None
None Logroño (La Rioja)
None

PROGRAMACIÓN DIDÁCTICA

2º BACHILLERATO- PROGRAMACIÓN 26-27

Programación (Optativa) - 2º Bachillerato de Ciencias y Tecnología

I.E.S. Tomás Mingot (26003209) 2026/2027

Fechas de comienzo y fin

Inicio aproximado: 09-09-2026

Finalización aproximada: 23-06-2027

Jefe del departamento responsable de la programación

Jesús Félix Jiménez González

Docentes implicados en el desarrollo de la programación

- Ismael Heras Cabezón
- Jesús Félix Jiménez González

PROCEDIMIENTO PARA LA ADOPCIÓN DE MEDIDAS DE ATENCIÓN A LA DIVERSIDAD

Atención a la diversidad: en nuestra metodología, la clave es garantizar el avance seguro, el logro paso a paso. Evitando lagunas conceptuales, competencias insuficientemente trabajadas y, en definitiva, frustraciones por no alcanzar cada alumno, dentro de los principios de atención individualizada y educación inclusiva, todo aquello de lo que es capaz.

En relación a las necesidades y diferencias individuales, las actividades y situaciones de aprendizaje se estructuran en base a un nivel creciente de complejidad o con posibilidades de ampliación en su defecto, de modo que, el alumno que encuentra más dificultades sea capaz de llegar a la solución estándar y aquellos más ágiles tengan más recorrido.

ORGANIZACIÓN Y SEGUIMIENTO DE LOS PLANES DE RECUPERACIÓN DEL ALUMNADO CON MATERIAS PENDIENTES DE CURSOS ANTERIORES

Se realizará una reunión a principio de curso (durante el mes de octubre) para establecer las fechas de los ejercicios de evaluación y los trabajos a realizar.

El Jefe de Departamento, como profesor responsable, se encargará de aclarar las dudas planteadas y de orientar la recuperación a los alumnos que lo requieran.

Como actividades de recuperación el alumno realizará:

Una colección de ejercicios, tareas y trabajos de recuperación, propuesta por el profesor.

Tres pruebas competenciales: cada una de ellas abordará una parte de los contenidos del curso. Es condición imprescindible entregar las actividades de recuperación, realizadas correctamente, para poder presentarse a cualquiera de las pruebas indicadas.

Dada la naturaleza de la materia, pudiera darse la circunstancia que los medios y aplicaciones digitales necesarias para abordar alguna parte del temario no estén al alcance de un alumno en su casa, en tal caso, se adaptará o se planteará una actividad similar de modo que pueda ser trabajada por este su domicilio.

Se considerará que un alumno ha recuperado el área cuando tenga aprobadas todas las competencias específicas de la materia.

LIBROS O MATERIALES VAN A SER UTILIZADOS PARA EL DESARROLLO DE LA MATERIA

Nombre	ISBN

ACTIVIDADES EXTRAESCOLARES/COMPLEMENTARIAS QUE SE VAN A LLEVAR A CABO

Nombre	Inicio	Fin

UNIDADES DE PROGRAMACIÓN

Las unidades de programación organizan la acción didáctica orientada hacia la adquisición de competencias. En este proceso se desarrollan los saberes básicos (conocimientos, destrezas y actitudes), cuyo aprendizaje resulta necesario para la adquisición de compentecias.

Los saberes básicos desarrollados en cada unidad de programación son impartidos en clase a través de las denominadas situaciones de aprendizaje. Éstas, a su vez, se evalúan a través de procedimientos de evaluación; los utilizados en esta programación didáctica son:

Según lo programado, el porcentaje de uso de los procedimientos de evaluación para obtener la calificación final del alumnado es:	
Pruebas de ejecución:	**44,12%**
Revisión del cuaderno o producto:	**33,82%**
Examen tradicional/Prueba objetiva/competencial:	**22,06%**

En este apartado, se muestran secuenciadas las diferentes unidades de programación asociadas con la materia (Programación (Optativa) de 2º Bachillerato de Ciencias y Tecnología). También se indican las fechas aproximadas de comienzo de cada una de las unidades así com el número de periodos lectivos que se estima serán necesarios para impartir la docencia correspondiente.

CONSIDERACIONES PARA LA EVALUACIÓN

1º Para aprobar la asignatura se deberá haber superado las tres evaluaciones por separado.

2º En las pruebas escritas (exámenes) de cada evaluación se deberá obtener al menos **un tres** para poder **hacer la media** ponderada con los demás ítems de esa evaluación (exposiciones, trabajos, actividades, ejercicios…)

3º En caso de que el alumno demuestre un exceso de conocimiento con respecto a su nivel y/o edad, se le podrá solicitar una prueba de verificación para acreditar ese nivel y trayectoria.

4º Los trabajos y/o actividades entregados fuera de plazo con un retraso máximo de tres días hábiles se evaluarán sobre el 70%. Si los trabajos y/o actividades entregados fuera de plazo sobrepasan este plazo se les calificará con la nota mínima permitida.

5º Cuando un alumno se presente a subir nota de un examen, la calificación final de esta prueba escrita será la media de las notas que haya obtenido en el primer examen y el examen de subir nota.

Comienzo aprox.	Nombre de la unidad de programación (UP)	Periodos
07-09-2026	1.- Introducción a la programación	24
01-12-2026	2.- Diseño de videojuegos 2D/3D con Godot	30
15-02-2027	3.- Diseño de videojuegos VR con Godot	30

None
None Logroño (La Rioja)
None

PÉRDIDA DERECHO EVALUACIÓN CONTINUA

Se incurrirá en la **pérdida del derecho a la evaluación continua** cuando:

1. El alumno acumule a lo largo de **un trimestre o evaluación**, en una misma área o materia, más de un **30 %** de faltas sobre el tiempo asignado a la misma, sean justificadas o no. El alumno pierde el derecho a ser evaluado conforme a los criterios ordinarios de evaluación, en la consideración de que la falta de asistencia a clase de forma reiterada puede impedir la aplicación de los criterios normales de evaluación continua.

