

DESARROLLO CURRICULAR DEL MÓDULO

PROGRAMACIÓN EN LENGUAJES ESTRUCTURADOS

CICLO FORMATIVO DE GRADO SUPERIOR

DESARROLLO DE APLICACIONES INFORMÁTICAS

I. E. S. SAN VICENTE 2009/10

<i><u>1. INTRODUCCIÓN</u></i>	<i><u>3</u></i>
<i><u>2. CAPACIDADES TERMINALES</u></i>	<i><u>3</u></i>
<i><u>3. OBJETIVOS</u></i>	<i><u>3</u></i>
<i><u>4. ORGANIZACIÓN DE LOS CONTENIDOS</u></i>	<i><u>6</u></i>
<i><u>5. ELEMENTOS CURRICULARES DE CADA UNIDAD</u></i>	<i><u>9</u></i>
<i><u>6. TEMPORALIZACIÓN</u></i>	<i><u>31</u></i>
<i><u>7. BIBLIOGRAFÍA</u></i>	<i><u>32</u></i>
<i><u>8. PLANTEAMIENTO DE LA ATENCIÓN A LA DIVERSIDAD</u></i>	<i><u>34</u></i>
<i><u>9. CRITERIOS DE EVALUACIÓN</u></i>	<i><u>35</u></i>
<i><u>10. ACTIVIDADES COMPLEMENTARIAS Y EXTRAESCOLARES.</u></i>	<i><u>37</u></i>
<i><u>11. MATERIALES Y RECURSOS DIDÁCTICOS.</u></i>	<i><u>37</u></i>

1. INTRODUCCIÓN

El módulo de **Programación en Lenguajes Estructurados** tiene una duración de 384 horas totales (12 horas semanales) y se encuadra en el primer curso del Ciclo formativo de grado superior correspondiente al título de Desarrollo de Aplicaciones Informáticas.

2. CAPACIDADES TERMINALES

La referencia del sistema productivo de este Módulo la encontramos en el Real Decreto 1661/1994, que especifica las siguientes capacidades terminales:

- Elaborar programas utilizando lenguajes estructurados, cumpliendo con las especificaciones establecidas en el diseño.
- Evaluar el funcionamiento de las aplicaciones mediante la realización de pruebas de los diferentes módulos de programación.
- Elaborar la documentación completa relativa a las aplicaciones desarrolladas.
- Adaptar aplicaciones a partir de nuevos requerimientos establecidos en el diseño.

3. OBJETIVOS

Los objetivos específicos del módulo se obtienen a partir de las capacidades terminales anteriores, considerando el contexto en el que se impartirá el módulo.

CT1	Elaborar programas utilizando lenguajes estructurados, cumpliendo con las especificaciones establecidas en el diseño.
OM01	Aplicar estrategias de programación estructurada y modular, y de programación orientada a objetos para la resolución de problemas con independencia del lenguaje de programación a utilizar.
OM02	Identificar estructuras de datos necesarias para la resolución del problema con un lenguaje estructurado.
OM03	Codificar un módulo de programación en lenguaje C#, empleando las construcciones modulares proporcionadas por dichos lenguajes (funciones, clases, objetos, etc.) y definiendo estructuras de datos apropiadas.
OM04	Documentar el código desarrollado con comentarios significativos, concisos y legibles.
OM05	Integrar y enlazar módulos de programación y librerías.

OM06	Obtener código ejecutable para sistemas operativos Windows y Linux, empleando para ello las distintas opciones de los compiladores, enlazadores, y gestores de configuraciones.
OM07	Depurar los módulos de programación manejando herramientas específicas.

CT2	Evaluar el funcionamiento de las aplicaciones mediante la realización de pruebas de los diferentes módulos de programación.
OM08	Planificar la realización de una fase de pruebas a partir de las especificaciones establecidas en el diseño y de los resultados esperados para cada módulo.
OM09	Realizar pruebas para cada módulo de una aplicación y pruebas de integración, detectando errores en la funcionalidad o en la presentación (formato) de los datos de entrada y salida.
OM10	Medir los rendimientos de la aplicación y evaluar la eficiencia de las prestaciones de la aplicación y el consumo de recursos.
OM11	Provocar y verificar los diversos tratamientos de error.

CT3	Elaborar la documentación completa relativa a las aplicaciones desarrolladas.
OM12	Elaborar documentación útil sobre la arquitectura y algoritmos diseñados.
OM13	Documentar y describir las estructuras de datos utilizadas.
OM14	Redactar guías de uso de las aplicaciones.

CT4	Adaptar aplicaciones a partir de nuevos requerimientos establecidos en el diseño.
OM15	Identificar los datos y módulos de programación afectados por la modificación de los requerimientos.
OM16	Probar que los nuevos datos y módulos no producen pérdidas de eficiencia ni funcionalidad de la aplicación y satisfacen los nuevos requerimientos funcionales.
OM17	Actualizar la documentación con los cambios realizados sobre los módulos y estructuras de datos de la aplicación.

Además, los objetivos incluyen un conjunto de principios y normas que guían el comportamiento del alumno en el aula, y lo harán en su vida profesional y también en otros aspectos de la vida. Dichos principios y normas están relacionados con la educación en valores, riesgos laborales y explotación de las Tecnologías de la Información y Comunicación (TIC).

Educación en valores: moral y cívica

- Utilizar adecuadamente las credenciales en el acceso a los sistemas informáticos, no usurpando la identidad de otros usuarios.
- Observar la legislación vigente en relación a la protección de datos, no accediendo a datos personales ajenos para obtención de algún beneficio. Manejar software de manera legal, respetando las condiciones de la licencia correspondiente.
- Respetar las fechas de entrega de actividades y trabajos, y cuidar su adecuada presentación.
- Mostrar lealtad y respeto al grupo de trabajo, realizando aportaciones en todas actividades grupales, colaborando con ellos y resolviendo conflictos de manera dialogada y pacífica.
- Preservar la originalidad de las actividades individuales, no copiando el trabajo de los compañeros o cediendo los resultados para su copia.
- Mantener los materiales de trabajo compartidos en buenas condiciones, utilizándolos y tratándolos correctamente.

