

DESARROLLO CURRICULAR DEL MÓDULO

PROGRAMACIÓN

CICLO FORMATIVO DE GRADO SUPERIOR

DESARROLLO DE APLICACIONES MULTIPLATAFORMA

(Modalidades presencial y a distancia)

I. E. S. SAN VICENTE 2011/12

Profesores:

José Ignacio Cabanes (Presencial)

Paula Sempere (Distancia)

1. INTRODUCCIÓN	4
2. COMPETENCIAS PERSEGUIDAS	4
3. OBJETIVOS	4
4. ORGANIZACIÓN DE LOS CONTENIDOS	8
4.1.- CONTENIDOS BÁSICOS.....	8
4.2.- ESTRUCTURA DE LOS CONTENIDOS.....	11
4.3.- RELACIÓN SECUENCIADA DE UNIDADES DE TRABAJO.....	13
4.4.- CONTENIDOS MÍNIMOS.....	16
5. ELEMENTOS CURRICULARES DE CADA UNIDAD	16
5.1. PLANTEAMIENTO DE LAS UNIDADES TEMÁTICAS.....	16
5.2. DETALLE DE CADA UNIDAD.....	18
UNIDAD 0. Conceptos básicos sobre programación	18
UNIDAD 1. Toma de contacto con C#	19
UNIDAD 2. Estructuras de control.....	20
UNIDAD 3. Tipos de datos básicos.....	22
UNIDAD 4. Arrays y estructuras.....	24
UNIDAD 5. Funciones.....	25
UNIDAD 6. Introducción a la programación orientada a objetos	26
UNIDAD 7. Utilización avanzada de clases.....	27
UNIDAD 8. Ficheros	28
UNIDAD 9. Acceso a bases de datos relacionales	30
UNIDAD 10. Persistencia de objetos.....	31
UNIDAD 11. Bibliotecas de uso frecuente.....	32
UNIDAD 12. Gestión dinámica de la memoria.....	34
UNIDAD 13. Otras características avanzadas de C#.....	36
UNIDAD 14. Depuración, prueba y documentación de programas.....	37
UNIDAD 15. Proyecto final en lenguaje C#.....	38
6. TEMPORALIZACIÓN	40
7. BIBLIOGRAFÍA	41
8. PLANTEAMIENTO DE LA ATENCIÓN A LA DIVERSIDAD	42
9. CRITERIOS DE EVALUACIÓN	43
10. RECUPERACIÓN DE PENDIENTES	45
11. ACTIVIDADES COMPLEMENTARIAS Y EXTRAESCOLARES.	46

12. ENSEÑANZA BILINGÜE_____ **46**

13. MATERIALES Y RECURSOS DIDÁCTICOS._____ **47**

1. INTRODUCCIÓN

El módulo de **Programación** tiene una duración de 256 horas totales (8 horas semanales) y se encuadra en el primer curso del Ciclo Formativo de Grado Superior correspondiente al título de Desarrollo de Aplicaciones Multiplataforma.

2. COMPETENCIAS PERSEGUIDAS

La referencia del sistema productivo de este Módulo la encontramos en el Real Decreto 450/2010, de 16 de abril, que especifica, entre otras, las siguientes competencias profesionales, personales y sociales:

- Gestionar entornos de desarrollo adaptando su configuración en cada caso para permitir el desarrollo y despliegue de aplicaciones.
- Desarrollar aplicaciones multiplataforma con acceso a bases de datos utilizando lenguajes, librerías y herramientas adecuados a las especificaciones.
- Desarrollar aplicaciones implementando un sistema completo de formularios e informes que permitan gestionar de forma integral la información almacenada.
- Realizar planes de pruebas verificando el funcionamiento de los componentes software desarrollados, según las especificaciones.
- Desplegar y distribuir aplicaciones en distintos ámbitos de implantación verificando su comportamiento y realizando las modificaciones necesarias.
- Gestionar su carrera profesional, analizando las oportunidades de empleo, autoempleo y de aprendizaje.
- Mantener el espíritu de innovación y actualización en el ámbito de su trabajo para adaptarse a los cambios tecnológicos y organizativos de su entorno profesional.
- Participar de forma activa en la vida económica, social y cultural, con una actitud crítica y responsable.

3. OBJETIVOS

En el citado Real Decreto se indican también los objetivos generales del ciclo formativo, de los que se puede extraer, considerando el contexto en el que se impartirá el módulo, los siguientes:

- Seleccionar y emplear lenguajes, herramientas y librerías, interpretando las especificaciones para desarrollar aplicaciones multiplataforma con acceso a bases

de datos.