2. El alumno acumule más de un **20 %** de faltas, sean o no justificadas, en cualquier área o materia, a lo largo del **curso escolar.** El alumno pierde el derecho a ser evaluado conforme a los criterios ordinarios de evaluación, en la consideración de que la falta de asistencia a clase de forma reiterada puede impedir la aplicación de los criterios normales de evaluación continua.

3. En el caso de sanción determinada por falta gravemente perjudicial para la convivencia en el centro. El alumno pierde el derecho a ser evaluado conforme a los criterios ordinarios de evaluación, en la consideración de que la falta de asistencia a clase de forma reiterada puede impedir la aplicación de los criterios normales de evaluación continua.

De esta forma, en las asignaturas de 4 horas semanales los alumnos perderían la evaluación continua al llegar a **15 faltas** durante una **evaluación** o a **28 faltas** durante todo el **curso lectivo**, en las asignaturas de 3 horas semanales los alumnos perderán la evaluación continua al llegar a **11 ausencias** durante una **evaluación** o a **21** durante todo el **curso lectivo**, y en las asignaturas de 2 horas semanales los alumnos perderán la evaluación continua al llegar a **8 ausencias** durante una **evaluación** o a **14** durante todo el **curso lectivo**.

Cuando los **alumnos** estén en **riesgo** de perder la **evaluación continua** se entregará a la familia el documento "**Aviso Riesgo Perdida de Evaluación Continua.doc**" debidamente cumplimentado por parte del profesor correspondiente de la materia en cuestión (documento compartido a todos los miembros del departamento en grupo teams).
Cuando los **alumnos** ya hayan **perdido** del todo el derecho a la **evaluación continua** se les entregará el documento "**Aviso Perdida de Evaluación Continua.doc**" debidamente cumplimentado por parte del profesor correspondiente de la materia en cuestión (documento compartido a todos los miembros del departamento en grupo teams) y será este mismo profesor el que hable con el alumno y la familia para que conozcan la forma en la que se evaluará al alumno posteriormente.
En estos casos el alumno deberá realizar un **examen final** de la materia preparado por el profesor correspondiente de esa asignatura. También deberá **entregar los trabajos**, si los hubiere, que se hayan mandado durante la evaluación o el curso académico en los que se ha perdido el derecho a la evaluación continua.

1.- INTRODUCCIÓN A LA PROGRAMACIÓN (24 PERIODOS)

Esta unidad de programación está compuesta por 1 situaciones de aprendizaje que son descritas a continuación.

DEL CÓDIGO A LA REALIDAD

Descripción y saberes básicos de la situación de aprendizaje, integrando metodologías:

El objetivo de la presente SAP es que el alumnado se familiarice con un lenguaje de programación de alto nivel, y su aplicación en sistemas microprogramables. Se trabajarán aspectos de los siguientes saberes básicos.

- Organización de la información y tipos de estructuras de datos.
- Descomposición de problemas y secuenciación.
- Generalización y reconocimiento de patrones.
- La abstracción como proceso específico, mediante representaciones simplificadas de los sistemas.
- Diseño mediante diagramas de flujos y pseudocódigo. La iteración, análisis recursivo, diseño iterativo, inducción.
- Representación binaria de la información: el bit, el byte. Codificación binaria. Sistemas de numeración. Representación hexadecimal. Códigos de Entrada/Salida. Medidas de la Información.
- Almacenamiento, transmisión y tratamiento básico de la información en binario: números, texto, imágenes, ficheros.
- Lenguajes de programación: Definición. Tipos de lenguajes. Lenguajes interpretados y compilados.
- Estructura de un programa informático y elementos básicos del lenguaje: Tipos básicos de datos.
- Constantes y variables. Operadores y expresiones. Comentarios. Estructuras de control.
- Condicionales.

Producto solicitado a los alumnos en la situación de aprendizaje:

Los alumnos deberan desarrollar pequeños códigos para trabajar cada uno de los aspectos principales de una placa microprogramable como puede ser Arduino. Cada código debe precederse de la correcta segmentación del problema y reflejar su algoritmo mediante un diagrama de flujo.

Competencias específicas que se van a trabajar en esta situación de aprendizaje:

3.- Organizar y analizar la información de un modo lógico, desde el punto de vista del pensamiento computacional, formulando y abordando problemas de forma que sus soluciones puedan ser representadas como secuencias de instrucciones y algoritmos.

4.- Identificar problemas o tareas susceptibles de ser resueltos u automatizados por medios informáticos. Discernir y concretar los requisitos del programa, idear la arquitectura del mismo y llevarlo a cabo mediante el uso lenguajes de programación de alto nivel, para posteriormente comprobar que se cumple con las especificaciones que previamente se establecieron en el análisis de requisitos.

5.- Comprobar, rediseñar y proponer mejoras a soluciones computacionales propias o ajenas, de forma individual o en grupo, estableciendo la evaluación y el análisis de satisfacción del resultado final en función de criterios funcionales, así como éticos, cívicos y medioambientales para depurar el funcionamiento mediante la detección de errores o la necesidad de incluir posibles mejoras al diseño.

