Proyecto/Guía docente de la asignatura

Asignatura	PROGRAMACIÓN I		
Materia	PROGRAMACIÓN		
Módulo			
Titulación	GRADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN		
Plan	727	**Código**	48067
Periodo de impartición	1er. CUATRIMESTRE	**Tipo/Carácter**	FORMACIÓN BÁSICA
Nivel/Ciclo	GRADO	**Curso**	1º
Créditos ECTS	6		
Lengua en que se imparte	CASTELLANO		
Profesor/es responsable/s	JOSÉ FERNANDO DÍEZ HIGUERA DAVID GONZÁLEZ ORTEGA		
Datos de contacto (E-mail, teléfono…)	JOSÉ FERNANDO DÍEZ HIGUERA DESPACHO: 2D079 TELÉFONO: 983 18 55 62 E-MAIL: josdie@tel.uva.es DAVID GONZÁLEZ ORTEGA DESPACHO: 2D007 TELÉFONO: 983423000 ext. 5552 E-MAIL: david.gonzalez.ortega@uva.es		
Departamento	TEORÍA DE LA SEÑAL Y COMUNICACIONES E INGENIERÍA TELEMÁTICA		
Fecha de revisión por el Comité de Título	8 de julio de 2024		

1. Situación / Sentido de la Asignatura

1.1. Contextualización

La asignatura *Programación I* es la primera asignatura de naturaleza informática a la que se enfrentan los alumnos del Grado en Ingeniería de Tecnologías de Telecomunicación.

Esta asignatura es una experiencia de aprendizaje diseñada para enseñar a los alumnos cómo desarrollar programas que puedan resolver problemas, convertir datos, almacenar y recuperar información, ayudar a las personas a comunicarse o hacer cualquier cosa que el programador pueda imaginar. En el fondo, esto se hace cuando el programador escribe las instrucciones para el ordenador en un "lenguaje de programación". En otras palabras, los programadores de ordenadores actúan como traductores entre personas y ordenadores, escribiendo las especificaciones de un programa deseado en un lenguaje que el ordenador pueda entender.

Por último, no hay que perder de vista que, una vez cursada esta asignatura, el alumno debe disponer de una herramienta que tendrá que utilizar en otras asignaturas del Grado y, posiblemente, en su vida profesional.

1.2. Relación con otras materias

Esta asignatura está especialmente relacionada con la asignatura “Programación II”, en la que se aplican las técnicas y procedimientos de una metodología de desarrollo software concreta al análisis y diseño de un sistema software y amplía los conocimientos sobre lenguajes de programación al paradigma orientado a objeto; y con la asignatura “Fundamentos de Ordenadores y Sistemas Operativos”, la cual incluye programación en lenguaje de bajo nivel (ensamblador) y en lenguaje de alto nivel con llamadas al sistema operativo.

1.3. Prerrequisitos

El alumno que curse esta asignatura ha de poseer unos conocimientos básicos de informática a nivel usuario. En lo referente a programación, y siendo la primera asignatura del Grado que aborda dicha materia, se parte de la base de que el alumno no tiene conocimientos previos de la misma.

2. Resultados del proceso de formación y de aprendizaje (RD 822/2021)

2.1. Conocimientos o contenidos (RD822/2021)

- C3. Conocer y aplicar lenguajes de programación para desarrollar aplicaciones software y conocer alguna metodología de ingeniería de software relevante.