Educación en valores: ambiental

- Ahorrar energía, procurando desconectar máquinas y dispositivos que no estén en uso, así como iluminación artificial y aire acondicionado.
- Ahorrar papel, fomentando la utilización de información en formato electrónico en la medida de lo posible.
- Reciclar correctamente sistemas obsoletos y consumibles de impresión (tinta para impresoras).

Educación en valores: educación para el consumidor

- Adquirir productos informáticos basándose en la adecuada relación coste/beneficio.
- Planificar el aprovisionamiento de productos considerando la evolución tecnológica y la utilidad futura de los mismos.

Riesgos laborales

- Trabajar en las condiciones adecuadas en relación al mobiliario, iluminación, ruidos y temperatura.
- Mantener una postura correcta frente a los instrumentos de trabajo
- Considerar los efectos perjudiciales de la exposición prolongada a la radiación electromagnética producida por antenas y otros dispositivos.
- Fomentar hábitos saludables de descanso y ejercicio físico para prevenir la fatiga excesiva y dolores de cabeza, cuello y espalda.
- Observación de las normas de seguridad en los edificios: extintores, salidas de emergencia, entre otras.

Tecnologías de la Información y Comunicación

- Utilizar de manera efectiva herramientas de comunicación como los foros y el correo electrónico, empleando un estilo correcto en la redacción.
- Recurrir a bases de datos y otra información en Internet para la resolución de problemas relacionados con el módulo.
- Aprovechar las herramientas ofimáticas para mejorar las presentaciones públicas y la elaboración de documentación.

4. ORGANIZACIÓN DE LOS CONTENIDOS

4.1.- ESTRUCTURA DE LOS CONTENIDOS

Teniendo en cuenta la naturaleza del módulo y las capacidades terminales a él ligadas, su contenido será de tipo procedimental encaminado a conseguir las capacidades terminales del módulo.

La programación está formada por una relación de unidades de trabajo agrupadas bajo los siguientes bloques conceptuales que desarrollan diferentes áreas de programación.

BLOQUES

1. Metodología de la programación. El lenguaje de programación C# (inicial)

2. El lenguaje de programación C# (avanzado)
3. Mantenimiento de programas
4. Programación web

La finalidad de los objetivos que se persiguen con cada bloque son:

Bloque 1: Metodología de la programación. El lenguaje de programación C# (inicial)

El objetivo principal es que el alumno adquiera los conceptos básicos de la programación, mediante la adquisición de métodos y técnicas para la resolución de problemas así como las formas descriptoras en forma de pseudocódigo para la resolución de algoritmos que resuelvan esos problemas planteados, siguiendo un diseño modular y estructurado.

Se iniciará el lenguaje relacionando la sintaxis de las estructuras de datos simples, y las sentencias elementales de este lenguaje con respecto al pseudocódigo. Casi desde el primer día se empezará a trabajar en lenguaje C#, para evitar que una carga teórica inicial excesiva pueda hacer la asignatura tediosa para el alumno.

Bloque 2: El lenguaje de programación C# (avanzado)

El objetivo es presentar todo aquello que, considerado como importante para el desarrollo de programas en C#, no se ha visto hasta el momento. Permite además aplicar de forma conjunta lo que hasta ahora se ha visto en parcelas independientes, y todo aquello que por su grado de dificultad el profesor ha preferido guardar para cuando el alumno haya adquirido cierta destreza y grado de conocimiento en el manejo del lenguaje C#.

Bloque 3: Mantenimiento de programas

En este bloque se intentará enfrentar al alumno a la creación y/o adaptación de aplicaciones dirigidas a la mejora de programas y/o el sistema. El desarrollo de este bloque dependerá en gran medida del conocimiento que haya adquirido el alumno hasta este momento.

Bloque 4: Programación web

En este bloque se intentará enfrentar al alumno a la creación y/o adaptación de aplicaciones dirigidas a la mejora de programas y/o el sistema. El desarrollo de este bloque dependerá en gran medida del conocimiento que haya adquirido el alumno hasta este momento.

4.2.- RELACIÓN SECUENCIADA DE UNIDADES DE TRABAJO

La propuesta de organización de contenidos está constituida por una relación secuenciada de unidades de trabajo donde se integran y desarrollan al mismo tiempo, alrededor de los procedimientos, conceptos, enseñanzas de enseñanza-aprendizaje y criterios de evaluación.

Bloque 1: Metodología de la programación. El lenguaje de programación C# (inicial)

0. Conceptos básicos sobre programación
1. Toma de contacto con C#
2. Tipos de datos básicos
3. Estructuras de control
4. Arrays y estructuras
5. Ficheros
6. Funciones

Bloque 2: El lenguaje de programación C# (avanzado)

7. Punteros y gestión dinámica de memoria
8. Librerías de uso frecuente
9. Introducción a la programación orientada a objetos
10. Otras características avanzadas de C#

Bloque 3: Mantenimiento de programas

11. Ejercicios completos
12. Proyecto final en lenguaje C#

Bloque 4: Programación web

13. El lenguaje de descripción de páginas web HTML
14. Hojas de estilo CSS
15. Javascript
16. Introducción a PHP

4.3.- CONTENIDOS MÍNIMOS

Los contenidos mínimos que deben alcanzar los alumnos en el módulo de Fundamentos de Programación están establecidos en el Real Decreto del Título, y su referencia son las capacidades terminales que el alumno debe conseguir y sus correspondientes criterios de evaluación, que marcan los niveles de consecución aceptable de dichas capacidades terminales.

Los alumnos deben ser capaces de resolver cuestiones teóricas y prácticas que indiquen que han adquirido las capacidades terminales. Para ello deben demostrar que son capaces de realizar las actividades de Enseñanza/Aprendizaje y alcanzar los Criterios de Evaluación desarrollados en cada Unidad de Trabajo.

5. ELEMENTOS CURRICULARES DE CADA UNIDAD

5.1. PLANTEAMIENTO DE LAS UNIDADES TEMÁTICAS

UT.0. Conceptos básicos sobre programación

La UT.0. tiene como fin presentar al alumno los conceptos básicos sobre la programación de tal manera que comience a familiarizarse con los términos, entornos, materiales y finalidades del Módulo completo. Es una Unidad eminentemente conceptual.

UT.1. Toma de contacto con C#

Esta unidad acerca al alumno al lenguaje C#: la estructura de un programa fundamental, como acceder a pantalla, como mostrar textos prefijados y números enteros, cómo leer números desde el teclado y realizar operaciones aritméticas básicas.