En esta situación de aprendizaje se van a llevar a cabo (al menos) 3 actividades:

Reto 1: Introducción a Tinkercad

Se plantea un reto simulado (Tinkercad) consistente en la simulación básica de un electrodoméstico de uso cotidiano que requiera la utilización y dominio de:

Hardware:

- Arduino, diodos, resistencias, pulsadores, potenciómetros, protoboard, display 7 segmentos.

Software:

- Estructura báscica del código de Arduino (setup y loop)
- Decalaración y tipos de variables y señales de entrada y salida.
- Operadores lógicos / matemáticos.
- Estructura de control if

Para evaluar el desarrollo de la actividad se hacen uso de procedimientos de evaluación. Estos procedimientos de evaluación miden la adquisición de las competencias por parte del alumnado utilizando los denominados criterios de evaluación.

A continuación se describen los procedimientos de evaluación con sus criterios asociados:

Tipo	Nombre	Criterios evaluados (peso)
Pruebas de ejecución	Reto 1:	3.1.- Comprender, analizar y descomponer problemas, reconociendo en ellos los patrones, secuencias y condiciones que permitan formular soluciones a los mismos en forma de algoritmo. (**1**) 3.2.- Conocer y elaborar representaciones gráficas de algoritmos mediante diagramas de flujo. Respetado las reglas de construcción, la simbología adecuada y las estructuras de control oportunas, sin perder de vista la eficacia y eficiencia en su diseño. (**1**) 3.3.- Traducir, interpretar y expresar los distintos pasos que va a realizar un programa, partiendo o no de diagramas de flujo, del modo más próximo posible a un lenguaje de programación, mediante pseudocódigo. (**1**) 4.1.- Comprender los modos de almacenamiento, trasmisión, representación y codificación de la información, así como la cuantificación de la misma. (**1**) 4.2.- Diferenciar algoritmo de programa informático, comprender la necesidad y el porqué de la diversidad de lenguajes informáticos existentes, así como sus tipos, para poder reconocer el ámbito de aplicación y objeto de estos. (**1**) 4.3.- Conocer la estructura de un lenguaje de programación de alto nivel, los tipos de datos, operadores y estructuras de control de los que se sirve, para poder ser capaz de idear, traducir y desarrollar programas informáticos. (**1**) 5.1.- Adoptar protocolos efectivos de análisis y comprobación a las soluciones computacionales desarrolladas. (**1**) 5.2.- Rediseñar y proponer mejoras a programas propios o ajenos que contribuyan a mejorar la solución técnica, valorando aspectos medioambientales y éticos si los hubiera. (**1**)

Reto 2:

Se plantea un reto (pudiéndo ser continuación o ampliación del reto 1) el que además se requieran nuevos conceptos de código que incluyen:

- Estructura de control Switch
- Operadores And, or, not.
- Declaración y uso de funciones con y sin retorno de datos.

- estructura de control while.

Para evaluar el desarrollo de la actividad se hacen uso de procedimientos de evaluación. Estos procedimientos de evaluación miden la adquisición de las competencias por parte del alumnado utilizando los denominados criterios de evaluación.

A continuación se describen los procedimientos de evaluación con sus criterios asociados:

Tipo	Nombre	Criterios evaluados (peso)
Pruebas de ejecución	Reto 2	3.1.- Comprender, analizar y descomponer problemas, reconociendo en ellos los patrones, secuencias y condiciones que permitan formular soluciones a los mismos en forma de algoritmo. (**1**) 3.2.- Conocer y elaborar representaciones gráficas de algoritmos mediante diagramas de flujo. Respetado las reglas de construcción, la simbología adecuada y las estructuras de control oportunas, sin perder de vista la eficacia y eficiencia en su diseño. (**1**) 3.3.- Traducir, interpretar y expresar los distintos pasos que va a realizar un programa, partiendo o no de diagramas de flujo, del modo más próximo posible a un lenguaje de programación, mediante pseudocódigo. (**1**) 4.1.- Comprender los modos de almacenamiento, trasmisión, representación y codificación de la información, así como la cuantificación de la misma. (**1**) 4.2.- Diferenciar algoritmo de programa informático, comprender la necesidad y el porqué de la diversidad de lenguajes informáticos existentes, así como sus tipos, para poder reconocer el ámbito de aplicación y objeto de estos. (**1**) 4.3.- Conocer la estructura de un lenguaje de programación de alto nivel, los tipos de datos, operadores y estructuras de control de los que se sirve, para poder ser capaz de idear, traducir y desarrollar programas informáticos. (**1**) 5.1.- Adoptar protocolos efectivos de análisis y comprobación a las soluciones computacionales desarrolladas. (**1**) 5.2.- Rediseñar y proponer mejoras a programas propios o ajenos que contribuyan a mejorar la solución técnica, valorando aspectos medioambientales y éticos si los hubiera. (**1**)

Examen/ Reto 3: Semáforo

Se plantea la simulación de un sistema prograd o/automatiado de uso habitual, como ejemplo se propone un cruce gobernado por semáforo pero podría variar. El alumno debe simular el hardware y desarrollar el código de control que responda a una serie de premisas de funcionamiento y operación.

Para evaluar el desarrollo de la actividad se hacen uso de procedimientos de evaluación. Estos procedimientos de evaluación miden la adquisición de las competencias por parte del alumnado utilizando los denominados criterios de evaluación.