2.1. Habilidades o destrezas (RD822/2021)

- HD6 - Capacidad de razonamiento, análisis y síntesis.
- HD7 - Capacidad para relacionar conceptos y adquirir una visión integrada, evitando enfoques fragmentarios.
- HD8 - Capacidad de toma de decisiones en la resolución de problemas básicos de ingeniería de telecomunicación, así como identificación y formulación de los mismos.
- HD9 - Capacidad para trabajar en grupo, participando de forma activa, colaborando con sus compañeros y trabajando de forma orientada al resultado conjunto, y en un entorno multilingüe.
- HD10 - Conocimiento de materias básicas, científicas y tecnologías, que le capacite para el aprendizaje de nuevos métodos y tecnologías.
- HD15 - Capacidad para resolver problemas con iniciativa, creatividad y razonamiento crítico.
- HD24 - Capacidad de organización, planificación y gestión del tiempo.
- HD25 - Capacidad para comunicar, tanto por escrito como de forma oral, conocimientos, procedimientos, resultados e ideas relacionadas con las telecomunicaciones y la electrónica.
- HD26 - Capacidad para trabajar en cualquier contexto, individual o en grupo, de aprendizaje o profesional, local o internacional, desde el respeto a los derechos fundamentales, de igualdad de sexo, raza o religión y los principios de accesibilidad universal, así como la cultura de paz.

2.1. Competencias (RD822/2021)

- B2. Conocimientos básicos sobre el uso y programación de los ordenadores, sistemas operativos, bases de datos y programas informáticos con aplicación en ingeniería.

3. Objetivos

Al finalizar la asignatura el alumno deberá ser capaz de:

- Conocer los conceptos fundamentales relacionados con la programación.
- Aplicar las técnicas y procedimientos de una metodología de programación de un sistema software.
- Codificar y probar dicho sistema, aplicando técnicas de programación orientada a procesos (nivel básico) y a datos (nivel avanzado).
- Codificar, poner a punto y ejecutar programas sencillos en lenguaje C.
- Diseñar algoritmos sencillos basados en los esquemas de recorrido y búsqueda.
- Autoevaluar el trabajo realizado e identificar los propios errores y aspectos a mejorar.

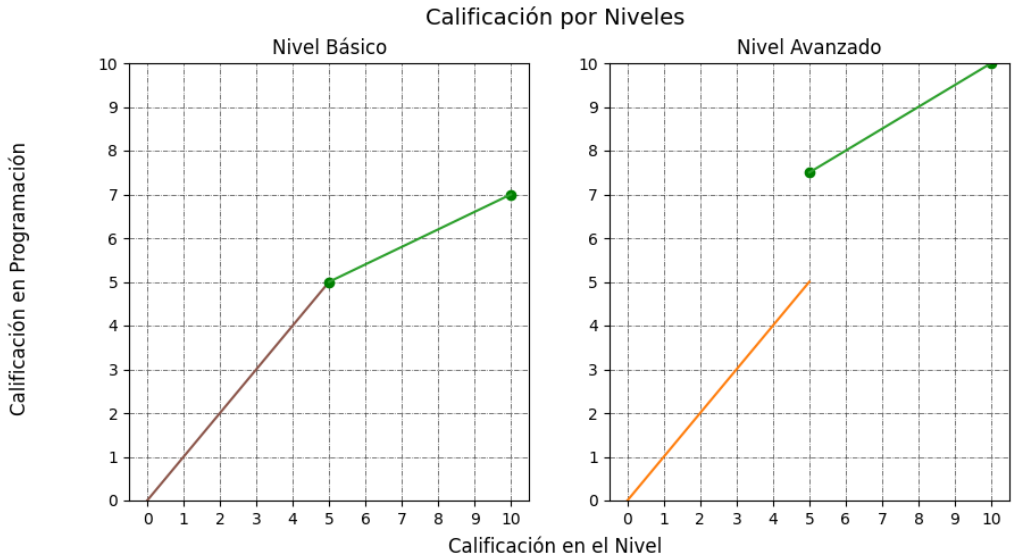
4. Niveles de Aprendizaje

En el presente curso, la asignatura se articula en dos niveles de aprendizaje, a semejanza de los cursos de idiomas, ya que el objetivo principal es el aprendizaje de un lenguaje (de programación): *Nivel Básico* y *Nivel Avanzado*. Con este esquema se pretende que sea el alumno el que decida los conocimientos y habilidades que quiere aprender sobre programación de ordenadores y/o la calificación que desea conseguir para su expediente. En este sentido, las calificaciones que se pueden obtener superando las pruebas de cada uno de los niveles, y la duración de cada nivel, se muestran en la siguiente tabla:

Nivel	Duración	Calificación
Básico	10 semanas	[5.0 – 7.0]
Avanzado	5 semanas	[7.5 - 10.0]

- Al finalizar las semanas correspondientes al nivel básico, se publicarán las calificaciones obtenidas en el nivel básico. Aquellos alumnos que superen este nivel con una nota igual o superior a 5.0 **obtienen una calificación en el intervalo [5.0 – 7.0] como nota de la asignatura**. A partir de ese momento, será potestad de cada alumno decidir si quiere continuar con el nivel avanzado.
- Al finalizar las semanas correspondientes al **nivel avanzado**, se publicarán las calificaciones de aquellos alumnos que hayan decidido cursar el nivel avanzado. Aquellos alumnos que superen este nivel con una nota igual o superior a 5.0 **podrán optar a una calificación entre 7.5 y 10.0**, en función de los resultados de la evaluación de este nivel.

En la siguiente gráfica se pueden observar las escalas de calificaciones correspondientes a cada uno de los niveles de aprendizaje.



NOTA: Como se indica más adelante, en la sección “Sistema y características de la evaluación”, *si un alumno no alcanza los requisitos mínimos de cada elemento de evaluación, su calificación final en la asignatura será el mínimo entre el valor calculado según la ponderación descrita en la tabla y 4.0*.

5. Contenidos y/o bloques temáticos

En consonancia con la sección Niveles de Aprendizaje, los conocimientos y habilidades que el alumno debe adquirir se articulan en dos bloques temáticos, correspondientes a los dos niveles de aprendizaje.

Los objetivos de aprendizaje describen en detalle todo lo que el alumno va a aprender durante este curso. Es importante que el alumno los tenga presentes desde el primer momento, aunque el profesor le irá recordando los objetivos que están implicados en las diferentes actividades del curso.

5.1. Características comunes a todos los bloques

a. Métodos docentes

- Clase expositiva participativa.
- Resolución de problemas.
- Yincanas digitales en el laboratorio.

Para más información, consultar la sección “Métodos docentes y principios metodológicos”.

b. Plan de trabajo

El Plan de Trabajo de la asignatura, donde se describe la planificación detallada, incluyendo fechas de las prácticas por grupo, se entregará al comienzo de la asignatura.

c. Material docente

La bibliografía propuesta para esta asignatura tiene como objeto servir de soporte al aprendizaje de los conceptos y del lenguaje de programación de la asignatura, por lo que estudiarse alguno (¡o todos!) los libros, no tiene sentido por sí mismo.

Sin embargo, recomendamos el uso de alguno de los libros mencionados en el enlace a la Plataforma Leganto de la Biblioteca de la UVA con la bibliografía recomendada. (en el orden en el que aparecen). Existen ejemplares en la biblioteca y todos ellos tratan el mismo tema, con diferentes enfoques, ejemplos y orden de exposición.

§ Otros recursos telemáticos

Los recursos documentales, audiovisuales y de hipermedia elaborados para la implantación de la clase inversa (*flipped classroom*), se enlazarán directamente en las secciones correspondientes del Campus Virtual.

d. Recursos necesarios

- Aula con proyector multimedia y pizarra para sesiones de discusión.
- Laboratorio de prácticas, con un ordenador por alumno, para las sesiones de laboratorio. Cada ordenador debe contar con el entorno de desarrollo para el lenguaje C que se va a utilizar.
- Plataforma educativa para publicar material didáctico, guías de ejercicios, soluciones, tareas, etc.

En esta asignatura se utilizan herramientas docentes online para la docencia y la evaluación. En caso de un transcurso normal de la docencia estarán disponibles las aulas informáticas del centro. En caso de una afección por medidas sanitarias especiales, el alumno debe contar con medios informáticos y telemáticos suficientes para interactuar con el Campus Virtual y con los sistemas de videoconferencia.