- Gestionar la información almacenada, planificando e implementando sistemas de formularios e informes para desarrollar aplicaciones de gestión.
- Seleccionar y utilizar herramientas específicas, lenguajes y librerías, evaluando sus posibilidades y siguiendo un manual de estilo, para manipular e integrar en aplicaciones multiplataforma contenidos gráficos y componentes multimedia.
- Emplear herramientas de desarrollo, lenguajes y componentes visuales, siguiendo las especificaciones y verificando interactividad y usabilidad, para desarrollar interfaces gráficos de usuario en aplicaciones multiplataforma.
- Seleccionar y emplear técnicas, motores y entornos de desarrollo, evaluando sus posibilidades, para participar en el desarrollo de juegos y aplicaciones en el ámbito del entretenimiento.
- Instalar y configurar módulos y complementos, evaluando su funcionalidad, para gestionar entornos de desarrollo.
- Verificar los componentes software desarrollados, analizando las especificaciones, para completar un plan de pruebas.
- Establecer procedimientos, verificando su funcionalidad, para desplegar y distribuir aplicaciones.

Que se podría plasmar en los siguientes objetivos más concretos para este módulo:

- 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.
- 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.
- 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.
- 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.
- OM18 Acceder a bases de datos relacionales desde programas realizados usando el lenguaje C#.
- OM19 Usar la persistencia para almacenar información sobre los objetos empleados por una 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.

Fomento de la lectura

- Fomentar la curiosidad y la inquietud por adquirir nuevos conocimientos, y valorar los libros (tanto en papel como electrónicos) como fuente de información e incluso de entretenimiento.

4. ORGANIZACIÓN DE LOS CONTENIDOS

4.1.- CONTENIDOS BÁSICOS

Los contenidos básicos que propone el Real Decreto son los siguientes.

Identificación de los elementos de un programa informático:

- Estructura y bloques fundamentales.
- Variables.
- Tipos de datos.
- Literales.
- Constantes.

- Operadores y expresiones.
- Conversiones de tipo.
- Comentarios.

Utilización de objetos:

- Características de los objetos.
- Instanciación de objetos.
- Utilización de métodos.
- Utilización de propiedades.
- Utilización de métodos estáticos.
- Constructores.
- Destrucción de objetos y liberación de memoria.

Uso de estructuras de control:

- Estructuras de selección.
- Estructuras de repetición.
- Estructuras de salto.
- Control de excepciones.

Desarrollo de clases:

- Concepto de clase.
- Estructura y miembros de una clase.
- Creación de atributos.
- Creación de métodos.
- Creación de constructores.
- Utilización de clases y objetos.
- Utilización de clases heredadas.

Lectura y escritura de información:

- Tipos de flujos. Flujos de bytes y de caracteres.
- Clases relativas a flujos.
- Utilización de flujos.
- Entrada desde teclado.
- Salida a pantalla.
- Ficheros de datos. Registros.
- Apertura y cierre de ficheros. Modos de acceso.
- Escritura y lectura de información en ficheros.
- Utilización de los sistemas de ficheros.
- Creación y eliminación de ficheros y directorios.
- Interfaces.
- Concepto de evento.
- Creación de controladores de eventos.

Aplicación de las estructuras de almacenamiento:

- Estructuras.
- Creación de arrays.
- Arrays multidimensionales.
- Cadenas de caracteres.
- Listas.

Utilización avanzada de clases:

- Composición de clases.
- Herencia.
- Superclases y subclases.

-
- Clases y métodos abstractos y finales.
 - Sobreescritura de métodos.
 - Constructores y herencia.

Mantenimiento de la persistencia de los objetos:

- Bases de datos orientadas a objetos.
- Características de las bases de datos orientadas a objetos.
- Instalación del gestor de bases de datos.
- Creación de bases de datos.
- Mecanismos de consulta.
- El lenguaje de consultas: sintaxis, expresiones, operadores.
- Recuperación, modificación y borrado de información.
- Tipos de datos objeto; atributos y métodos.
- Tipos de datos colección.

Gestión de bases de datos relacionales:

- Establecimiento de conexiones.
- Recuperación de información.
- Manipulación de la información.
- Ejecución de consultas sobre la base de datos.

4.2.- ESTRUCTURA DE LOS CONTENIDOS

Estos contenidos básicos propuestos por el Real Decreto se redistribuirán, de modo que su aprendizaje en clase sea progresivo y permita la realización de prácticas desde el primer día. Así, 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. Programación orientada a objetos
 3. Acceso a ficheros, persistencia y a acceso a bases de datos
 4. Otras características de los lenguajes de programación actuales
 5. Proyecto final

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 comenzará relacionando la sintaxis de las estructuras de datos simples, y las sentencias elementales de este lenguaje, comparando con el 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: Programación orientada a objetos

Una vez que el alumno conoce la programación estructurada, se le introducirá en la programación orientada a objetos, que le permitirá usar técnicas de diseño más naturales y descomponer problemas de mayor tamaño.

Bloque 3: Almacenamiento de la información

Como distintas alternativas para que la información quede almacenada de forma permanente, se tratará el acceso a ficheros, la persistencia de objetos y el acceso a bases de datos relacionales.