UT.2. Tipos de datos básicos

Una vez que el alumno tiene soltura con los números enteros, se le introducen otros tipos de datos imprescindibles, como los números reales y los caracteres. También se comenzará a trabajar con cadenas de texto.

UT.3. Estructuras de control

En esta unidad se muestra al alumno la forma de comprobar condiciones y de crear bloques de instrucciones que se repitan dentro de un programa.

UT.4. Arrays y estructuras

Se muestran al alumno las estructuras de datos estáticas y su manejo, tanto en lo que se refiere a los arrays unidimensionales, bidimensionales, multidimensionales, como a las estructuras, simples o anidadas. También se profundiza algo más en el uso de las cadenas de texto.

UT.5. Ficheros

Se muestra al alumno la forma de conservar la información, ya sea en fichero de texto, guardando datos tipificados (por ejemplo, el contenido de un struct) o datos de cualquier otro tipo, así como la forma de recuperar esa información que se había guardado, y de comprobar los errores que pueden ocurrir en el proceso. Esto servirá también para introducir el concepto de “excepciones”.

UT.6. Funciones

El uso de funciones es una característica peculiar y muy importante en el lenguaje C#, pues permite realizar un desarrollo modular del programa al tiempo que facilita su mantenimiento y futuro uso en otras aplicaciones.

UT.7. Punteros y gestión dinámica de memoria

En esta unidad se introducirá al alumno en la problemática de la gestión dinámica de memoria, los punteros y las “colecciones” dinámicas existentes en los lenguajes de creación reciente.

UT.8. Librerías de uso frecuente

El alumno necesitará con frecuencia recurrir a bibliotecas de funciones ya existentes y que todavía no se le han presentado, por ejemplo para realizar operaciones matemáticas y generar números aleatorios, o para el acceso avanzado a la pantalla de texto (consola). Esas bibliotecas serán utilizadas en esta Unidad.

UT.9. Introducción a la programación orientada a objetos

Se introducirá al alumno en conceptos como “objeto” y “clase”, así como su definición y uso desde C#. También se formalizarán conceptos que hasta esta unidad habían quedado en el aire, como los especificadores de acceso.

UT.10. Otras características avanzadas de C#

En esta unidad se le presentarán otras características de uso menos frecuente, como los operadores a nivel de bits, el operador coma y otras posibilidades menos habituales.

UT.11. Ejercicios completos

Esta unidad trata de enfrentar al alumno con creación de aplicaciones y la adaptación de aplicaciones que ya están hechas para satisfacer nuevos requerimientos. Para el desarrollo de esta Unidad será definitivo el grado de conocimiento que haya adquirido el alumno hasta este momento, así como la ayuda que pueda prestar el profesor.

UT.12. Proyecto final en lenguaje C#

En esta unidad, los alumnos tendrán que enfrentarse a un proyecto de programación de

mayor complejidad, y que deberá ser realizado en grupo junto con alguno(s) de sus compañeros.

UT.13. El lenguaje de descripción de páginas web HTML

Esta unidad introduce a los alumnos en la creación de páginas Web utilizando el lenguaje de descripción HTML.

UT.14. Hojas de estilo CSS

En esta unidad, los alumnos aprenderán a hacer páginas accesibles y con apariencias cuidadas, gracias al empleo de hojas de estilo en cascada (CSS).

UT.15. Javascript

Esta unidad acercará a los alumnos a la realización de ciertas tareas en el navegador que está mostrando una página web, orientado básicamente a la validación de formularios y a la modificación del documento visualizado en tiempo real.

UT.16. Introducción a PHP

En esta unidad, los alumnos conocerán las nociones básicas de los lenguajes de script orientados al servidor, y las aplicarán al lenguaje PHP para crear páginas interactivas, comunicadas mediante formularios.

5.2. DETALLE DE CADA UNIDAD

UNIDAD 0. Conceptos básicos sobre programación

CONTENIDOS

Conceptos

- ✓ Evolución y clasificación de lenguajes: Lenguajes de bajo nivel y de alto nivel.
- ✓ Ensambladores, compiladores e intérpretes
- ✓ Pseudocódigo

Procedimientos

- ✓ Descripción de la evolución de los lenguajes informáticos
- ✓ Ventajas de los lenguajes compilados sobre los interpretados y viceversa.

Actividades de enseñanza-aprendizaje

- ✓ Identificación de los lenguajes interpretados de los lenguajes compilados
- ✓ Ejemplo de programa escrito en un lenguaje de bajo nivel
- ✓ Toma de contacto con un lenguaje de alto nivel interpretado (Basic) y otro compilado (Pascal y C).
- ✓ Ejemplo de un algoritmo sencillo en pseudocódigo.

Criterios de evaluación

- ✓ Distinguir las características de los lenguajes de alto y bajo nivel.
- ✓ Describir los tipos de lenguajes, compiladores y traductores de uso mas común.
- ✓ Ser capaz de expresar algoritmos elementales en pseudocódigo.

UNIDAD 1. Toma de contacto con C#

CONTENIDOS

Conceptos

- ✓ Escribir un texto en C#
- ✓ Mostrar números enteros en pantalla
- ✓ Operaciones aritméticas básicas
- ✓ Introducción a las variables: int
- ✓ Identificadores
- ✓ Comentarios
- ✓ Datos introducidos por el usuario

Procedimientos

- ✓ Manejo básico del compilador/entorno escogido para el módulo.
- ✓ Realización de programas sencillos capaces de mostrar textos en pantalla, o valores numéricos enteros, o el resultado de operaciones con números enteros.
- ✓ Previsión del resultado de operaciones matemáticas, teniendo en cuenta la prioridad de los operadores.
- ✓ Lectura de datos tecleados por el usuario del programa, mediante el uso de variables.

Actividades de enseñanza-aprendizaje

- ✓ Mostrar en pantalla textos prefijados.
- ✓ Mostrar en pantalla números enteros y realizar operaciones con ellos.
- ✓ Prever el resultado de operaciones matemáticas que implican diversos operadores.
- ✓ Leer números enteros tecleados por el usuario y realizar operaciones aritméticas con ellos.

Criterios de evaluación

- ✓ Mostrar en pantalla textos prefijados.
- ✓ Prever el resultado de operaciones matemáticas que implican diversos operadores.