5.2. Temporalización

BLOQUE TEMÁTICO	CARGA ECTS	PERIODO PREVISTO DE DESARROLLO
Bloque 1: Nivel Básico	**4.0**	**Semanas 1 a 10**
Bloque 2: Nivel Avanzado	**2.0**	**Semanas 11 a 15**

5.3. Desglose por bloques

Bloque 1: Nivel Básico

Carga de trabajo en créditos ECTS: 4.0

a. Contextualización y justificación

En este primer bloque se introducen los conceptos básicos sobre los diferentes métodos para el desarrollo de aplicaciones informáticas, se presenta C, el lenguaje de programación que se utilizará para la implantación de los programas desarrollados, y se proporcionan los elementos básicos de la **programación orientada a procesos.**

b. Objetivos de aprendizaje

Al final de este bloque el alumno será capaz de:

- En relación con los **tipos de datos** elementales y sus **operaciones** (enteros, caracteres y reales):
  - Escribir los tipos de datos elementales y las operaciones que actúan sobre ellos.
  - Escribir la declaración de datos de cualquiera de los tipos elementales.
  - Indicar el código ASCII de cualquier carácter, con la ayuda de la tabla correspondiente.
- En relación con los **punteros**:
  - Comprender el concepto de puntero y sus diferencias con una variable simple.
  - Escribir el código necesario para acceder al dato apuntado por un puntero.
- En relación con las **sentencias básicas de asignación** y de **entrada/salida**:
  - Describir el funcionamiento de las sentencias básicas.
  - Predecir el resultado de una secuencia de sentencias básicas.
  - Codificar una tarea convenientemente especificada, utilizando la secuencia de sentencias básicas adecuada.
- En relación con los **archivos**:
  - Explicar el concepto de archivo, para qué sirve, y cuáles son las operaciones típicas sobre archivos de texto (crear, abrir, leer, escribir, preguntar por fin de archivo y cerrar).
  - Escribir las sentencias necesarias para realizar las operaciones básicas con un archivo de texto.
  - Escribir las sentencias necesarias para determinar el tipo de error que se ha producido al realizar una operación con un archivo de texto.
- En relación con las **sentencias de control**:
  - Conocer y usar los operadores que se utilizan para construir expresiones lógicas.
  - Usar las sentencias de selección `if`, `if-else` para elegir entre varias acciones alternativas.
  - Utilizar la sentencia de selección múltiple `switch` para escoger entre muchas alternativas de acción.
  - Utilizar las etiquetas `case` para identificar las acciones alternativas en las sentencias `switch`.
  - Ejecutar sentencias repetidamente con la sentencia repetitiva `for`.
  - Usar la sentencia de repetición `while` para ejecutar sentencias repetidamente.
- En relación con las **funciones**:
  - Definir el concepto de función y su utilidad.
  - Definir los conceptos: cabecera de función, parámetros formales, variables locales, resultado de la función, llamada a función, parámetros reales, paso de parámetros.
  - Describir la diferencia entre paso de parámetros por valor o por referencia.
  - Codificar convenientemente una llamada a función, pasando correctamente los parámetros.
  - Codificar en forma de función una tarea convenientemente especificada, estableciendo adecuadamente los parámetros necesarios.
  - Agrupar las funciones de un programa relacionadas entre sí en un módulo.
- En relación con el **entorno de desarrollo**:
  - Definir los conceptos de compilación, construcción y ejecución.

- Realizar las operaciones necesarias para crear/abrir un proyecto y añadir y eliminar elementos a un proyecto.
- Diseñar el archivo *makefile* para una aplicación determinada.
- Realizar las operaciones necesarias para editar, compilar, montar y ejecutar un programa, y localizar las carpetas donde están los archivos generados en cada uno de los pasos.
- Interpretar adecuadamente los mensajes de error de compilación y corregir el error de compilación correspondiente.
- Describir las funcionalidades básicas del *depurador*.
- Realizar correctamente las operaciones básicas del depurador (insertar un punto de parada, ejecutar paso a paso y visualizar valores de variables).
- Identificar y subsanar los errores de ejecución de un programa, utilizando adecuadamente el depurador.

c. Contenidos

- Elementos Básicos del lenguaje C.
- Anatomía de un programa en C.
- Funciones.
- Sentencias de Control.
- Vectores numéricos
- Algoritmos de recorrido y búsqueda.