A continuación se describen los procedimientos de evaluación con sus criterios asociados:

Tipo	Nombre	Criterios evaluados (peso)

Tipo	Nombre	Criterios evaluados (peso)
Examen tradicional/Prueba objetiva/competencial	Exmaen / Reto 3	3.1.- Comprender, analizar y descomponer problemas, reconociendo en ellos los patrones, secuencias y condiciones que permitan formular soluciones a los mismos en forma de algoritmo. (**1**) 3.2.- Conocer y elaborar representaciones gráficas de algoritmos mediante diagramas de flujo. Respetado las reglas de construcción, la simbología adecuada y las estructuras de control oportunas, sin perder de vista la eficacia y eficiencia en su diseño. (**1**) 3.3.- Traducir, interpretar y expresar los distintos pasos que va a realizar un programa, partiendo o no de diagramas de flujo, del modo más próximo posible a un lenguaje de programación, mediante pseudocódigo. (**1**) 4.1.- Comprender los modos de almacenamiento, trasmisión, representación y codificación de la información, así como la cuantificación de la misma. (**1**) 4.2.- Diferenciar algoritmo de programa informático, comprender la necesidad y el porqué de la diversidad de lenguajes informáticos existentes, así como sus tipos, para poder reconocer el ámbito de aplicación y objeto de estos. (**1**) 4.3.- Conocer la estructura de un lenguaje de programación de alto nivel, los tipos de datos, operadores y estructuras de control de los que se sirve, para poder ser capaz de idear, traducir y desarrollar programas informáticos. (**1**) 5.1.- Adoptar protocolos efectivos de análisis y comprobación a las soluciones computacionales desarrolladas. (**1**) 5.2.- Rediseñar y proponer mejoras a programas propios o ajenos que contribuyan a mejorar la solución técnica, valorando aspectos medioambientales y éticos si los hubiera. (**1**)

2.- DISEÑO DE VIDEOJUEGOS 2D/3D CON GODOT (30 PERIODOS)

Esta unidad de programación está compuesta por 1 situaciones de aprendizaje que son descritas a continuación.

DISEÑO DE VIDEOJUEGOS CON GODOT

Descripción y saberes básicos de la situación de aprendizaje, integrando metodologías:

Desarrollo de videojuegos sencillos mediante el motor de vídeo juegos "Godot". En esta primera situación de aprendizaje se pretende familiarizar al alumnado con el entorno de desarrollo y la terminología propia del sistema, así como al código de programación empleado "GdScript". Por otro lado, es necesario estructurar bien el trabajo y desarrollo del proyecto, por lo que la gestión de proyectos tecnológicos y el uso de herramientas digitales asociadas al proyecto se vuelven un factor clave.

1. Gestión de proyectos tecnológicos.

- Fases de un proyecto. El proyecto de desarrollo de software.
- Estructuras organizativas de grupos de trabajo.
- Metodologías de gestión de proyectos.

2. Método computacional.

- Organización de la información y tipos de estructuras de datos.
- Descomposición de problemas y secuenciación.
- Generalización y reconocimiento de patrones.
- La abstracción como proceso específico, mediante representaciones simplificadas de los sistemas.
- Diseño mediante diagramas de flujos y pseudocódigo. La iteración, análisis recursivo, diseño iterativo, inducción.

3. Desarrollo de programas y aplicaciones.

- Representación binaria de la información: el bit, el byte. Codificación binaria. Sistemas de numeración. Representación hexadecimal. Códigos de Entrada/Salida. Medidas de la Información.
- Almacenamiento, transmisión y tratamiento básico de la información en binario: números, texto, imágenes, ficheros.
- Lenguajes de programación: Definición. Tipos de lenguajes. Lenguajes interpretados y compilados.
- Estructura de un programa informático y elementos básicos del lenguaje: Tipos básicos de datos.
- Constantes y variables. Operadores y expresiones. Comentarios. Estructuras de control.
- Condicionales.
- Iterativas. Estructuras de datos. Funciones y bibliotecas de funciones. Manipulación de archivos.

Producto solicitado a los alumnos en la situación de aprendizaje:

Juego 2D/3D

Competencias específicas que se van a trabajar en esta situación de aprendizaje:

3.- Organizar y analizar la información de un modo lógico, desde el punto de vista del pensamiento computacional, formulando y abordando problemas de forma que sus soluciones puedan ser representadas como secuencias de instrucciones y algoritmos.

4.- Identificar problemas o tareas susceptibles de ser resueltos u automatizados por medios informáticos. Discernir y concretar los requisitos del programa, idear la arquitectura del mismo y llevarlo a cabo mediante el uso lenguajes de programación de alto nivel, para posteriormente comprobar que se cumple con las especificaciones que previamente se establecieron en el análisis de requisitos.

5.- Comprobar, rediseñar y proponer mejoras a soluciones computacionales propias o ajenas, de forma individual o en grupo, estableciendo la evaluación y el análisis de satisfacción del resultado final en función de criterios funcionales, así como éticos, cívicos y medioambientales para depurar el funcionamiento mediante la detección de errores o la necesidad de incluir posibles mejoras al diseño.

En esta situación de aprendizaje se van a llevar a cabo (al menos) 3 actividades:

Videojuegos 2D

Recrear vídeojuegos clásicos (por ejemplo Invaders). El objeto es dar los primeros pasos en este nuevo mundo y familiarizarnos con los conceptos, el interfaz de usuario y workflow que Godot requiere.

Para evaluar el desarrollo de la actividad se hacen uso de procedimientos de evaluación. Estos procedimientos de evaluación miden la adquisición de las competencias por parte del alumnado utilizando los denominados criterios de evaluación.