Bloque 4: Otras características de los lenguajes de programación actuales

Es interesante el alumno adquiera las nociones básicas de otras características de los lenguajes actuales, como el acceso a la fecha y hora del sistema, la conexión mediante red o a un servidor web, la generación de números aleatorios, el acceso avanzado a la consola o la creación visual de interfaces gráficas. Algunos de estos temas se verán con más detalle en asignaturas de segundo curso, por lo que este bloque apenas tendrá carácter introductorio.

Bloque 5: Proyecto final

Como forma de que los alumnos apliquen todos los conocimientos adquiridos a un

proyecto de programación real, la última parte se dedicará a realizar un proyecto individual, de forma casi totalmente independiente.

4.3.- 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.

Junto al nombre de cada tema se indican los contenidos básicos (propuestos por el Real Decreto) que cubre. En algunos casos, se trata de temas que no se mencionan directamente en el Real Decreto, pero que se han incluido por completitud y/o por facilitar la empleabilidad de los alumnos. Estos contenidos están precedidos por (*).

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

Tema	Contenidos básicos
0. Conceptos básicos sobre programación	
1. Toma de contacto con C#	Identificación de los elementos de un programa informático: Estructura y bloques fundamentales, Variables, Tipos de datos, Literales, Constantes, Operadores y expresiones, Conversiones de tipo, Comentarios. Lectura y escritura de información (1): Entrada desde teclado, Salida a pantalla.
2. Estructuras de control	Uso de estructuras de control: Estructuras de selección, Estructuras de repetición, Estructuras de salto, Control de excepciones.
3. Tipos de datos básicos	Aplicación de las estructuras de almacenamiento: Cadenas de caracteres.
4. Arrays y estructuras	Aplicación de las estructuras de almacenamiento: Estructuras, Creación de arrays, Arrays multidimensionales, Listas.
5. Funciones	(*). Diseño modular de programas; Parámetros; Valor devuelto; Variables locales; Modificación de parámetros

Bloque 2: Programación orientada a objetos

Tema	Contenidos básicos
6. Introducción a la programación orientada a objetos	<p>Utilización de objetos: Características de los objetos, Instanciación de objetos, Utilización de métodos, Utilización de propiedades, Utilización de métodos estáticos, Constructores, Destrucción de objetos y liberación de memoria.</p> <p>Desarrollo de clases: Concepto de clase, Estructura y miembros de una clase, Creación de atributos, Creación de métodos, Creación de constructores, Utilización de clases y objetos, Utilización de clases heredadas.</p>
7. Utilización avanzada de clases	<p>Utilización avanzada de clases: Composición de clases, Herencia, Superclases y subclases, Clases y métodos abstractos y finales, Sobreescritura de métodos, Constructores y herencia.</p>

Bloque 3: Almacenamiento de la información

Tema	Contenidos básicos
8. Manejo de ficheros	<p>Lectura y escritura de información (2): Tipos de flujos. Flujos de bytes y de caracteres; Clases relativas a flujos; Utilización de flujos; Ficheros de datos. Registros; Apertura y cierre de ficheros. Modos de acceso; Escritura y lectura de información en ficheros; Utilización de los sistemas de ficheros; Creación y eliminación de ficheros y directorios.</p>
9. Acceso a bases de datos relacionales	<p>Gestión de bases de datos relacionales: Establecimiento de conexiones, Recuperación de información, Manipulación de la</p>

	información, Ejecución de consultas sobre la base de datos.
10. Persistencia de objetos	Mantenimiento de la persistencia de los objetos: Bases de datos orientadas a objetos, Características de las bases de datos orientadas a objetos, Instalación del gestor de bases de datos, Creación de bases de datos, Mecanismos de consulta, El lenguaje de consultas: sintaxis, expresiones, operadores, Recuperación, modificación y borrado de información, Tipos de datos objeto; atributos y métodos, Tipos de datos colección.

Bloque 4: Otras características de los lenguajes de programación actuales

Tema	Contenidos
11. Bibliotecas de uso frecuente	(*) Funciones matemáticas; Generación de números aleatorios; Acceso avanzado a la pantalla de texto (consola); Fecha y hora; Temporización; Lectura de directorios; El entorno; Llamadas al sistema; Juegos con Tao.SDL; Algunos servicios de red.
12. Gestión dinámica de memoria	(*) Punteros. Operaciones con punteros. Estructuras dinámicas: Listas, Pilas, Colas, Árboles. ArrayList.
13. Otras características avanzadas de C#	(*) Espacios de nombres; Operaciones con bits; Enumeraciones; Propiedades; Parámetros de salida (out); Introducción a las expresiones regulares; El operador coma; Interfaces; Concepto de evento; Creación de controladores de eventos; Diseño visual de interfaces.
14. Depuración, prueba y documentación de programas	(*) Conceptos básicos sobre depuración. Prueba de programas. Documentación básica de programas.