Bloque 2: Nivel Avanzado

Carga de trabajo en créditos ECTS: 2.0

a. Contextualización y justificación

El bloque 2 se centra en la introducción de los conceptos relacionados con la **programación orientada a datos**: tipos de datos estructurados, registros, archivos binarios, vectores multidimensionales y vectores de registros, estructuras de datos dinámicas, y asignación dinámica de memoria.

b. Objetivos de aprendizaje

Al final de este bloque el alumno será capaz de:

- En relación con las **estructuras estáticas de datos**:
  - Describir las estructuras de datos fundamentales, y las operaciones típicas sobre ellas.
  - Escribir la declaración de una estructura de datos convenientemente especificada.
  - Escribir el código necesario para acceder a un elemento o conjunto de elementos de una estructura de datos.
  - Elegir la estructura de datos más adecuada para una aplicación determinada.
- En relación con los **archivos binarios**:
  - Recordar el concepto de archivo, para qué sirve, y cuáles son las operaciones típicas sobre archivos (crear, abrir, leer, escribir, preguntar por fin de archivo y cerrar).
  - Escribir las sentencias necesarias para realizar las operaciones básicas con archivos binarios.
  - Escribir las sentencias necesarias para determinar el tipo de error que se ha producido al realizar una operación con un archivo binario.
- En relación con las **estructuras dinámicas de datos**:
  - Asignar y liberar memoria dinámicamente para diferentes tipos de datos.
  - Crear y redimensionar vectores dinámicos.
- En relación con los **vectores dinámicos**:
  - Escribir la declaración de un vector dinámico.
  - Escribir el código necesario para asignar memoria a un vector dinámico.
  - Escribir el código necesario para reasignar memoria a un vector dinámico.
  - Escribir el código necesario para liberar la memoria asignada un vector dinámico.

c. Contenidos

- Aritmética de punteros.
- Estructuras Estáticas de Datos.
  - Registros, uniones, enumeraciones, campos de bits.
- Asignación dinámica de memoria.
- Estructuras Dinámicas de Datos.
  - Vectores dinámicos.
  - Estructuras autoreferenciadas.
  - Listas enlazadas, pilas y colas

6. Métodos docentes y principios metodológicos

Siguiendo una de las recomendaciones de la Dirección de la ETSIT en cursos anteriores, el flujo de trabajo semanal de esta asignatura está edificado sobre los cimientos del modelo pedagógico denominado *Flipped Classroom* (FC), complementado con técnicas de *Just-in-time Teaching y ConcepTest.*

- *Flipped Classroom* (o *clase inversa*) es "un modelo pedagógico que transfiere el trabajo de determinados procesos de aprendizaje fuera del aula y utiliza el tiempo de clase, junto con la experiencia del docente, para facilitar y potenciar otros procesos de adquisición y práctica de conocimientos dentro del aula".
- *Just-in-time (JIT) Teaching (o enseñanza justo a tiempo)* es una estrategia pedagógica que utiliza la retroalimentación entre las actividades de aula y el trabajo que el alumnado hace en casa para preparar las sesiones que se dan en el aula. Los objetivos son: aumentar el aprendizaje durante el tiempo que se está en el aula, fomentar la motivación de los alumnos, animar a los estudiantes a que se preparen las clases y permitir al profesor encontrar las actividades más adecuadas para las necesidades de sus estudiantes.
- *ConcepTests* (o *pruebas de concepto*) son pruebas cortas, informales y dirigidas que se administran durante la clase para ayudar a los instructores a evaluar si los estudiantes comprenden los conceptos clave. Se pueden usar tanto para evaluar los conocimientos previos de los estudiantes (entrar en un curso o unidad) como la comprensión del contenido del curso o unidad actual. Por lo general, estas pruebas consisten en entre una a cinco preguntas de opción múltiple. Se les pide a los estudiantes que utilicen su portátil o teléfono móvil para cumplimentar el cuestionario.