A continuación se describen los procedimientos de evaluación con sus criterios asociados:

Tipo	Nombre	Criterios evaluados (peso)
Pruebas de ejecución	Videojuegos 2D	3.1.- Comprender, analizar y descomponer problemas, reconociendo en ellos los patrones, secuencias y condiciones que permitan formular soluciones a los mismos en forma de algoritmo. (**1**) 3.2.- Conocer y elaborar representaciones gráficas de algoritmos mediante diagramas de flujo. Respetado las reglas de construcción, la simbología adecuada y las estructuras de control oportunas, sin perder de vista la eficacia y eficiencia en su diseño. (**1**) 3.3.- Traducir, interpretar y expresar los distintos pasos que va a realizar un programa, partiendo o no de diagramas de flujo, del modo más próximo posible a un lenguaje de programación, mediante pseudocódigo. (**1**) 4.1.- Comprender los modos de almacenamiento, trasmisión, representación y codificación de la información, así como la cuantificación de la misma. (**1**) 4.2.- Diferenciar algoritmo de programa informático, comprender la necesidad y el porqué de la diversidad de lenguajes informáticos existentes, así como sus tipos, para poder reconocer el ámbito de aplicación y objeto de estos. (**1**) 4.3.- Conocer la estructura de un lenguaje de programación de alto nivel, los tipos de datos, operadores y estructuras de control de los que se sirve, para poder ser capaz de idear, traducir y desarrollar programas informáticos. (**1**) 5.1.- Adoptar protocolos efectivos de análisis y comprobación a las soluciones computacionales desarrolladas. (**1**) 5.2.- Rediseñar y proponer mejoras a programas propios o ajenos que contribuyan a mejorar la solución técnica, valorando aspectos medioambientales y éticos si los hubiera. (**1**)

Videojuego 3D

Se avanza en la complejidad del desarrollo de videojuegos, añadiendo una dimensión más y por tanto nuevos nodos y operandores de código, sentando las bases para la siguiente parte del curso.

Para evaluar el desarrollo de la actividad se hacen uso de procedimientos de evaluación. Estos procedimientos de evaluación miden la adquisición de las competencias por parte del alumnado utilizando los denominados criterios de evaluación.

A continuación se describen los procedimientos de evaluación con sus criterios asociados:

Tipo	Nombre	Criterios evaluados (peso)
Pruebas de ejecución	Videojuego 3D	3.1.- Comprender, analizar y descomponer problemas, reconociendo en ellos los patrones, secuencias y condiciones que permitan formular soluciones a los mismos en forma de algoritmo. (**1**) 3.2.- Conocer y elaborar representaciones gráficas de algoritmos mediante diagramas de flujo. Respetado las reglas de construcción, la simbología adecuada y las estructuras de control oportunas, sin perder de vista la eficacia y eficiencia en su diseño. (**1**) 3.3.- Traducir, interpretar y expresar los distintos pasos que va a realizar un programa, partiendo o no de diagramas de flujo, del modo más próximo posible a un lenguaje de programación, mediante pseudocódigo. (**1**) 4.1.- Comprender los modos de almacenamiento, trasmisión, representación y codificación de la información, así como la cuantificación de la misma. (**1**) 4.2.- Diferenciar algoritmo de programa informático, comprender la necesidad y el porqué de la diversidad de lenguajes informáticos existentes, así como sus tipos, para poder reconocer el ámbito de aplicación y objeto de estos. (**1**) 4.3.- Conocer la estructura de un lenguaje de programación de alto nivel, los tipos de datos, operadores y estructuras de control de los que se sirve, para poder ser capaz de idear, traducir y desarrollar programas informáticos. (**1**) 5.1.- Adoptar protocolos efectivos de análisis y comprobación a las soluciones computacionales desarrolladas. (**1**) 5.2.- Rediseñar y proponer mejoras a programas propios o ajenos que contribuyan a mejorar la solución técnica, valorando aspectos medioambientales y éticos si los hubiera. (**1**)

Examen

Al alumno se le ofrece un videojuego terminado (puediera ser uno de los desarrollados en clase de forma conjunta) pero no 100% operativo. El código incluye errores que evitan que funcione como debería (ejm: cuando el jugador debería perder una vida, se le suma una en su lugar).

Para evaluar el desarrollo de la actividad se hacen uso de procedimientos de evaluación. Estos procedimientos de evaluación miden la adquisición de las competencias por parte del alumnado utilizando los denominados criterios de evaluación.

A continuación se describen los procedimientos de evaluación con sus criterios asociados:

Tipo	Nombre	Criterios evaluados (peso)

Tipo	Nombre	Criterios evaluados (peso)
Examen tradicional/Prueba objetiva/competencial	Examen videojuego 2D/3D	3.1.- Comprender, analizar y descomponer problemas, reconociendo en ellos los patrones, secuencias y condiciones que permitan formular soluciones a los mismos en forma de algoritmo. (**1**) 3.2.- Conocer y elaborar representaciones gráficas de algoritmos mediante diagramas de flujo. Respetado las reglas de construcción, la simbología adecuada y las estructuras de control oportunas, sin perder de vista la eficacia y eficiencia en su diseño. (**1**) 3.3.- Traducir, interpretar y expresar los distintos pasos que va a realizar un programa, partiendo o no de diagramas de flujo, del modo más próximo posible a un lenguaje de programación, mediante pseudocódigo. (**1**) 4.1.- Comprender los modos de almacenamiento, trasmisión, representación y codificación de la información, así como la cuantificación de la misma. (**1**) 4.2.- Diferenciar algoritmo de programa informático, comprender la necesidad y el porqué de la diversidad de lenguajes informáticos existentes, así como sus tipos, para poder reconocer el ámbito de aplicación y objeto de estos. (**1**) 4.3.- Conocer la estructura de un lenguaje de programación de alto nivel, los tipos de datos, operadores y estructuras de control de los que se sirve, para poder ser capaz de idear, traducir y desarrollar programas informáticos. (**1**) 5.1.- Adoptar protocolos efectivos de análisis y comprobación a las soluciones computacionales desarrolladas. (**1**) 5.2.- Rediseñar y proponer mejoras a programas propios o ajenos que contribuyan a mejorar la solución técnica, valorando aspectos medioambientales y éticos si los hubiera. (**1**)

3.- DISEÑO DE VIDEOJUEGOS VR CON GODOT (30 PERIODOS)

Esta unidad de programación está compuesta por 1 situaciones de aprendizaje que son descritas a continuación.