Bloque 5: Proyecto final

Tema	Contenidos
15. Proyecto final	(*) -

4.4.- 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 los objetivos 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 y de muy corta duración.

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 textos y números desde el teclado y realizar operaciones aritméticas básicas.

UT.2. 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, así como nociones básicas de control de excepciones.

UT.3. 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.

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 más en el uso de las cadenas de texto.

UT.5. Funciones

El uso de funciones es una característica muy importante en cualquier convencional, pues permite realizar un desarrollo modular del programa al tiempo que facilita su mantenimiento y futuro uso en otras aplicaciones. Por eso, se dedica un tema a la creación y uso de funciones, todavía sin hablar de programación orientada a objetos, de modo que se trate de conocimientos aplicables a otros lenguajes de programación.

UT.6. 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.7. Utilización avanzada de clases

En esta unidad se profundizará en detalles más avanzados, como los constructores y destructores, métodos y atributos static, el polimorfismo y la sobrecarga, la creación de arrays de objetos, y las palabras reservadas override y this.

UT.8. 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 repasar el concepto de “excepciones”.

UT.9. Acceso a bases de datos relacionales

En grandes y medianos proyectos, es mucho más habitual almacenar la información en bases de datos que en ficheros convencionales. Por eso, se introducirá al alumno en la conexión a bases de datos y la forma de introducir, recuperar y modificar información.

UT.10. Persistencia de objetos

Como alternativa a las bases de datos relacionales y a los ficheros convencionales, se introducirá al alumno en las nociones básicas de persistencia de objetos, tanto a nivel de

estructuras del lenguaje como utilizando bases de datos orientadas a objetos.

UT.11. Bibliotecas 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, para temporización de procesos, o para el acceso avanzado a la pantalla de texto (consola). Esas bibliotecas serán utilizadas en esta Unidad.

UT.12. Gestión dinámica de la 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.13. Otras características avanzadas de C#

Esta unidad temática presentará otras características de uso menos frecuente, como los operadores a nivel de bits, el operador coma y otras posibilidades menos habituales.

UT.14. Depuración, prueba y documentación de programas.

El alumno aprenderá las pautas básicas para depurar un programa y así localizar errores, así como a crear casos de prueba automatizados. También se le darán ciertas nociones de documentación de programas, principalmente relativas inclusión de comentarios y generación automática de la documentación a partir de éstos.

UT.15. Proyecto final en lenguaje C#

En esta unidad, los alumnos tendrán que enfrentarse a un proyecto de programación individual de mayor complejidad.

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 y 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. 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
- ✓ Introducción a las excepciones

Procedimientos

- ✓ Comprobación de la alternativa correcta entre dos o u 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.
 - ✓ Interceptación de errores en tiempo de ejecución mediante el mecanismo de excepciones.

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.
- ✓ Interceptar errores básicos en tiempo de ejecución (conversiones de tipo) usando control de excepciones.

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.
 - ✓ Interceptar de forma correcta errores básicos de conversión de tipo usando control de excepciones.

UNIDAD 3. 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.
 - ✓ Manejar textos simples: leer cadenas de caracteres desde el teclado, mostrarlas en pantalla, asignarles un valor y comparar con otras cadenas.

UNIDAD 4. Arrays y estructuras

CONTENIDOS

Conceptos

- ✓ Conceptos básicos sobre tablas
- ✓ Arrays unidimensionales
- ✓ Arrays bidimensionales
- ✓ Arrays multidimensionales
- ✓ 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.
- ✓ Estructuras: definición, carga de datos y acceso a los datos.

Actividades de enseñanza-aprendizaje

- ✓ Almacenar datos repetitivos empleando arrays.
- ✓ Almacenar datos compuestos empleando estructuras.
- ✓ 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..
- ✓ Combinar el uso de estructuras y arrays para datos repetitivos formados por unidades más complejas.

UNIDAD 5. 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 6. 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.
- ✓ 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 7. Utilización avanzada de clases

CONTENIDOS

Conceptos

- ✓ Composición de clases
- ✓ Herencia, superclases y subclases
- ✓ Polimorfismo. Sobrecarga.
- ✓ Sobreescritura de métodos.
- ✓ Métodos abstractos y finales.
- ✓ Constructores y herencia.
- ✓ Arrays de objetos.
- ✓ La palabra “this”

-
- ✓ Sobrecarga de operadores

Procedimientos

- ✓ Resolución de problemas más complejos que incluyan herencia a varios niveles.
- ✓ Reescritura de métodos en subclases.
- ✓ Herencia en constructores.
- ✓ Diferencias entre un array de struct y uno de objetos. Forma de uso de los arrays de objetos.
- ✓ Mayor legibilidad del código fuente, gracias a la sobrecarga de operadores

Actividades de enseñanza-aprendizaje

- ✓ A partir de diagramas de clases, plasmar fuentes que incluyan herencia a varios niveles.
- ✓ Crear subclases que redefinan algunos de los métodos existentes en las superclases.
- ✓ Crear constructores que se apoyen en otros constructores o en otros métodos de la superclase.
- ✓ Crear arrays de objetos.
- ✓ Crear clases que contengan operadores sobrecargados.