Como ya se ha comentado en la sección 4, la asignatura se estructura en niveles de aprendizaje. Cada nivel se compone de varias etapas (por lo general, de **dos semanas** de duración), que a su vez se articulan en una serie de tramos de diferentes actividades. A continuación, se describe la hoja de ruta de una etapa estándar (sin jornadas no lectivas), con la enumeración de los tramos a cubrir.

I. Briefing

Tipo de actividad: no presencial

Al comienzo de cada tramo, y a través del Campus Virtual de la asignatura, el profesor proporciona a los alumnos la hoja de ruta correspondiente a dicho tramo, con la siguiente información:

- Fecha de inicio y final de tramo.
- Objetivos de aprendizaje.
- Enlaces a los recursos documentales necesarios para conseguir los objetivos de aprendizaje previstos.
- Actividades presenciales y de trabajo individual no presencial a realizar durante el tramo, indicando cuáles de ellas son *formativas*, y cuáles tendrán una calificación *sumativa*.
- Criterios específicos de calificación de las actividades evaluables.

II. Periplo a través de los recursos

Tipo de actividad: no presencial	*No evaluable*

En este tramo el alumno debe utilizar los recursos proporcionados por el profesor, y/o leyendo PDFs interactivos, tutoriales y otros tipos de documentación escrita, relacionados con el material que se abordará durante la próxima sesión de clase.

III. Aplicación de la enseñanza *Just-in-time*

Tipo de actividad: no presencial	Tipo de evaluación: diagnóstica

Para esta actividad el profesor habrá desarrollado previamente un conjunto de preguntas efectivas que se publicarán en el Campus Virtual para que los estudiantes respondan antes de la siguiente sesión de teoría. Estas preguntas son generalmente de opción múltiple, y requieren que los estudiantes hayan completado el tramo anterior. Después de la fecha límite de publicación, pero siempre antes de que comience la clase, el profesor examina las respuestas de los estudiantes. En la mayoría de los casos, el profesor puede utilizar esta revisión para hacer ajustes en las actividades de aula previstas. Si el profesor cree que el alumnado ha dominado un tema, puede reducir o eliminar la discusión de dicho tema durante la clase. De igual manera, si las respuestas muestran que los alumnos tienen dificultades concretas, estas serán vistas durante la clase con más detenimiento.

IV. Sesión de clase

A. Procedimiento de desarrollo de la sesión

En circunstancias normales, y atendiendo a las características específicas de esta asignatura, la sesión de clase en aula se desarrolla mediante el siguiente procedimiento: el profesor conecta su portátil al proyector del aula, con el fin de que todos los alumnos puedan visualizar en la pantalla del aula una presentación (*PowerPoint, Google Colab, H5P, etc.*), el desarrollo de un ejercicio en el portátil del profesor utilizando el entorno de desarrollo del lenguaje de programación utilizado, y cualquier otro documento o aplicación que el profesor desee mostrar. En otras palabras, el profesor **comparte** la pantalla de su portátil con los alumnos a través del proyector y la pantalla de proyección instalados en el aula. Con cierta frecuencia, el profesor puede 'invitar' a algunos alumnos a salir a la pizarra para realizar un ejercicio, lo que en programación se traduce en utilizar el ordenador del profesor para editar el código, de forma que el resto de la clase pueda seguir el desarrollo, sugiriendo soluciones, detectando errores en el código, etc., en lo que se podría calificar como una actividad de *aprendizaje cooperativo*, ya que todos los alumnos cooperan en la resolución del problema.