- ✓ Distinguir qué nombres de identificadores son válidos y cuales no.
- ✓ Leer números enteros tecleados por el usuario y realizar operaciones aritméticas con ellos.
- ✓ Comentar programas para aclarar las partes de código más confusas.

UNIDAD 2. Tipos de datos básicos

CONTENIDOS

Conceptos

- ✓ Tipos de enteros: short/long, signed/unsigned.
- ✓ Sistemas de numeración: binario, octal, hexadecimal.
- ✓ Representación interna de los enteros
- ✓ Incremento y decremento
- ✓ Operaciones abreviadas: +=
- ✓ Tipo de dato real: Simple y doble precisión, cómo mostrar en pantalla.
- ✓ Tipo de dato carácter; cómo mostrar en pantalla
- ✓ Secuencias de escape
- ✓ Introducción a las cadenas de texto

Procedimientos

- ✓ Decidir el modificador adecuado para almacenar números de mayor o menor tamaño.
- ✓ Cambiar números enteros de un sistema de numeración a otra (decimal, binario, octal, hexadecimal).
- ✓ Incrementar/decrementar el valor de una variable.
- ✓ Expresar operaciones con notación abreviada.
- ✓ Realizar operaciones con números reales, conociendo la diferencia entre simple y doble precisión y sabiendo cual elegir según la situación.
- ✓ Conocer el espacio ocupado por una variable o por un tipo de datos.
- ✓ Manejar variables de tipo carácter, incluyendo secuencias de escape.

Actividades de enseñanza-aprendizaje

- ✓ Realizar operaciones con números enteros con o sin signo, de mayor o menor tamaño.
- ✓ Cambiar números enteros de un sistema de numeración a otra (decimal, binario, octal, hexadecimal).
- ✓ Incrementar/decrementar el valor de una variable.
- ✓ Expresar operaciones con notación abreviada.
- ✓ Realización de operaciones con números reales, tanto de simple como de doble precisión.
- ✓ Cálculo del espacio ocupado por una variable o por un tipo de datos.
- ✓ Manejo de variables de tipo carácter, incluyendo secuencias de escape.

Criterios de evaluación

- ✓ Operar con soltura con números enteros con o sin signo, de mayor o menor tamaño.
- ✓ Ser capaz de cambiar números enteros de un sistema de numeración a otra (decimal, binario, octal, hexadecimal).
- ✓ Saber cómo incrementar/decrementar el valor de una variable y cómo expresar operaciones con notación abreviada.
- ✓ Realizar operaciones con números reales, conociendo la diferencia entre simple y doble precisión y sabiendo cual elegir según la situación.
- ✓ Saber cómo conocer el espacio ocupado por una variable o por un tipo de datos.
- ✓ Manejar variables de tipo carácter, incluyendo secuencias de escape.

UNIDAD 3. Estructuras de control

CONTENIDOS

Conceptos

- ✓ Estructuras alternativas: If, If-else, Switch, Operador condicional
- ✓ Estructuras repetitivas: While, Do ... While, For. Break, continue
- ✓ Sentencia goto

- ✓ Nociones de diseño: diagramas de flujo

Procedimientos

- ✓ Comprobación de la alternativa correcta entre dos o un número reducido, con if y if-else.
- ✓ Comprobación de la alternativa entre varias posibles con switch.
- ✓ Realización de bloques repetitivos con while (si la condición de salida se comprueba al comienzo), con do..while (si la condición se comprueba al final) y con for (especialmente cuando es un número de iteraciones conocido).
- ✓ Interrupción de bucles con break y con continue.
- ✓ Uso de la sentencia goto y problemática asociada.
- ✓ Creación de diagramas de flujo para programas elementales.
- ✓ Introducción al depurador del entorno escogido.

Actividades de enseñanza-aprendizaje

- ✓ Realización de programas en los que se deba escoger entre dos opciones usando if.
- ✓ Realización de programas en los que se deba escoger entre un número reducido de opciones usando if-else.
- ✓ Escoger una alternativa entre varias posibles con switch.
- ✓ Realización de bloques repetitivos cuya condición de salida se comprueba al comienzo (con while)
- ✓ Realización de bloques repetitivos cuya condición se comprueba al final (con do..while).
- ✓ Realización de bloques repetitivos con un número de iteraciones conocido (con for).
- ✓ Realización de bloques repetitivos de todo tipo, en los que el alumno deba ser quien decida qué método usar.
- ✓ Ejemplo de uso de la sentencia goto y explicación de la problemática asociada (desorden en el fuente).
- ✓ Creación de diagramas de flujo para programas elementales.
- ✓ Seguimiento del valor de variables usando el depurador del entorno escogido.

Criterios de evaluación

- ✓ Saber escoger entre dos opciones usando if.
- ✓ Ser capaz de crear programas en los que se deba escoger entre un número reducido de opciones usando if-else.
- ✓ Escoger una alternativa entre varias posibles con switch, conociendo el uso correcto de las sentencias break y default.
- ✓ Corrección en la realización de bloques repetitivos cuya condición de salida se comprueba al comienzo (con while)
- ✓ Corrección en la realización de bloques repetitivos cuya condición se comprueba al final (con do..while).
- ✓ Corrección en la realización de bloques repetitivos con un número de iteraciones conocido (con for).
- ✓ Ser capaz de plantear bloques repetitivos de todo tipo, escogiendo un método adecuado para el problema.
- ✓ Conocer la sentencia goto y los problemas que se pueden derivar de su uso.
- ✓ Ser capaz de crear diagramas de flujo para programas elementales que incluyan condiciones o repetición de bloques de programa.
- ✓ Poder analizar el flujo del programa y comprobar el valor de variables usando el depurador del entorno escogido.

UNIDAD 4. Arrays y estructuras

CONTENIDOS

Conceptos

- ✓ Conceptos básicos sobre tablas
- ✓ Arrays unidimensionales
- ✓ Arrays bidimensionales
- ✓ Arrays multidimensionales
- ✓ Arrays de caracteres
- ✓ Arrays indeterminados
- ✓ Estructuras

- ✓ Estructuras anidadas
- ✓ Arrays de estructuras

Procedimientos

- ✓ Definición de arrays.
- ✓ Carga de datos en un array, en la inicialización o posteriormente.
- ✓ Acceso a los datos de un array.
- ✓ Empleo de las funciones específicas para manejo de arrays de caracteres.
- ✓ Estructuras: definición, carga de datos y acceso a los datos.