Criterios de evaluación

- ✓ Ser capaz de crear crear programas con más de un nivel de herencia.
- ✓ Poder redefinir correctamente métodos existentes en las superclases.
- ✓ Saber crear constructores que se apoyen en otros constructores de la misma clase.
- ✓ Poder crear constructores que se apoyen en otros constructores de la superclase.
- ✓ Saber crear, inicializar y consultar arrays de objetos.
- ✓ Ser capaz de clases que contengan operadores sobrecargados.

UNIDAD 8. 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.
 - ✓ Formas de acceso a un fichero binario usando C#

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.
- ✓ 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.

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.
- ✓ 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 9. Acceso a bases de datos relacionales

CONTENIDOS

Conceptos

- ✓ Nociones básicas sobre bases de datos relacionales
- ✓ Conexiones a bases de datos
- ✓ Formas de introducción de información
- ✓ Métodos de recuperación de información
- ✓ Manipulación de la información: cambios, borrados

Procedimientos

- ✓ Establecimiento de conexiones a una base de datos
- ✓ Introducción de información
- ✓ Recuperación de la información introducida
- ✓ Recuperación de información calculada

-
- ✓ Alteración de la información
 - ✓ Borrado de la información existente

Actividades de enseñanza-aprendizaje

- ✓ Establecimiento de conexiones a una base de datos previamente creada
- ✓ Introducción de información desde un programa
- ✓ Recuperación de la información introducida anteriormente
- ✓ Recuperación de información calculada a partir de datos existentes
- ✓ Alteración de la información que contiene la base de datos
- ✓ Borrado de la información existente
- ✓ Creación de bases de datos desde programa

Criterios de evaluación

- ✓ Ser capaz de establecer conexiones a una base de datos
- ✓ Poder introducción información a la base de datos
- ✓ Recuperar información introducida
- ✓ Saber obtener información calculada a partir de los datos almacenados
- ✓ Ser capaz de modificar la información existente
- ✓ Poder borrar la información existente

UNIDAD 10. Persistencia de objetos

CONTENIDOS

Conceptos

- ✓ Características de las bases de datos orientadas a objetos
- ✓ Creación de bases de datos
- ✓ El lenguaje de consultas: sintaxis, expresiones, operadores
- ✓ Recuperación, modificación y borrado de información
- ✓ Tipos de datos objeto; atributos y métodos

-
- ✓ Tipos de datos colección.

Procedimientos

- ✓ Instalación del gestor de bases de datos.
- ✓ Creación de bases de datos.
- ✓ Recuperación, modificación y borrado de información.

Actividades de enseñanza-aprendizaje

- ✓ Instalar un gestor de bases de datos orientado a objetos.
- ✓ Crear bases de datos.
- ✓ Introducir información en la base de datos.
- ✓ Recuperación de información de la base de datos.
- ✓ Borrado de información de la base de datos.

Criterios de evaluación

- ✓ Ser capaz de instalar un gestor de bases de datos orientado a objetos.
- ✓ Crear bases de datos.
- ✓ Saber introducir información en la base de datos.
- ✓ Poder recuperación información de la base de datos.
- ✓ Ser capaz de borrar información de la base de datos.

UNIDAD 11. Bibliotecas de uso frecuente

CONTENIDOS

Conceptos

- ✓ La consola (pantalla en modo texto)
- ✓ Nociones básicas de entornos gráficos
- ✓ Fecha y hora. Temporización
- ✓ El entorno. Llamadas al sistema
- ✓ Servicios de red

-
- ✓ Nociones básicas sobre juegos

Procedimientos

- ✓ Acceso mejorado a la consola: posicionamiento, colores, lectura directa de teclas
- ✓ Creación visual de entornos gráficos. Componentes elementales: botones, etiquetas, casillas de texto, listas.
- ✓ Lectura de la fecha y hora del sistema.
- ✓ Temporización y pausas
- ✓ Acceso a datos sobre el sistema operativo y el entorno de ejecución.
- ✓ Llamadas a otras órdenes del sistema
- ✓ Conexión con otro ordenador mediante red
- ✓ Acceso a un servidor web desde un programa
- ✓ Nociones básicas sobre juegos: el bucle de juego, representación de elementos gráficos, detección básica de colisiones, descomposición en clases de objetos que cooperan

