres de Hanoi y fractales

Objetivos específicos:
Asimilar el concepto de programación recursiva
Ser capaz de implementar algoritmos recursivos

Material:
Enunciado de los ejercicios y MS Visual C++

Entregable:
El resultado de la actividad corresponde a un conjunto de ficheros de código C++ que se tendrán que enviar al espacio de entregas del campus y eventualmente presentalos delante del grupo del aula. Estas entregas y presentaciones serviran para evaluar el 2% de la nota correspondiente a la participación del estudiante.

Dedicación: 8h
Grupo mediano/Prácticas: 4h
Aprendizaje autónomo: 4h

Actividades de algoritmos de ordenación

Descripción:
Practicar diferentes métodos de ordenación de series numéricas
Conjunto de ejercicios de programación para implementar y utilizar los algoritmos de ordenación: i) burbujas, ii) heapsort y iii) quicksort
Incorporar las estructuras de datos necesarias para cada método de ordenación

Objetivos específicos:
Identificar y utilizar las estructuras de datos adecuadas para casda método de ordenación
Saber aplicar y entender el funcionamiento de los algoritmos recursivos en los métodos de ordenación que los incorporen
Ser capaz de utilizar los algoritmos de ordenación para ordenar diferentes objetos en diferentes escenarios

Material:
Enunciado de los ejercicios y MS Visual C++

Entregable:
El resultado de la actividad corresponde a un conjunto de ficheros de código C++ que se tendrán que enviar al espacio de entregas del campus y eventualmente presentalos delante del grupo del aula. Estas entregas y presentaciones serviran para evaluar el 2% de la nota correspondiente a la participación del estudiante.

Dedicación: 8h
Grupo mediano/Prácticas: 4h
Aprendizaje autónomo: 4h

SISTEMA DE CALIFICACIÓN

Cualquier acto de fraude académico, plagio o uso o mera tenencia al alcance de medios no autorizados en cualquier actividad de evaluación comportará la calificación de cero (0) en la prueba o entrega afectada. Además, de acuerdo con la normativa de la Universidad, la posible derivación de los hechos para la apertura de un expediente disciplinario implicará que la asignatura quede en el estado provisional de "pendiente de evaluación" hasta la resolución del expediente. La gestión de estas incidencias se lleva a cabo de acuerdo con el **Marco de actuación para la integridad académica en la evaluación de la UPC**.

Distinguimos tres tipos de actividades evaluables:
- Cuatro pruebas de evaluación sobre temas concretos de la asignatura que los alumnos resolverán en horario de clase (ET)
- Un examen parcial realizado durante la semana prevista en el calendario académico (EP)
- El examen final de la asignatura. (EF)

La nota de la asignatura (NF) se calculará utilizando la siguiente fórmula:
NF = 0.15* EP + 0.4 * ET + 0.35 * EF + 0.1 * PART

siendo EP la nota del examen parcial, ET las notas de los 4 ejercicios puntuables a realizar durante el curso (cada uno vale un 10% de la nota final) y EF la nota del examen final. También se evaluará la participación (PART) del alumno a partir de sus intervenciones y del interés de aprendizaje demostrado en clase.

Los alumnos que no superen la asignatura tendrán la posibilidad de presentarse a la prueba de Reevaluación (se reevaluará lo correspondiente a los exámenes parcial y final. La nota final de la asignatura después de la reevaluación, será como máximo un 5).

Las acciones irregulares que puedan llevar a una variación significativa de la calificación de uno o más estudiantes constituyen una realización fraudulenta de un acto de evaluación. Esta acción comporta la calificación descriptiva de suspenso y numérica de 0 del acto de evaluación ordinario global de la asignatura, sin derecho a reevaluación.

Si los docentes tienen indicios de la utilización de herramientas de IA no permitidas en las pruebas de evaluación, podrán convocar a los estudiantes implicados a una prueba oral o a una reunión para verificar la autoría.

NORMAS PARA LA REALIZACIÓN DE LAS PRUEBAS.

La realización de las diferentes pruebas se realizarán exclusivamente a través de los mecanismos establecidos, en las fechas y horarios indicados.

BIBLIOGRAFÍA

Básica:
- Stroustrup, B. A tour of C++. Addison-Wesley, 2013. ISBN 9780321958310.
- Kernighan, Brian W; Ritchie, Dennis M. The C programming language. 2nd ed. New Jersey: Prentice Hall, 1988. ISBN 0131103628.
- Franch Gutiérrez, Xavier [et al.]. Fonaments de programació: problemes resolts en C++ [en línea]. Barcelona: Edicions UPC, 2006 [Consulta: 10/12/2018]. Disponible a: http://hdl.handle.net/2099.3/36692. ISBN 9788483018828.

RECURSOS

Otros recursos:
C++ Reference: http://www.cplusplus.com/reference/ />C and C++ Programming: http://www.cprogramming.com/