

PROGRAMACIÓN DEL MÓDULO PROFESIONAL:

FUNDAMENTOS DE PROGRAMACIÓN

CICLO FORMATIVO:

ADMINISTRACIÓN DE SISTEMAS INFORMÁTICOS.

Programación del curso 2009-2010
Profesora: Gabriela Cortés García.

1. INTRODUCCIÓN

El módulo se denomina FUNDAMENTOS DE PROGRAMACIÓN y se imparte en el primer curso del Ciclo formativo de grado superior correspondiente al título de “Técnico superior en Administración de Sistemas Informáticos”.

La referencia del sistema productivo de este módulo, la encontramos en la unidad de competencia número 4 del correspondiente R.D. de título: **Proponer y coordinar cambios para mejorar la explotación del sistema y las aplicaciones.**

Sus realizaciones son:

- Formular técnicamente los cambios y mejoras necesarios en el sistema y/o aplicaciones para proporcionar criterios de decisión a la persona autorizada
- Realizar, a su nivel, los cambios propuestos en el sistema y/o aplicaciones de acuerdo con las prestaciones requeridas.
- Realizar pruebas funcionales y de usuario previas a la implantación de los cambios desarrollados en el sistema y/o aplicaciones.
- Elaborar y mantener la documentación y guías del usuario descriptivas de los cambios y mejoras introducidas en el sistema y/o aplicaciones según las normas y procedimientos establecidos.

El módulo se imparte a lo largo de los tres trimestres de un curso, con una duración total de 285 horas a razón de 9 horas semanales.

El grupo de alumnos es bastante heterogéneo en cuanto a edad y a estudios anteriores. Por ello, las primeras semanas y en parte durante todo el curso, es importante considerar la atención personalizada o por grupos que hay que llevar a cabo. No es fácil conseguir buenos resultados al respecto puesto que si el alumno no adquiere los niveles mínimos de las Unidades de Trabajo previas a una dada, ésta le resulta más compleja de superar.

2. CAPACIDADES TERMINALES: OBJETIVOS

- **Estructuras de datos**

- . Variables, tipos de variables.
- . Registros, ficheros, "arrays", listas, árboles.
- . Algoritmos de utilización.
- . Aplicación de las estructuras a la resolución de problemas en programación.

- **Metodología de la programación y programación estructurada**

- . Características de los lenguajes estructurados de tercera generación.
 - . Estructuras e instrucciones típicas.
 - . Procedimientos y funciones.
 - . Paso de argumentos.
- . Características de la programación estructurada. Estructuras básicas.
- . Métodos de diseño de programas y datos de prueba en programación estructurada.
 - . Análisis descendente.
 - . Métodos orientados a las estructuras de datos.
- . Documentación y medidas de calidad en la programación.
- . Aplicación de métodos de diseño de programas y datos de prueba en programación estructurada.
- . Documentación de programas.

- **Programación en lenguajes estructurados: lenguaje C**

- . Entidades que maneja el lenguaje C: tipos de variables y estructuras de datos.
- . Instrucciones del lenguaje.
 - . Función y sintaxis.
 - . Declaración de estructuras.
 - . E/S.
 - . Instrucciones de control...
- . Aplicación práctica del lenguaje.

Módulo: Fundamentos de Programación

Dept. Informática

- . Diseño.
- . Procedimiento de codificación.
- . Obtención de código ejecutable.
- . Depuración de errores.

- . Funciones y librerías básicas del entorno de desarrollo.
- . Documentación del programador del lenguaje C.
- . Desarrollo de funciones sencillas de usuario.

- **Utilización de estructuras dinámicas: punteros en lenguaje C**
 - . Punteros, listas: pilas, colas. Árboles. Algoritmos de utilización.
 - . Funciones: paso de argumentos por parámetros y por dirección.
 - . Utilización de ficheros.
 - . Diseño y codificación de programas sobre: punteros, listas, pilas, colas. Árboles.
 - . Diseño y codificación de funciones.
 - . Diseño y codificación de programas sobre ficheros.