Actividades de enseñanza-aprendizaje

- ✓ Escribir texto en colores y en distintas posiciones
- ✓ Detener el programa hasta que se pulsen ciertas teclas
- ✓ Lectura del teclado sin detener el programa
- ✓ Contacto con diseñadores visuales de entornos gráficos
- ✓ Introducción de botones, etiquetas, casillas de texto y listas a un interfaz de usuario
- ✓ Asignación de eventos a los componentes visuales de un interfaz de usuario
- ✓ Lectura de la fecha y hora del sistema
- ✓ Realización de pausas temporizadas en un programa
- ✓ Acceder a datos sobre el sistema operativo y el entorno de ejecución.
- ✓ Llamar a otras órdenes del sistema desde el programa actual
- ✓ Establecer un diálogo con otro ordenador mediante una conexión de red
- ✓ Acceder a la información de una página web desde un programa
- ✓ Crear un esqueleto de juego en el que un elemento gráfico se mueve respondiendo al

teclado.

- ✓ Ampliar el esqueleto de juego para que un segundo elemento gráfico se mueva de forma independiente.
- ✓ Ampliar el esqueleto de juego para añadir detección simple de colisiones.
- ✓ Ampliar el esqueleto de juego, descomponiéndolo en varios objetos que cooperan

Criterios de evaluación

- ✓ Ser capaz de acceder a la consola de forma mejorada
- ✓ Ser capaz de crear entornos gráficos simples, de modo que el programa responda a las acciones del usuario sobre botones
- ✓ Poder leer la fecha y la hora del sistema
- ✓ Saber obtener información del entorno
- ✓ Llamar a otras órdenes del sistema
- ✓ Acceder a información disponible en una página web
- ✓ Comunicar con otro ordenador mediante red
- ✓ Ser capaz de crear juegos sencillos

UNIDAD 12. Gestión dinámica de la memoria

CONTENIDOS

Conceptos

- ✓ Memoria dinámica: motivación.
- ✓ Pilas
- ✓ Colas
- ✓ Listas
- ✓ Tablas hash
- ✓ Enumeradores
- ✓ Punteros. Operaciones con punteros.

Procedimientos

-
- ✓ Creación de pilas usando las estructuras existentes en el lenguaje
 - ✓ Colas usando las estructuras existentes en el lenguaje
 - ✓ Listas: nociones generales; ArrayList; SortedList
 - ✓ Características y uso de las tablas hash
 - ✓ Empleo de enumeradores
 - ✓ Acceso a punteros. Incremento y decremento de punteros y sus efectos.

Actividades de enseñanza-aprendizaje

- ✓ Crear pilas usando las estructuras existentes en el lenguaje
- ✓ Crear colas usando las estructuras existentes en el lenguaje
- ✓ Crear ArrayList, introducir en ellos datos simples y extraerlos
- ✓ Introducir y extraer structs y objetos en un ArrayList
- ✓ Manipulaciones y recorrido de un ArrayList
- ✓ Crear SortedList, introducir datos y extraer datos
- ✓ Crear tablas hash, introducir datos y extraer datos
- ✓ Empleo de enumeradores para recorrer estructuras dinámicas
- ✓ Obtener punteros a datos. Recorrer estructuras usando aritmética de punteros.
- ✓ Imitar pilas, colas y listas usando arrays y ArrayList

Criterios de evaluación

- ✓ Saber crear de pilas usando las estructuras existentes en el lenguaje
- ✓ Poder crear colas usando las estructuras existentes en el lenguaje
- ✓ Manejar correctamente ArrayList, tanto con datos simples como complejos
- ✓ Guardar y obtener información de un SortedList
- ✓ Saber guardar y leer información de una tabla hash
- ✓ Poder emplear enumeradores para recorrer una estructura dinámica
- ✓ Ser capaz de imitar pilas, colas y listas usando arrays y ArrayList

UNIDAD 13. Otras características avanzadas de C#**CONTENIDOS****Conceptos**

- ✓ Espacios de nombres
- ✓ Operadores para tratamiento de bits
- ✓ Enumeraciones
- ✓ Propiedades
- ✓ Parámetros “out”
- ✓ Operador coma
- ✓ Introducción a las expresiones regulares

Procedimientos

- ✓ Motivación y uso de espacios de nombres
- ✓ Operadores AND, OR, NOT, XOR, desplazamientos
- ✓ Enumeraciones como forma rápida de declaración de constantes
- ✓ Propiedades, get y set
- ✓ Parámetros “out” frente a parámetros por referencia
- ✓ Operador coma en órdenes “for”
- ✓ Ejemplos de expresiones regulares

Actividades de enseñanza-aprendizaje

- ✓ Crear de espacios de nombres y añadir fuentes a los espacios existentes
- ✓ Uso de las operaciones AND, OR, NOT, XOR, desplazamientos
- ✓ Creación de enumeraciones para listas de constantes
- ✓ Acceso a propiedades “al estilo de C#”
- ✓ Uso de parámetros “out” en vez de parámetros por referencia cuando la situación sea la adecuada
- ✓ Órdenes “for” con varias condiciones enlazadas mediante el operador coma
- ✓ Comprobación de patrones mediante expresiones regulares