Por otro lado, y en el mejor de los escenarios en el que todos los alumnos puedan encontrar acomodo en el aula, se plantea el problema de que los alumnos más alejados de la pantalla de proyección pueden tener serias dificultades para visualizar correctamente los contenidos allí proyectados. Como solución a este inconveniente, durante las sesiones de aula se utilizará la aplicación de videoconferencia *Collaborate* para que los alumnos que tengan dificultades audiovisuales para seguir la clase lo puedan hacer a través de un portátil que necesariamente tendrán que llevar a clase. [*Collaborate es una plataforma que permite generar espacios de comunicación en forma de salas virtuales, donde varios participantes se reúnen y pueden compartir diferentes aplicaciones: presentaciones, escritorio, pizarra virtual, etc*]. De esta forma la pantalla del profesor se compartirá simultáneamente a través del proyector y de *Collaborate*. Este procedimiento permite también mantener el aprendizaje cooperativo en el desarrollo de ejercicios en clase, ya que el salir a la pizarra, en este contexto, se traduce en que un alumno comparta la ventana del entorno de desarrollo con el resto de la clase a través de *Collaborate*. Así mismo, este procedimiento permite que alumnos con riesgo, o alumnos que no puedan acceder al aula por exceso de aforo, puedan seguir la clase de forma telepresencial, e interactuar con sus compañeros y con el profesor en las mismas condiciones que los alumnos presentes en el aula. Huelga decir que en ningún caso se plantea la utilización de dispositivos de captación de imagen (ya que lo que interesa al alumno es el contenido y la explicación del profesor y la interacción con sus compañeros), y en ningún caso se va a grabar la sesión de clase compartida a través de la plataforma *Collaborate*.

Y lo que es más importante, este procedimiento es igualmente válido en el caso de que las circunstancias aconsejen u obliguen a conmutar toda la docencia a modo telepresencial, sin tener que efectuar cambio alguno, salvo la ubicación física de los alumnos.

B. Desarrollo de la sesión

Como se ha comentado en la sección III, el profesor, después de analizar los resultados de la enseñanza *Just-in-time*, habrá planificado la sesión con recursos y actividades que profundicen en conceptos deficientemente asimilados por los alumnos. Una vez desplegados dichos recursos, el profesor utilizará la técnica de *ConcepTest* para recabar nueva información sobre la asimilación grupal de conceptos, al menos una vez durante la sesión (evaluación diagnóstica), con aplicaciones del tipo **WooClap**, *Kahoot*, *Socrative*, etc., y al final de cada sesión (evaluación sumativa), mediante un cuestionario de Moodle realizado con la app para móviles de dicha plataforma (se plantea esta alternativa ya que es posible que algún alumno no disponga de portátil, pero es casi seguro que todos tienen un teléfono móvil).

V. Sesión de laboratorio

A. Procedimiento de desarrollo de la sesión

Las sesiones de laboratorio se emplearán fundamentalmente en aplicar el aprendizaje basado en la resolución de problemas, bajo el formato de yincana *digital* "competitiva". Los alumnos individualmente, o en parejas, tienen que ir superando una serie de etapas, hasta llegar al ***trofeo*** que les identifica como ganadores. Una vez conseguida una posible solución, se introducirá en un programa proporcionado por el profesor (*unit-test*, *stubs & drivers, etc.*) con el siguiente resultado:

- Si la solución es correcta, le proporcionará el recurso necesario para acceder a las especificaciones de la siguiente etapa.
- Si la solución no es correcta, se lo indicará al equipo, para que revise sus resultados.

Las sesiones serán de dos tipos:

- **Formativas**: los alumnos se agrupan en parejas, y contarán con el apoyo del profesor. El objetivo de estas sesiones es que los alumnos vayan adquiriendo progresivamente los conocimientos y habilidades de cada nivel de aprendizaje, siguiendo la modalidad "*learning by doing*". Como ya postulaba Aristóteles, "lo que tenemos que aprender a hacer, lo aprendemos haciendo".
- **Sumativas**: los alumnos tienen que resolver la yincana correspondiente de forma autónoma. El objetivo de estas sesiones es evaluar el grado de adquisición de conocimientos y habilidades de cada alumno.

Los resultados de ambos tipos de sesiones se tendrán en cuenta en la calificación de las prácticas de laboratorio.