DISEÑO DE VIDEOJUGOS CON VR Y RELAIDAD AUMENTADA/MIXTA

Descripción y saberes básicos de la situación de aprendizaje, integrando metodologías:

Expandir las posibilidades de la programación y el diseño de videojuegos llevándolos al siguiente nivel, la realidad virtual. El alumno tendrá la posibildad de entender los rudimientos de estos sistemas que si bien ya llevan un tiempo con nosotros, no han calado en la vida cotidiana tanto como otros, aunque presumiblemente lo hagan en un futuro.

El uso de los visores de realidad virtual y de la orientación gamer de estas situaciones de aprendizaje supone un factor motivador que compensa las dificultad que conlleva el desarrollo de código para las mismas, y paralelamente dota a los alumnos de conocimientos extrapolables a otros ramas de la programación y el pensamiento computacional.

2. Método computacional.

- Organización de la información y tipos de estructuras de datos.
- Descomposición de problemas y secuenciación.
- Generalización y reconocimiento de patrones.
- La abstracción como proceso específico, mediante representaciones simplificadas de los sistemas.
- Diseño mediante diagramas de flujos y pseudocódigo. La iteración, análisis recursivo, diseño iterativo, inducción.

3. Desarrollo de programas y aplicaciones.

- Representación binaria de la información: el bit, el byte. Codificación binaria. Sistemas de numeración. Representación hexadecimal. Códigos de Entrada/Salida. Medidas de la Información.
- Almacenamiento, transmisión y tratamiento básico de la información en binario: números, texto, imágenes, ficheros.
- Lenguajes de programación: Definición. Tipos de lenguajes. Lenguajes interpretados y compilados.
- Estructura de un programa informático y elementos básicos del lenguaje: Tipos básicos de datos.
- Constantes y variables. Operadores y expresiones. Comentarios. Estructuras de control.
- Condicionales.
- Iterativas. Estructuras de datos. Funciones y bibliotecas de funciones. Manipulación de archivos.
- Orientación a objetos: Clases. Objetos. Constructores. Metodologías de desarrollo de software.
- Desarrollo web: Lenguaje de marcas de hipertexto (HTML), estructura, etiquetas y atributos, formularios, multimedia y gráficos. Hoja de estilo en cascada (CSS), diseño adaptativo y plantillas.
- Computación física. Robótica: Características principales de los robots: cuerpo, control y comportamiento. Microcontroladores, entrada/salida, sensores, actuadores, RFID.

Producto solicitado a los alumnos en la situación de aprendizaje:

Videojuego en realidad virtual/ aumentada/ mixta.

Competencias específicas que se van a trabajar en esta situación de aprendizaje:

1.- Planificar, desarrollar y gestionar proyectos de índole computacional y de desarrollo de software, siguiendo los pasos de la metodología tecnológica, de manera individual o cooperando cuando sea necesario, para dar soluciones a aspectos relacionados con la sociedad en situaciones de la vida cotidiana, profesional o industrial.

2.- Plantear y definir necesidades, observando, buscando y seleccionando la información adecuada proveniente de diversas fuentes, de manera crítica y segura, aplicando procesos de investigación, métodos de análisis de productos y experimentando con herramientas de simulación, para determinar problemas tecnológicos e iniciar procesos de creación de soluciones a partir de la información obtenida.

3.- Organizar y analizar la información de un modo lógico, desde el punto de vista del pensamiento computacional, formulando y abordando problemas de forma que sus soluciones puedan ser representadas como secuencias de instrucciones y algoritmos.

4.- Identificar problemas o tareas susceptibles de ser resueltos u automatizados por medios informáticos. Discernir y concretar los requisitos del programa, idear la arquitectura del mismo y llevarlo a cabo mediante el uso lenguajes de programación de alto nivel, para posteriormente comprobar que se cumple con las especificaciones que previamente se establecieron en el análisis de requisitos.

5.- Comprobar, rediseñar y proponer mejoras a soluciones computacionales propias o ajenas, de forma individual o en grupo, estableciendo la evaluación y el análisis de satisfacción del resultado final en función de criterios funcionales, así como éticos, cívicos y medioambientales para depurar el funcionamiento mediante la detección de errores o la necesidad de incluir posibles mejoras al diseño.

En esta situación de aprendizaje se van a llevar a cabo (al menos) 2 actividades:

Diseño de videojuego VR. Introducción a los visores VR meta Quest 3 Relación de los visores con Godot. Métodos de programación asociados a VR.

Para evaluar el desarrollo de la actividad se hacen uso de procedimientos de evaluación. Estos procedimientos de evaluación miden la adquisición de las competencias por parte del alumnado utilizando los denominados criterios de evaluación.