Criterios de evaluación

- ✓ Ser capaz de crear de espacios de nombres y de añadir fuentes a los espacios existentes
- ✓ Usar correctamente las operaciones AND, OR, NOT, XOR y los desplazamientos a nivel de bits
- ✓ Saber crear de enumeraciones como alternativa a las listas de constantes
- ✓ Poder acceder a propiedades “al estilo de C#”
- ✓ Saber cuándo usar parámetros “out” en vez de parámetros por referencia
- ✓ Poder crear órdenes “for” con varias condiciones enlazadas mediante el operador coma
- ✓ Saber comprobación patrones básicos mediante expresiones regulares

UNIDAD 14. Depuración, prueba y documentación de programas

CONTENIDOS

Conceptos

- ✓ Aplicación de los conocimientos previos a la creación de aplicaciones de pequeño tamaño.
- ✓ Mantenimiento de aplicaciones ya creadas.
- ✓ Conceptos básicos sobre depuración
- ✓ Prueba de programas
- ✓ Documentación básica de programas

Procedimientos

- ✓ Creación de aplicaciones completas desde cero.
- ✓ Uso del depurador para encontrar fallos en aplicaciones: paso a paso, evaluación de variables.
- ✓ Nociones básicas de prueba de programas
- ✓ Consejos para comentar el código
- ✓ Generación de documentación a partir del código fuente

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.
- ✓ Búsqueda y corrección de fallos usando herramientas de depuración
- ✓ Correcta inclusión de comentarios en un fuente
- ✓ Generación de documentación automática a partir de los comentarios existentes en el código fuente

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.
- ✓ Legibilidad del fuente. Inclusión de comentarios en cualquier bloque de código que no sea evidente.
- ✓ Poder generar documentación automática a partir de los comentarios existentes en el código fuente.

UNIDAD 15. Proyecto final en lenguaje C#

CONTENIDOS

Conceptos

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

Procedimientos

- ✓ Nociones de planificación de productos software.
- ✓ 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 individual o 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 el alumno.

6. TEMPORALIZACIÓN

TEMA	SEMANAS
0. Conceptos básicos sobre programación	0
1. Toma de contacto con C#	1
2. Estructuras de control	3
3. Tipos de datos básicos	1
4. Arrays y estructuras	3
5. Funciones	3
6. Introducción a la programación Orientada a Objetos	2
7. Utilización avanzada de clases	2
8. Ficheros	3
9. Acceso a bases de datos relacionales	2
10. Persistencia de objetos	2
11. Bibliotecas de uso frecuente	2
12. Gestión dinámica de memoria	3
13. Otras características avanzadas de C#	1
14. Depuración, prueba y documentación	1
15. Proyecto final en lenguaje C#	3
	32

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:

- Moreno, J.C. *Programación (CFGS)*, Ra-Ma, 2011
- 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

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.
- Tutoriales de programación web en www.librosweb.es
- Cursos y foros de programación en www.aprendeaprogramar.com

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. Estas actividades no serán obligatorias, y no tendrán repercusión en la nota final, pero son altamente recomendables para los alumnos que tengan dificultades para seguir el ritmo normal del curso.

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 será, aproximadamente (más adelante se desglosa el detalle para el caso del grupo presencial y del grupo a distancia):

- 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:

- En el caso del grupo presencial, 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.
- En el proyecto globalizador deberá haber obtenido al menos la calificación de 5 sobre 10.

Con relación a las faltas de asistencia **no justificadas** (en el grupo presencial), 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. Deberán presentarse a una prueba final en junio los alumnos cuya nota media del curso sea inferior a 5,00.

Con relación al proyecto final, es importante insistir en que, como requisito

IMPRESINDIBLE para superar la asignatura, se deberá completar el proyecto (y su memoria), de modo que el proyecto sea “realmente utilizable”, tenga un nivel de dificultad razonable para un proyecto final, y muestre buenas costumbres de programación (en cuanto a estructuración, uso de comentarios y limpieza de código)

Por otra parte, dado que la asignatura, en su modalidad presencial, incluye enseñanza en inglés (ver apartado 11), para poder superar el curso deberá demostrarse una cierta destreza en inglés técnico, especialmente a nivel de lectura y comprensión de textos (la mayoría de los ejercicios de clase tendrán su enunciado en inglés). Sin esa destreza, no se calculará la nota media, y la asignatura se considerará suspensa.