3. PROGRAMACIÓN

3.1 RELACION SECUENCIADA DE UNIDADES DE TRABAJO

La propuesta de programación está constituida por una relación de unidades de trabajo donde se integran y desarrollan, al mismo tiempo, alrededor de los procedimientos (contenidos organizadores), los conceptos (contenidos soporte), las actividades de enseñanza-aprendizaje y los criterios de evaluación.

De la estructura de los contenidos se pueden deducir cuatro grandes bloques constituidos de la siguiente manera:

Bloques	Título de las Unidades de trabajo
1 Conocimientos Básicos.	1. Algoritmos y programas. 2. Conceptos básicos de metodología de la programación. 3. C, un lenguaje estructurado. El compilador.
2	4. Comenzando a programar.

Técnicas de programación.	.5. Estructuras estáticas. 6. Estructuras externas. 7. Estructuras dinámicas.
3 Programación avanzada.	8. Programación avanzada.
4 Mantenimiento de programas.	9. Adaptación y/o creación de aplicaciones y/o funciones sencillas para el sistema.

La U.T. 1 tiene como fin presentar al alumno los conceptos básicos de la programación de tal manera que comience a familiarizarse con los términos, entornos, materiales y finalidades del módulo completo. Es una unidad eminentemente conceptual que pretende presentar de forma global el contenido que se verá durante todo el módulo.

La U.T. 2 va a presentar al alumno los métodos y técnicas que permitirán desarrollar buenos programas. Esto es la metodología de la programación. Asimismo se expondrán las herramientas de diseño de algoritmos, así como las pautas a seguir en su diseño y las técnicas de programación utilizadas en la actualidad. Se pretende que el alumno adquiera el conocimiento y destreza suficientes para la interpretación de problemas y el diseño y construcción de los algoritmos que los resuelvan. Los contenidos son de tipo conceptual y procedimental y serán complementados con los contenidos de las siguientes unidades.

La U.T. 3 hace una presentación de un lenguaje de programación procedimental estructurado, el C. Se pretende dar una visión general del lenguaje, de sus características, utilidad, ventajas, inconvenientes e implantación actual, y de la estructura de un programa en C. Asimismo se pretende que el alumno adquiera los suficientes conocimientos sobre el compilador y su entorno de trabajo(editor, depurador, librerías, etc.) como para poder empezar a codificar desde este momento.

La U.T. 4 pretende en su inicio que el alumno descubra los diferentes tipos de datos que se utilizan en C, para dedicarse después a la descripción de los tipos de datos sencillos que maneja el C y su forma de utilizarlos, así como de las estructuras de programación características del lenguaje, de manera que el alumno pueda empezar a resolver problemas sencillos, siguiendo siempre los mismos pasos: interpretación de los problemas, diseño del algoritmo utilizando alguna de las herramientas en la U.T. 2, codificación en C, pruebas, depuración y documentación. Asimismo se irán introduciendo de forma práctica los conceptos de programación modular con la creación y utilización de funciones.

La U.T. 5 presentará las primeras estructuras complejas de datos: las estructuras internas estáticas. Se enseñará al alumno a aplicar alguna de las herramientas de diseño de algoritmo a este tipo de estructuras, para continuar con el conocimiento de las características de estas estructuras en C a fin de pasar a la codificación. Es fundamental su

Módulo: Fundamentos de Programación

Dept. Informática

comprensión por el alumno dada la variedad de estas estructuras en C, su importancia, y su continua utilización de aquí en adelante. También se planteará el tema de los punteros que serán muy utilizados a partir de ahora. Al finalizar la unidad el alumno debe encontrarse en situación de poder resolver problemas de una cierta entidad. El contenido de esta unidad es eminentemente procedimental.

La U.T.6 presentará al alumno las estructuras externas de datos. El esquema a seguir es el mismo que en la unidad anterior: aplicación de la herramienta de diseño de algoritmos elegida al tratamiento de ficheros, estudio de las características y peculiaridades de los ficheros en C para la codificación y finalmente la obtención del programa ejecutable. La comprensión de estas estructuras es de gran importancia ya que son fundamentales en cuanto a que permitirán conseguir el almacenamiento de datos para su posterior utilización. Al finalizar la unidad el alumno debe haber adquirido los conocimientos y destrezas necesarios para el manejo de los ficheros, estructuras fundamentales en problemas de gestión. Es una unidad de contenido procedimental.