Actividades de enseñanza-aprendizaje

- ✓ Almacenar datos repetitivos empleando arrays.
- ✓ Almacenar datos compuestos empleando estructuras.
- ✓ Manejar cadenas de caracteres, empleando las principales funciones disponibles para ellas: lectura y escritura, asignación de valores y comparación con otras cadenas.
- ✓ Combinar el uso de estructuras y arrays para datos repetitivos formados por unidades más complejas.

Criterios de evaluación

- ✓ Saber cómo almacenar datos repetitivos empleando arrays, acceder a ellos y manipularlos.
- ✓ Almacenar datos compuestos empleando estructuras, acceder a ellos y manipularlos..
- ✓ Leer cadenas de caracteres desde el teclado, mostrarlas en pantalla, asignarles un valor y comparar con otras cadenas.
- ✓ Combinar el uso de estructuras y arrays para datos repetitivos formados por unidades más complejas.

UNIDAD 5. Ficheros

CONTENIDOS

Conceptos

- ✓ Conceptos teóricos sobre ficheros:

- ✓ Fichero, registro lógico, campo, subcampo, clave, registro físico (bloque)
- ✓ Clasificación de registros: de longitud fija, de longitud variable
- ✓ Operaciones con registros: Altas, bajas, modificaciones, consultas
- ✓ Clasificación de ficheros: Permanentes, temporales
- ✓ Operaciones con ficheros: creación, apertura, cierre, consulta, actualización; otras menos habituales.
- ✓ Organización de ficheros y acceso: organización secuencial, organización relativa: Directa (hash); indirecta o aleatoria (clave)
- ✓ Apertura y cierre de ficheros.
- ✓ Acceso secuencial (como carácter, cadenas, formateado o bloques de datos).
- ✓ Acceso directo.
- ✓ Ficheros especiales: la impresora, la salida de errores.

Procedimientos

- ✓ Apertura y cierre de ficheros.
- ✓ Acceso secuencial como carácter.
- ✓ Acceso secuencial como cadenas de texto.
- ✓ Acceso secuencial formateado.
- ✓ Acceso secuencial como bloques de datos.
- ✓ Acceso directo: conocer la posición actual y saltar a una cierta posición.
- ✓ Manejo de la impresora
- ✓ Envío de información a la salida de errores.
- ✓ Determinación de la estructura de fichero adecuada a un cierto problema.
- ✓ Realización de programas que manipulen las estructuras de fichero escogidas.

Actividades de enseñanza-aprendizaje

- ✓ Abrir y cerrar ficheros.
- ✓ Acceso secuencial de lectura y de escritura a un fichero como carácter.
- ✓ Acceso secuencial a un fichero como cadenas de texto.

- ✓ Acceso secuencial formateado.
- ✓ Acceso secuencial como bloques de datos.
- ✓ Saltar a una cierta posición del fichero y conocer la posición actual en que nos encontramos.
- ✓ Enviar información textual a la impresora
- ✓ Enviar información a la salida de errores.

Criterios de evaluación

- ✓ Ser capaz de acceder secuencialmente para lectura y escritura a un fichero de carácter en carácter.
- ✓ Poder acceder secuencialmente para lectura y escritura a un fichero formado por cadenas de texto.
- ✓ Acceder secuencialmente a un fichero formado por información formateada.
- ✓ Acceso secuencial a un fichero para lectura y escritura bloques de datos.
- ✓ Ser capaz de saltar a una cierta posición del fichero y conocer la posición actual en que se encuentra.
- ✓ Saber cómo enviar información textual a la impresora
- ✓ Poder enviar información a la salida de errores.
- ✓ Ser capaz de determinar la estructura de fichero adecuada a un cierto problema.
- ✓ Ser capaz de realizar un programa que manipule la estructura de fichero escogida.

UNIDAD 6. Funciones

CONTENIDOS

Conceptos

- ✓ Nociones de diseño modular de programas: Descomposición modular
- ✓ Conceptos básicos sobre funciones: Definición, Declaración, Llamada
- ✓ Retorno de una función
- ✓ Clases de funciones
- ✓ Clases de almacenamiento de las variables

- ✓ Parámetros de funciones
- ✓ Funciones y arrays
- ✓ Recursividad

Procedimientos

- ✓ Descomposición modular de problemas como método de obtención de funciones.
- ✓ Definición y uso de funciones.
- ✓ Especificación del valor de retorno de una función
- ✓ Uso de parámetros de funciones
- ✓ Creación de funciones recursivas

Actividades de enseñanza-aprendizaje

- ✓ Dado un problema, aplicar técnicas de descomposición modular para obtener las funciones más adecuadas para su resolución.
- ✓ Definición y uso de funciones, que devuelvan un valor o no (procedimientos).
- ✓ Definición de variables locales y globales.
- ✓ Uso de parámetros de funciones, por valor y por referencia.
- ✓ Creación de funciones recursivas.

Criterios de evaluación

- ✓ El alumno debe ser capaz de, dado un problema, aplicar técnicas de descomposición modular para obtener las funciones más adecuadas para su resolución.
- ✓ Definir funciones, llamarlas cuando sea necesario, usando valores de retorno o parámetros para intercambiar información.
- ✓ Definir variables locales para el trabajo interno de la función.
- ✓ Definir funciones recursivas, como forma rápida de resolver muchos problemas.

UNIDAD 7. Punteros y gestión dinámica de memoria

CONTENIDOS

Conceptos

- ✓ Necesidad de memoria dinámica
- ✓ Punteros. Operaciones con punteros.
- ✓ Estructuras dinámicas: Listas (contiguas, enlazadas o encadenadas, doblemente enlazadas, circulares), Pilas, Colas, Árboles. ArrayList. Colecciones.

Procedimientos

- ✓ Definición de punteros y asignación de direcciones.
- ✓ Asignación de punteros y aritmética de punteros.
- ✓ Uso de estructuras dinámicas.
- ✓ Definición y uso de los arrays de punteros.
- ✓ Definición y uso de los punteros a estructuras.
- ✓ Lectura de los parámetros pasados a un programa, usando los parámetros de "main".