En la modalidad presencial, los baremos detallados son los siguientes:

Calificación del **primer** trimestre:

- 70% nota ponderada de los exámenes de la siguiente manera:
 - Examen tema 0 y 1: 10%
 - Examen tema 2 y 3: 30%
 - Examen tema 4: 30%
 - Examen tema 5: 30%
- 30% ejercicios realizados en clase

Calificación del **segundo** trimestre:

- 40% primera evaluación
- 60% segunda evaluación, desglosada de la siguiente forma:
 - 50% media de los exámenes realizados
 - 50% ejercicios realizados en clase

Calificación **final**:

- 80% de las evaluaciones
 - 20% primera evaluación
 - 40% segunda evaluación

-
- 40% tercera evaluación, que se puede desglosar como:
 - 40% media de los exámenes realizados
 - 60% ejercicios realizados en clase
 - 10% proyecto globalizador
 - 10% actitud

En la modalidad a distancia, los baremos detallados son los siguientes:

Calificación del **primer** trimestre:

- 70% nota ponderada de los exámenes de la siguiente manera:
 - Examen tema 0 y 1: 15%
 - Examen tema 2 y 3: 25%
 - Examen tema 4: 30%
 - Examen tema 5: 30%
- 30% ejercicios realizados en el Aula Virtual

Calificación del **segundo** trimestre:

- 40% primera evaluación
- 60% segunda evaluación, de la siguiente forma:
 - 70% media de los exámenes realizados
 - 30% ejercicios realizados en el Aula Virtual

Calificación **final**:

- 80% de las evaluaciones
 - 20% primera evaluación
 - 40% segunda evaluación
 - 40% tercera evaluación

- 10% proyecto globalizador

- 10% actitud

10. RECUPERACIÓN DE PENDIENTES

En el bloque de programación estructurada y orientada a objetos, dado que se trata de conocimientos claramente incrementales, en los que cada nuevo tema se apoya en los anteriores, no habrá actividades extraordinarias de recuperación durante el curso: las posibles notas negativas se deberán compensar con ejercicios y exámenes posteriores (que, además, tendrán más ponderación a medida que avance el curso). Si aun así, algún alumno llega a final de curso sin haber logrado una media de aprobado, podrá realizar el examen final de junio, en el que se le evaluará de todos los conocimientos adquiridos a lo largo del curso. En caso de suspender dicho examen final, se propondrá al alumno una serie de actividades a realizar en verano, y deberá realizar un examen final extraordinario en septiembre, también de todo el contenido de este bloque.

En la modalidad a distancia, en caso de tener que realizar el examen final de junio, el alumno deberá presentar también una serie de actividades propuestas.

11. 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.

12. ENSEÑANZA BILINGÜE

En el citado Real Decreto, en su Artículo 6, “Enseñanza bilingüe” se detalla que:

- El currículo de este ciclo formativo incorpora la lengua inglesa de forma integrada en al menos dos módulos profesionales de entre los que componen la totalidad del ciclo formativo. Estos módulos se impartirán por el profesorado con atribución docente en los mismos y que, además, posea la habilitación lingüística correspondiente al nivel B2 del Marco Común Europeo de referencia para las lenguas.
- Al objeto de garantizar que la enseñanza bilingüe se imparta en los dos cursos académicos del ciclo formativo de forma continuada se elegirán módulos profesionales de ambos cursos.
- Los módulos susceptibles de ser impartidos en lengua inglesa son los señalados el anexo III.
- Con carácter excepcional, y para quienes lo soliciten, en el caso de alumnos o alumnas con discapacidad que puedan presentar dificultades en su expresión oral (parálisis cerebral, sordera...), se establecerán medidas de flexibilización y/o alternativas en el requisito de impartición de módulos en lengua inglesa, de forma que puedan cursar todas las enseñanzas de los módulos profesionales en su lengua materna.

En el caso del ciclo de Desarrollo de Aplicaciones Multiplataforma en el I.E.S. San Vicente, se ha optado por impartir en inglés los módulos de Programación (para el grupo de enseñanza presencial) y de Entornos de Desarrollo (para el grupo de enseñanza a distancia).

Esto supone que, para los alumnos del grupo presencial, los apuntes elaborados por el equipo docente estarán realizados en inglés, así como los exámenes y gran parte de los ejercicios de clase.

No se espera que el alumno pueda llegar a dominar la lengua inglesa con sólo 3 horas a la semana, pero sí que coja la costumbre de manejar textos técnicos en inglés, y que sea capaz de entender un enunciado de un problema o un texto explicativo básico.

13. MATERIALES Y RECURSOS DIDÁCTICOS.

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

HARDWARE:

- Un servidor Pentium IV.
- Veintidós estaciones de trabajo Pentium IV conectadas en red.
- Un sistema de proyección (proyector SVGA y pantalla).
- Conexión a Internet ADSL.

SOFTWARE:

- Sistemas operativos en red: Windows XP Professional, Ubuntu Linux 10.04.
- Entornos de desarrollo en lenguaje C#: Mono, SharpDevelop sobre la plataforma ".Net", Visual Studio (versión 2010 Express)