A continuación se describen los procedimientos de evaluación con sus criterios asociados:

Tipo	Nombre	Criterios evaluados (peso)

Tipo	Nombre	Criterios evaluados (peso)
Revisión del cuaderno o producto	Diseño de videojuego VR	1.1.- Diseñar la planificación, ejecución, seguimiento y control de las fases de un proyecto, realizando una coordinación adecuada y aplicando metodologías propias de la gestión de proyectos. (**1**) 1.2.- Gestionar con diligencia grupos de trabajo, estableciendo la estructura organizativa conveniente, determinando responsabilidades y asignando recursos de una manera eficiente, adoptando empatía y liderazgo. (**1**) 1.3.- Cooperar dentro de un grupo de proyecto, asumiendo responsablemente una función concreta y valorando constructivamente aportaciones del resto de miembros del grupo. (**1**) 2.1.- Observar, plantear y definir problemas o necesidades sobre fenómenos sociales cotidianos y/o profesionales de los diferentes sectores productivos, que puedan ser respondidas utilizando métodos tecnológicos. (**1**) 2.2.- Buscar y contrastar información procedente de diferentes fuentes de manera crítica, evaluando su fiabilidad y pertinencia. (**1**) 2.3.- Comprender, examinar y analizar objetos y sistemas, empleando el proceso tecnológico y utilizando herramientas de simulación en la construcción de conocimiento. (**1**) 2.4.- Adoptar medidas preventivas, éticas y críticas para la protección de los datos, la salud y el medioambiente, identificando riesgos en los problemas planteados. (**1**) 3.1.- Comprender, analizar y descomponer problemas, reconociendo en ellos los patrones, secuencias y condiciones que permitan formular soluciones a los mismos en forma de algoritmo. (**2**) 3.2.- Conocer y elaborar representaciones gráficas de algoritmos mediante diagramas de flujo. Respetado las reglas de construcción, la simbología adecuada y las estructuras de control oportunas, sin perder de vista la eficacia y eficiencia en su diseño. (**2**) 3.3.- Traducir, interpretar y expresar los distintos pasos que va a realizar un programa, partiendo o no de diagramas de flujo, del modo más próximo posible a un lenguaje de programación, mediante pseudocódigo. (**2**) 4.1.- Comprender los modos de almacenamiento, trasmisión, representación y codificación de la información, así como la cuantificación de la misma. (**2**) 4.2.- Diferenciar algoritmo de programa informático, comprender la necesidad y el porqué de la diversidad de lenguajes informáticos existentes, así como sus tipos, para poder reconocer el ámbito de aplicación y objeto de estos. (**2**) 4.3.- Conocer la estructura de un lenguaje de programación de alto nivel, los tipos de datos, operadores y estructuras de control de los que se sirve, para poder ser capaz de idear, traducir y desarrollar programas informáticos. (**2**) 5.1.- Adoptar protocolos efectivos de análisis y comprobación a las soluciones computacionales desarrolladas. (**2**) 5.2.- Rediseñar y proponer mejoras a programas propios o ajenos que contribuyan a mejorar la solución técnica, valorando aspectos medioambientales y éticos si los hubiera. (**2**)

ANEXO I - CÁLCULO DE CALIFICACIONES

LISTADO DE COMPETENCIAS ESPECÍFICAS

La superación de Programación (Optativa) implica la adquisición de una serie de competencias específicas. Cada una de estas competencias específicas contribuirá en parte a la calificación que finalmente obtendrán sus alumnos.

No obstante, es posible que su departamento considere que una competencia específica tenga más importancia que otras en la calificación final. Esta importancia la puede fijar introduciendo un "peso" a cada competencia específica; este peso se representa por un número asociado a dicha competencia. Cuanto mayor es el peso (el número asignado) mayor es la importancia de la competencia.

A través de los criterios de evaluación se valora el grado de adquisición de cada competencia específica; la media ponderada de esas valoraciones será la calificación que el alumnado obtendrá en Programación (Optativa).

Competencias específicas	Peso
Programación (Optativa)	
1.- Planificar, desarrollar y gestionar proyectos de índole computacional y de desarrollo de software, siguiendo los pasos de la metodología tecnológica, de manera individual o cooperando cuando sea necesario, para dar soluciones a aspectos relacionados con la sociedad en situaciones de la vida cotidiana, profesional o industrial.	1
2.- Plantear y definir necesidades, observando, buscando y seleccionando la información adecuada proveniente de diversas fuentes, de manera crítica y segura, aplicando procesos de investigación, métodos de análisis de productos y experimentando con herramientas de simulación, para determinar problemas tecnológicos e iniciar procesos de creación de soluciones a partir de la información obtenida.	1
3.- Organizar y analizar la información de un modo lógico, desde el punto de vista del pensamiento computacional, formulando y abordando problemas de forma que sus soluciones puedan ser representadas como secuencias de instrucciones y algoritmos.	5
4.- Identificar problemas o tareas susceptibles de ser resueltos u automatizados por medios informáticos. Discernir y concretar los requisitos del programa, idear la arquitectura del mismo y llevarlo a cabo mediante el uso lenguajes de programación de alto nivel, para posteriormente comprobar que se cumple con las especificaciones que previamente se establecieron en el análisis de requisitos.	5
5.- Comprobar, rediseñar y proponer mejoras a soluciones computacionales propias o ajenas, de forma individual o en grupo, estableciendo la evaluación y el análisis de satisfacción del resultado final en función de criterios funcionales, así como éticos, cívicos y medioambientales para depurar el funcionamiento mediante la detección de errores o la necesidad de incluir posibles mejoras al diseño.	5

La calificación de Programación (Optativa) se calculará a través de la siguiente media ponderada:

$$\text{calificación Programación (Optativa)} = \frac{CE1 \times 1 + CE2 \times 1 + CE3 \times 5 + CE4 \times 5 + CE5 \times 5}{1 + 1 + 5 + 5 + 5}$$

En la anterior fórmula, CE1 es la calificación que un alumno obtiene en la competencia específica 1,
En la anterior fórmula, CE2 es la calificación que un alumno obtiene en la competencia específica 2,
...
CEn sería la calificación obtenida en la competencia específica "n".