Actividades de enseñanza-aprendizaje

- ✓ Definir punteros, reservarles espacio, asignarles valores, liberar el espacio reservado.
- ✓ Incrementar y decrementar punteros y valores apuntados.
- ✓ Comparar punteros y de valores apuntados.
- ✓ Acceder a los elementos de un array usando punteros.
- ✓ Crear arrays de punteros, almacenar datos y acceder a ellos.
- ✓ Definir punteros a estructuras, almacenar datos y acceder a ellos.
- ✓ Leer los parámetros pasados a un programa, usando los parámetros de "main".
- ✓ Manejar estructuras dinámicas.

Criterios de evaluación

- ✓ Ser capaz de definir punteros, reservar espacio, asignar valores y liberar finalmente el espacio ocupado.
- ✓ Saber comparar, incrementar y decrementar punteros y valores apuntados.
- ✓ Definir punteros a estructuras, almacenar datos y acceder a ellos.
- ✓ Elegir y emplear estructuras dinámicas adecuadas a un problema concreto.

- ✓ Poder crear programas no interactivos que respondan a los deseos del usuario, leyendo los parámetros de “main”.

UNIDAD 8. Librerías de uso frecuente

CONTENIDOS

Conceptos

- ✓ Acceso a consola
- ✓ Funciones matemáticas
- ✓ Números aleatorios
- ✓ Fecha y hora
- ✓ Llamada a órdenes del sistema

Procedimientos

- ✓ Aprovechamiento de la pantalla de texto (consola): escribir en cierta posición, indicar colores para el texto, leer pulsaciones de tecla con o sin espera, otras posibilidades.
- ✓ Uso de funciones matemáticas de uso frecuente: potencia, raíz cuadrada, trigonométricas, redondeo, valor absoluto.
- ✓ Generación números aleatorios.
- ✓ Acceso a la fecha y hora del sistema.
- ✓ Llamada a órdenes del sistema.

Actividades de enseñanza-aprendizaje

- ✓ Manipular la pantalla: borrar, escribir en cierta posición, indicar colores para el texto, leer pulsaciones de tecla con y sin espera.
- ✓ Calcular potencias con números reales, raíces cuadradas, expresiones trigonométricas y redondear valores, empleando las funciones de “math.h”.
- ✓ Generar números aleatorios.
- ✓ Leer la fecha y hora del sistema.
- ✓ Acceder a órdenes externas del sistema.

Criterios de evaluación

- ✓ Ser capaz de manipular la consola: borrar, escribir en cierta posición, indicar colores para el texto, leer pulsaciones de tecla con y sin espera.
- ✓ Ser capaz de calcular potencias con números reales, raíces cuadradas, expresiones trigonométricas y redondear valores.
- ✓ Saber cómo generar números aleatorios y emplearlos.
- ✓ Leer la fecha y hora del sistema.
- ✓ Acceder a órdenes externas del sistema.

UNIDAD 9. Introducción a la programación orientada a objetos

CONTENIDOS

Conceptos

- ✓ Conceptos de “objeto” y “clase”
- ✓ Definición de clases y de objetos desde C#
- ✓ Especificadores de acceso
- ✓ Métodos y atributos. Propiedades.
- ✓ Encapsulación. Herencia. Polimorfismo. Sobrecarga.
- ✓ Nociones básicas de UML. Generadores de código.

Procedimientos

- ✓ Descripción de un enunciado en términos de clases, objetos y relaciones de herencia.
- ✓ Elaboración de diagramas de clases para especificar la solución a un problema concreto.
- ✓ Desarrollo de programas en que se apliquen los conceptos de la programación orientada a objetos.

Actividades de enseñanza-aprendizaje

- ✓ Extraer las clases, objetos y relaciones de herencia existentes en un problema real.
- ✓ Elaborar diagramas de clases para especificar la solución a un problema real.

- ✓ Desarrollar programas aplicando los conceptos de la programación orientada a objetos.
- ✓ Rediseñar programas realizados anteriormente.

Criterios de evaluación

- ✓ Manejar con soltura conceptos como los de clase, herencia, encapsulación, polimorfismo y sobrecarga.
- ✓ Ser capaz de extraer las clases, objetos y relaciones de herencia en un enunciado basado en el mundo real.
- ✓ Elaborar diagramas de clases para especificar la solución a un problema real.
- ✓ Saber desarrollar programas aplicando los conceptos de la programación orientada a objetos.
- ✓ Poder rediseñar programas realizados en puntos anteriores del curso, aplicando en ellos las técnicas de la programación orientada a objetos.

UNIDAD 10. Otras características avanzadas de C#

CONTENIDOS

Conceptos

- ✓ Operadores para tratamiento de bits
- ✓ Operador coma
- ✓ Otros tipos de datos: Campos de bits, Uniones, Enumeraciones
- ✓ Otras características

Procedimientos

- ✓ Operaciones a nivel de bit, empleando los operadores adecuados.
- ✓ Uso del operador coma, tanto en for como en asignaciones.
- ✓ Manejo de otros tipos de datos que no se habían empleado hasta ahora: Campos de bits, Uniones, Enumeraciones.

Actividades de enseñanza-aprendizaje

- ✓ Realizar operaciones AND, OR, XOR, NOT a nivel de bit.

- ✓ Usar el operador coma para encadenar asignaciones.
- ✓ Crear y manejar datos enumerados.
- ✓ Crear y manejar uniones.
- ✓ Crear y manejar campos de bits.

Criterios de evaluación

- ✓ Saber realizar operaciones AND, OR, XOR, NOT a nivel de bit.
- ✓ Poder definir y manejar datos enumerados, uniones y campos de bits.

UNIDAD 11. Ejercicios completos

CONTENIDOS

Conceptos

- ✓ Aplicación de los conocimientos previos a la creación de aplicaciones de pequeño tamaño.
- ✓ Mantenimiento de aplicaciones ya creadas.

Procedimientos

- ✓ Creación de aplicaciones completas desde cero.
- ✓ Uso del depurador para encontrar fallos en aplicaciones.

Actividades de enseñanza-aprendizaje

- ✓ Se propondrá a los alumnos diversos retos, de complejidad creciente, que deberán resolver en forma de programa en lenguaje C#.
- ✓ Se propondrá también la corrección, mejora o modificación de programas ya existentes.