La U.T. 7 presentará al alumno las estructuras internas dinámicas de datos. Este tipo de estructuras son características del lenguaje C y le proporcionarán una forma muy flexible de gestionar la memoria. Se seguirá el esquema ya expuesto: concreción del problema, diseño del algoritmo, codificación, ejecución, prueba, depuración y documentación. Al finalizar la unidad el alumno debe haber adquirido los conocimientos y destrezas necesarios para el manejo de estas estructuras en problemas de gestión. El contenido es procedimental.

La U.T. 8 presenta todo aquello que, considerado como importante para el desarrollo de programas de gestión, no se ha visto hasta el momento. Así aparece como un cajón de sastre donde aplicar de forma conjunta lo que hasta ahora se ha visto en parcelas independientes, y todo aquello que por sus características o grado de dificultad el profesor ha preferido guardar para cuando el alumno haya adquirido un cierto conocimiento y manejo del lenguaje. Por esta razón aunque en la unidad se sugieren algunos contenidos deberá de ser cada profesor el que completará o rediseñará según sus propias necesidades. Se trata de una unidad procedimental.

La U.T. 9 trata de enfrentar al alumno a la creación y/o adaptación de aplicaciones dirigidas a mejorar los programas y/o el sistema 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 el momento, así como la ayuda que pueda prestar el profesor. Es una unidad eminentemente procedimental.

UNIDAD DE TRABAJO Nº 1: ALGORITMOS Y PROGRAMAS.

Contenidos Conceptuales	Contenidos Procedimentales
<ul style="list-style-type: none"> - Los sistemas de procesamiento de la información. - Algoritmos. - Aplicación informática. - Ciclo de vida de una aplicación informática: <ul style="list-style-type: none"> ▪ Diseño del programa. ▪ Instalación y explotación del programa. - Errores: <ul style="list-style-type: none"> ▪ Tipos de errores. - Programación: <ul style="list-style-type: none"> ▪ Tipos de programación. ▪ Calidad de los programas. - Documentación de los programas: <ul style="list-style-type: none"> ▪ Formas de documentación. - Nociones sobre estructuras de datos. <ul style="list-style-type: none"> Objetivos de un programa. ▪ Tipos sencillos de datos: <ul style="list-style-type: none"> • Identificadores. • Tipos de datos. • Constantes. • Variables. • Expresiones: <ul style="list-style-type: none"> ➤ Tipos. ➤ Operadores. ▪ Tipos complejos de datos: <ul style="list-style-type: none"> • Estructuras estáticas: <ul style="list-style-type: none"> ➤ Definición. ➤ Características ➤ Almacenamiento • Estructuras externas. Ficheros: <ul style="list-style-type: none"> ➤ Definición. ➤ Características. ➤ Almacenamiento. • Estructuras dinámicas: <ul style="list-style-type: none"> ➤ Definición. ➤ Características. ➤ Almacenamiento. • Bases de datos. 	<ul style="list-style-type: none"> - Manejo e interpretación del material bibliográfico. - Procesamiento de la información. - Descripción del ciclo de vida de una aplicación informática. - Interpretación de problemas. - Interpretación de algoritmos. - Interpretación de programas. - Interpretación de errores. - Descripción de las características que debe tener un buen programa. - Documentación de algoritmos y programas. - Identificación y utilización de los objetos de un programa.