En la primera semana de clase, se pasará un formulario a los alumnos, con un doble objetivo:

- Una parte del cuestionario permitirá obtener un primer indicio de los conocimientos de cada alumno antes de comenzar la asignatura.
- La segunda parte del cuestionario permitirá establecer una metodología de emparejamiento basada en la similitud de las capacidades de cada alumno para el aprendizaje de la programación. La selección de esta metodología de emparejamiento se basa en varios estudios que demuestran un mejor desempeño de los estudiantes que fueron emparejados por su capacidad demostrada con el de los estudiantes que fueron emparejados al azar o trabajaron solos (Braught et al., 2010).

En las semanas posteriores, se proporcionarán nuevos cuestionarios de diagnóstico relacionados con los conceptos de la semana en curso, para analizar la evolución del aprendizaje de cada alumno.

Además, al comienzo de cada sesión de laboratorio se realizarán los **cuestionarios de conocimientos básicos**.

7. Tabla de dedicación del estudiantado a la asignatura

ACTIVIDADES PRESENCIALES o PRESENCIALES A DISTANCIA (1)	HORAS	ACTIVIDADES NO PRESENCIALES	HORAS
Sesiones de aula	20	Visionado y/o lectura de recursos didácticos	35
Resolución de problemas en laboratorio	40	Cuestionarios de enseñanza JIT	10
		Cuestionarios de Aprendizaje	20
		Estudio autónomo	25
Total presencial	**60**	Total no presencial	**90**
		TOTAL presencial + no presencial	**150**

(1) Actividad presencial a distancia es cuando un grupo sentado en un aula del campus sigue una clase por videoconferencia de forma síncrona, impartida por el profesor.

8. Sistema y características de la evaluación

La evaluación de la asignatura se articula en dos elementos: **cuestionarios de conocimientos básicos** y **resolución de problemas en laboratorio**.

INSTRUMENTO/PROCEDIMIENTO	PESO EN LA NOTA FINAL	OBSERVACIONES
TEORÍA: Cuestionarios de Conocimientos básicos	35%	Para superar cada cuestionario de conocimientos básicos el alumno dispone de cuatro intentos durante el curso, más dos intentos adicionales en las convocatorias ordinaria y extraordinaria.

PRÁCTICAS: resolución de problemas en laboratorio	65%	Para promediar con la nota de teoría, el alumno debe obtener una calificación mínima en prácticas de 5 sobre 10.
Si un alumno no alcanza los requisitos mínimos descritos en la tabla anterior, su calificación final en la asignatura será el mínimo entre el valor calculado según la ponderación descrita en la tabla y 4.0.		

CRITERIOS DE CALIFICACIÓN
La **Convocatoria Ordinaria** se realiza en la fecha prevista en el calendario de exámenes. Esta convocatoria está indicada para: 1. Alumnos que no han aprobado alguno de los niveles en la Evaluación Continua. 2. Alumnos que, habiendo aprobado alguno de los niveles en la evaluación continua, deseen mejorar su calificación presentándose a un nivel superior al aprobado. En este caso, si un alumno no supera el nivel al que se presenta en esta convocatoria, mantiene la calificación obtenida en la Evaluación Continua. En cualquier caso, el alumno se puede presentar a cualquiera de los dos niveles, pero solamente a uno de ellos. Solamente se tiene que examinar de los cuestionarios asociados al nivel del que se quiere examinar, y resolver los problemas asociados a dicho nivel.
La **Convocatoria Extraordinaria** se realiza en la fecha prevista en el calendario de exámenes. Esta convocatoria está indicada solamente para alumnos que no han aprobado ninguno de los niveles en la Evaluación Continua ni en la Convocatoria Ordinaria. El alumno se puede presentar a cualquiera de los dos niveles, pero solamente a uno de ellos. Solamente se tiene que examinar de los cuestionarios asociados al nivel del que se quiere examinar, y resolver los problemas asociados a dicho nivel.

9. Consideraciones finales

10. Referencias

- Braught, G., MacCormick, J., & Wahls, T. (2010). The Benefits of Pairing by Ability. In Proceedings of the 41st ACM Technical Symposium on Computer Science Education (SIGCSE '10). Association for Computing Machinery, New York, NY, USA, 249–253. https://doi.org/10.1145/1734263.1734348