Criterios de evaluación

- ✓ Capacidad para desarrollar un programa completo como respuesta a un problema propuesto.
- ✓ Capacidad para encontrar fallos de problemas ya existentes, o ampliar sus funcionalidades.

UNIDAD 12. Proyecto final en lenguaje C#

CONTENIDOS

Conceptos

- ✓ No se aportarán conceptos nuevos, sólo técnicas que ayuden a los alumnos a trabajar en grupo de la forma más coordinada y más eficiente posible.

Procedimientos

- ✓ Trabajo en grupo en paralelo (cada alumno desarrollando una parte distinta, que luego se integran).
- ✓ Trabajo en grupo colaborativo (un alumno teclea mientras el otro se sienta a su lado, comprueba la corrección y da ideas; al cabo de cierto tiempo, se intercambian los papeles).

Actividades de enseñanza-aprendizaje

- ✓ Realización de un proyecto de mayor dificultad, gracias al trabajo colaborativo de varios alumnos. Los propios alumnos podrán proponer el ejercicio que desean realizar, si bien el profesor tendrá la última palabra en caso de que no se proponga nada, o se trate de problemas de dificultad demasiado alta o demasiado baja.

Criterios de evaluación

- ✓ Se valorará la dificultad del proyecto escogido, su resolución y el trabajo realizado por cada uno de los integrantes del grupo.

UNIDAD 13. El lenguaje de descripción de páginas web HTML

CONTENIDOS

Conceptos

- ✓ Editores HTML visuales.
- ✓ Modificación y creación desde editores de texto.
- ✓ Etiquetas básicas de HTML.
- ✓ Tablas.
- ✓ Formato básico desde HTML.

- ✓ XHTML

Procedimientos

- ✓ Creación de páginas web elementales mediante herramientas visuales.
- ✓ Creación de páginas web elementales mediante editores de texto.
- ✓ Consecución de apariencias más complejas mediante tablas y etiquetas de formato.
- ✓ Creación de páginas que “validen” según estándares.

Actividades de enseñanza-aprendizaje

- ✓ Se propondrá a los alumnos la creación y modificación de diversas páginas, de complejidad creciente, que deberán realizar tanto con herramientas visuales como con un editor de texto.

Criterios de evaluación

- ✓ Capacidad para desarrollar una página que cumpla los requisitos solicitados y que sea fiel a los estándares..

UNIDAD 14. Hojas de estilo CSS

CONTENIDOS

Conceptos

- ✓ Etiquetas básicas de CSS.
- ✓ Formas de integrar CSS en una página HTML.
- ✓ Etiquetas de CSS para formato de bloques.
- ✓ Varios estilos en una misma página HTML.
- ✓ Validación de CSS.

Procedimientos

- ✓ Creación de hojas de estilo simples.
- ✓ Modificación y adaptación de hojas de estilo complejas.
- ✓ Reformateo de páginas web sin necesidad de usar etiquetas HTML de colores ni tablas.
- ✓ Creación de distintos estilos para pantalla, impresora y otros dispositivos.

Actividades de enseñanza-aprendizaje

- ✓ Crear hojas de estilo simples desde cero.
- ✓ Modificar y adaptar de hojas de estilo complejas para que la apariencia pase a ser la que el profesor indique.
- ✓ Crear páginas web con columnas y distintos formatos de fuente, sin emplear etiquetas HTML de colores ni tablas.
- ✓ Crear de distintos estilos alternativos para una misma página web, que sean aplicables a pantalla, impresora y otros dispositivos.

Criterios de evaluación

- ✓ Se valorará la completitud en los ejercicios propuestos por el profesor.
- ✓ Además, el alumno deberá crear varias páginas web apoyadas en estilos CSS, en las que se valorará la apariencia final y la dificultad.

UNIDAD 15. Javascript

CONTENIDOS

Conceptos

- ✓ Nociones básicas de Javascript.
- ✓ Redirección. Diálogos.
- ✓ Validación de formularios.
- ✓ Recorrido y modificación del DOM.

Procedimientos

- ✓ Trabajo en grupo en paralelo (cada alumno desarrollando una parte distinta, que luego se integran).
- ✓ Ejercicios en orden de dificultad ascendente incorporando las novedades en lo referente a programación que este lenguaje aporta.

Actividades de enseñanza-aprendizaje

- ✓ Realización de un proyecto de mayor dificultad, gracias al trabajo colaborativo de varios alumnos. Los propios alumnos podrán proponer el ejercicio que desean realizar, si bien el profesor tendrá la última palabra en caso de que no se proponga nada, o se trate de problemas de dificultad demasiado alta o demasiado baja.

Criterios de evaluación

- ✓ Se valorará la dificultad del proyecto escogido, su resolución y el trabajo realizado por cada uno de los integrantes del grupo.

UNIDAD 16. Introducción a PHP

CONTENIDOS

Conceptos

- ✓ Nociones de lenguajes de script en servidor. Requisitos.
- ✓ Sintaxis básica de PHP.
- ✓ Variables y estructuras de control en PHP.
- ✓ Comunicación entre páginas mediante formularios y mediante parámetros en la URL.
- ✓ Introducción al acceso a bases de datos desde PHP.

Procedimientos

- ✓ Creación de programas básicos en PHP.
- ✓ Creación de programas interactivos en PHP mediante formularios.
- ✓ Creación de programas interactivos en PHP mediante paso de parámetros en la URL.
- ✓ Almacenamiento de datos desde PHP. Acceso a datos almacenados.

Actividades de enseñanza-aprendizaje

- ✓ Realización de ejercicios básicos en los que se apliquen las estructuras básicas de PHP.
- ✓ Realización de pequeñas aplicaciones interactivas

Criterios de evaluación

- ✓ Se valorará la correcta realización de los ejercicios de clase, así como dificultad del proyecto escogido, su resolución y el trabajo realizado por cada uno de los integrantes del grupo.