PESO ASOCIADO A CADA CRITERIO DE EVALUACIÓN

Para concretar el nivel de adquisición de cada competencia específica, se utilizarán una serie de criterios de

evaluación. Así pues, las competencias no son evaluadas directamente; la evaluación se hace a través los citados criterios de evaluación; que a su vez servirán de referencia para generar la calificación obtenida por el alumnado.

Cada criterio de evaluación puede tener, a su vez, un "peso" que determina su contribución ponderada a la valoración del grado de adquisición de la competencia específica.

La calificación de cada competencia específica será la media ponderada de las calificaciones que usted otorgue a cada alumno en cada criterio de evaluación.

Competencias específicas con sus criterios de evaluación asociados	Peso
1.- Planificar, desarrollar y gestionar proyectos de índole computacional y de desarrollo de software, siguiendo los pasos de la metodología tecnológica, de manera individual o cooperando cuando sea necesario, para dar soluciones a aspectos relacionados con la sociedad en situaciones de la vida cotidiana, profesional o industrial.	
1.1.- Diseñar la planificación, ejecución, seguimiento y control de las fases de un proyecto, realizando una coordinación adecuada y aplicando metodologías propias de la gestión de proyectos.	1
1.2.- Gestionar con diligencia grupos de trabajo, estableciendo la estructura organizativa conveniente, determinando responsabilidades y asignando recursos de una manera eficiente, adoptando empatía y liderazgo.	1
1.3.- Cooperar dentro de un grupo de proyecto, asumiendo responsablemente una función concreta y valorando constructivamente aportaciones del resto de miembros del grupo.	1
2.- Plantear y definir necesidades, observando, buscando y seleccionando la información adecuada proveniente de diversas fuentes, de manera crítica y segura, aplicando procesos de investigación, métodos de análisis de productos y experimentando con herramientas de simulación, para determinar problemas tecnológicos e iniciar procesos de creación de soluciones a partir de la información obtenida.	
2.1.- Observar, plantear y definir problemas o necesidades sobre fenómenos sociales cotidianos y/o profesionales de los diferentes sectores productivos, que puedan ser respondidas utilizando métodos tecnológicos.	1
2.2.- Buscar y contrastar información procedente de diferentes fuentes de manera crítica, evaluando su fiabilidad y pertinencia.	1
2.3.- Comprender, examinar y analizar objetos y sistemas, empleando el proceso tecnológico y utilizando herramientas de simulación en la construcción de conocimiento.	1
2.4.- Adoptar medidas preventivas, éticas y críticas para la protección de los datos, la salud y el medioambiente, identificando riesgos en los problemas planteados.	1
3.- Organizar y analizar la información de un modo lógico, desde el punto de vista del pensamiento computacional, formulando y abordando problemas de forma que sus soluciones puedan ser representadas como secuencias de instrucciones y algoritmos.	
3.1.- Comprender, analizar y descomponer problemas, reconociendo en ellos los patrones, secuencias y condiciones que permitan formular soluciones a los mismos en forma de algoritmo.	1
3.2.- Conocer y elaborar representaciones gráficas de algoritmos mediante diagramas de flujo. Respetado las reglas de construcción, la simbología adecuada y las estructuras de control oportunas, sin perder de vista la eficacia y eficiencia en su diseño.	1
3.3.- Traducir, interpretar y expresar los distintos pasos que va a realizar un programa, partiendo o no de diagramas de flujo, del modo más próximo posible a un lenguaje de programación, mediante pseudocódigo.	1
4.- Identificar problemas o tareas susceptibles de ser resueltos u automatizados por medios informáticos. Discernir y concretar los requisitos del programa, idear la arquitectura del mismo y llevarlo a cabo mediante el uso lenguajes de programación de alto nivel, para posteriormente comprobar que se cumple con las especificaciones que previamente se establecieron en el análisis de requisitos.	
4.1.- Comprender los modos de almacenamiento, trasmisión, representación y codificación de la información, así como la cuantificación de la misma.	1
4.2.- Diferenciar algoritmo de programa informático, comprender la necesidad y el porqué de la diversidad de lenguajes informáticos existentes, así como sus tipos, para poder reconocer el ámbito de aplicación y objeto de estos.	1
4.3.- Conocer la estructura de un lenguaje de programación de alto nivel, los tipos de datos, operadores y estructuras de control de los que se sirve, para poder ser capaz de idear, traducir y desarrollar programas informáticos.	1

Competencias específicas con sus criterios de evaluación asociados	Peso
5.- Comprobar, rediseñar y proponer mejoras a soluciones computacionales propias o ajenas, de forma individual o en grupo, estableciendo la evaluación y el análisis de satisfacción del resultado final en función de criterios funcionales, así como éticos, cívicos y medioambientales para depurar el funcionamiento mediante la detección de errores o la necesidad de incluir posibles mejoras al diseño.	
5.1.- Adoptar protocolos efectivos de análisis y comprobación a las soluciones computacionales desarrolladas.	1
5.2.- Rediseñar y proponer mejoras a programas propios o ajenos que contribuyan a mejorar la solución técnica, valorando aspectos medioambientales y éticos si los hubiera.	2

A modo de ejemplo, la calificación de la competencia específica 5 se calculará a través de la siguiente media ponderada:

$$\text{calificación CE5} = \frac{\text{CEV5.1} \times 1 + \text{CEV5.2} \times 2}{1 + 2}$$

En la anterior fórmula, CEV5.1 es la calificación que un alumno ha obtenido al evaluar el criterio de evaluación 5.1,
en general, CEV5.n sería la calificación obtenida en el criterio de evaluación "n".