6. TEMPORALIZACIÓN

TEMA	HORAS
0. Conceptos básicos sobre programación	3
1. Toma de contacto con C#	10
2. Tipos de datos básicos	15
3. Estructuras de control	24
4. Arrays y estructuras	10
5. Ficheros	18
6. Funciones	24
7. Punteros y gestión dinámica de memoria	22
8. Librerías de uso frecuente	30
9. Programación Orientada a Objetos	35
10. Otras características avanzadas de C#	30
11. Ejercicios completos	35
12. Proyecto final en lenguaje C#	30
13. HTML	18
14. Hojas de estilo CSS	18
15. Javascript	20
16. Introducción a PHP	30
	384

En el trabajo diario en clase, se alternarán todas las semanas los fundamentos y la programación en C# (9 horas/semana) y la programación web (3 horas/semana).

7. BIBLIOGRAFÍA

Se prepararán apuntes para los alumnos, que se les entregarán al final de cada unidad temática. Adicionalmente, se recomendará las siguientes lecturas adicionales a los alumnos que deseen profundizar:

- Criado, M.A. *Programación en Lenguajes Estructurados*, Ra-Ma, 2005
- Molina, M. *Programación en Lenguajes Estructurados*, McGraw-Hill, 2006
- Quero E., *Fundamentos de programación*, Paraninfo 2001
- Alcalde, E., García, M., *Metodología de la programación*, McGraw-Hill, 1992.
- Joyanes, L., *Fundamentos de Programación*, McGraw-Hill
- Ceballos, F.J. *Curso de programación con C*. Ra-Ma, Madrid 1993
- Kernighan, B., Ritchie, D., *El lenguaje de programación C*, Prentice-Hall
- Musciano, C., Kennedy B., *HTML & XHTML*, O'Reilly, 2002.
- Meyer, E., *Eric Meyer on CSS*, New Riders Publishing, 2004
- Negrino, T., Smith, D., *JavaScript*, Prentice Hall, 2005
- Ullman, L., *Guía de aprendizaje de PHP*, Prentice Hall, 2001

Así como otros documentos disponibles en Internet, entre los que se puede citar:

- Tutor de C# de José Antonio González Seco, en www.josanguapo.com
- "Thinking in C#", de Bruce Eckel.
- Curso de C# en CSharpStation
- Curso de C por Angel Salas (Universidad de Zaragoza).
- "Aprenda ANSI C como si estuviera en primero", de la Escuela Superior de Ingenieros Industriales, Universidad de Navarra.

- Curso de C, por Nacho Cabanes.
- Curso de C (en línea) de "El rincón del C".
- Referencia del lenguaje PHP en www.php.net

8. PLANTEAMIENTO DE LA ATENCIÓN A LA DIVERSIDAD

Es evidente que el ritmo de desarrollo de las capacidades no tiene por qué ser el mismo en todo un colectivo como es el grupo de alumnas y alumnos. En un proceso de aprendizaje en el que lo principal o exclusivo es la adquisición de conocimientos, las adaptaciones curriculares a los diferentes ritmos de aprendizaje deben realizarse actuando sobre el método (entendido aquí como un elemento curricular más), proponiendo actividades diversas que conduzcan a metas semejantes.

Por ello, existirá una serie de conceptos teóricos y de ejercicios prácticos que todos los alumnos deberán realizar.

Para aquellos alumnos y alumnas con nivel elevado de conocimientos o con un ritmo de enseñanza-aprendizaje más rápido, se plantearán una serie de actividades de ampliación.

Finalmente, también se plantearán actividades que puedan servir de refuerzo para aquellos alumnos y alumnas con un menor ritmo de aprendizaje.

9. CRITERIOS DE EVALUACIÓN

Los criterios específicos de evaluación de los contenidos de cada unidad temática se han indicado con anterioridad. Aquí se hace referencia a la evaluación global del curso.

La **calificación** de los alumnos dependerá:

- Actividades de enseñanza aprendizaje..... 30%
- Actividades específicas de evaluación..... 50%
- Proyecto globalizador..... 10%
- Actitud y participación en clase..... 10%

Además para superar un módulo es necesario:

- Las ausencias a clase no superarán el 20% del horario lectivo según lo regulado en el Reglamento de Régimen Interior de Centro.
- No tener actitudes contrarias a las normas de convivencia.
- Haber realizado satisfactoriamente las actividades programadas, como indispensables, por el profesor/a.
- En el proyecto globalizador deberá haber obtenido al menos la calificación de 5 sobre 10.

Con relación a las faltas de asistencia no justificadas, cada una penalizará rebajando en 0,20 puntos la nota de la correspondiente evaluación, como forma efectiva de que un 20% de las faltas (20 horas de las casi 100 que integran cada evaluación) supongan un suspenso casi automático en la evaluación en cuestión.

Durante la evaluación se realizarán frecuentes ejercicios con nota, como forma de comprobar la evolución del alumnado. Al final de cada evaluación se realizarán actividades teóricas y prácticas de evaluación que será necesario superar (al menos obtener 5 sobre 10) para aprobar esa evaluación. Se realizará una recuperación de estas pruebas de la 1ª y 2ª evaluación para los alumnos que, habiéndose presentado a ellas, no las hayan superado. Deberán presentarse a una prueba final en junio los alumnos que tengan pendientes alguna de las evaluaciones.

Por otra parte, dado que durante el trabajo en el aula, la asignatura se considerará dividida en dos grandes bloques (programación estructurada y orientada a objetos, y programación web), a efectos de evaluación será necesario obtener una nota de 4 o superior en ambos bloques a fin de poder calcular la nota media y optar al aprobado en la asignatura. Si la nota obtenida en alguno de estos dos bloques es inferior a 4, no se podrá calcular la nota media, y la asignatura se considerará suspensa.

10. ACTIVIDADES COMPLEMENTARIAS Y EXTRAESCOLARES.

Sería conveniente, si existiera la posibilidad, visitar las instalaciones de alguna empresa de desarrollo de software. De igual modo, sería deseable una visita a la feria de la informática conocida como SIMO, para ayudar a que los alumnos se mantengan en contacto con las novedades del sector.

11. MATERIALES Y RECURSOS DIDÁCTICOS.

Los materiales y recursos que se emplearán en la asignatura son:

HARDWARE:

- Un servidor Pentium IV.
- Treinta estaciones de trabajo Pentium IV conectadas en red.
- Una impresora láser y un escáner A4.
- Un sistema de proyección (proyector SVGA y pantalla).
- Conexión a Internet ADSL.

SOFTWARE:

- Sistemas operativos en red: Windows XP Professional, Ubuntu Linux 7.10.