**UNIDAD DE TRABAJO Nº 2:
CONCEPTOS BÁSICOS DE METODOLOGÍA DE LA
PROGRAMACIÓN.**

Contenidos Conceptuales	Contenidos Procedimentales
<ul style="list-style-type: none"> - Herramientas y notaciones para el diseño de algoritmos: <ul style="list-style-type: none"> ▪ Diagrama de flujo. ▪ Pseudocódigo. ▪ Tablas de decisión. ▪ Otros. - Estructura general de un programa: <ul style="list-style-type: none"> ▪ Partes de un programa: <ul style="list-style-type: none"> • Entrada. • Proceso. • Salida. ▪ Clasificación de las instrucciones: <ul style="list-style-type: none"> • De declaración. • Primitivas. • De control. • Compuestas. ▪ Variables auxiliares: <ul style="list-style-type: none"> • Contadores. • Acumuladores. • Switches. - Nociones básicas sobre técnicas de programación: <ul style="list-style-type: none"> ▪ Programación convencional. ▪ Programación estructurada: <ul style="list-style-type: none"> • Teorema. • Herramientas. ▪ Programación modular: <ul style="list-style-type: none"> • Subprogramas. • Procedimientos. • Funciones. • Recursividad. 	<ul style="list-style-type: none"> - Manejo e interpretación de los manuales y del material bibliográfico. - Interpretación del problema. - Elección de las estructuras de programación necesarias para la resolución de problema. - Construcción del algoritmo utilizando las estructuras elegidas. - Edición del algoritmo. - Realización de pruebas. - Corrección de los errores observados. - Documentación del programa.

UNIDAD DE TRABAJO Nº 3:

C, UN LENGUAJE ESTRUCTURADO. EL COMPILADOR.

Contenidos Conceptuales	Contenidos Procedimentales
<ul style="list-style-type: none"> - Lenguajes de programación: <ul style="list-style-type: none"> ▪ Tipos de lenguajes. ▪ Ensambladores. ▪ Intérpretes. ▪ Compiladores. - Compiladores frente a intérpretes. - Historia del lenguaje C. - C como lenguaje estructurado. - Ciclo de creación de un programa. - Estructura general de un programa: <ul style="list-style-type: none"> ▪ Bloques de declaración. ▪ Bloques de ejecución. - Estructura de un programa C: <ul style="list-style-type: none"> ▪ Directrices para el preprocesador. ▪ Declaración de variables y funciones externas. ▪ Declaración de variables globales y funciones prototipo. ▪ Funciones (main(),etc.). - Un editor de texto. Elementos. - Funciones: <ul style="list-style-type: none"> ▪ De usuario. ▪ De librería. - La compilación. Características del compilador que se empleará. Opciones más utilizadas. - El enlazado. Opciones más utilizadas. - Librerías de C. Librerías más utilizadas. - Ejecución de un programa. - La depuración. Opciones del depurador. 	<ul style="list-style-type: none"> - Manejo e interpretación de los manuales y del material bibliográfico. - Utilización de los recursos del sistema. - Creación de una guía resumen de instalación y utilización del compilador empleado, a partir de los manuales del producto. - Creación de una guía resumen de utilización del editor de texto empleado, a partir de los manuales del producto. - Descripción e identificación de los distintos elementos del listado de un programa fuente escrito en C. - Identificación de las distintas estructuras de programación que aparecen en el listado fuente. - Edición de un programa a partir de su listado fuente. - Utilización del compilador de C empleado. - Realización de pruebas. - Corrección de los errores observados. - Documentación del programa.

UNIDAD DE TRABAJO Nº 4: COMENZANDO A PROGRAMAR.

Contenidos Conceptuales	Contenidos Procedimentales
<ul style="list-style-type: none"> - Tipos de datos: <ul style="list-style-type: none"> ▪ Simples. ▪ Estructuras de datos. - Elementos del lenguaje C: <ul style="list-style-type: none"> ▪ Caracteres de C ▪ Tipos de datos: <ul style="list-style-type: none"> • Fundamentales. • Derivados. ▪ Nombres de tipos. Typedef. ▪ Constantes. ▪ Identificadores. ▪ Palabras claves. ▪ Comentarios. ▪ Variables. ▪ Declaración de constantes. ▪ Expresiones numéricas. ▪ Operadores. ▪ Evaluación de operadores. ▪ Conversión de tipos. - Accesibilidad de variables.Ámbito: <ul style="list-style-type: none"> ▪ Variables locales y globales. ▪ Clases de almacenamiento. ▪ Variables externas. ▪ Variables internas. - Sintaxis de las sentencias y funciones de C. - Entrada y salida estándar por consola. <ul style="list-style-type: none"> ▪ Funciones de entrada y salida con formato. ▪ Otras funciones de entrada y salida de caracteres. - Estructuras de programación: <ul style="list-style-type: none"> ▪ Sentencias de asignación. ▪ Sentencias de control de programa: <ul style="list-style-type: none"> • Sentencias de selección. • Sentencias de iteración. • Sentencias de salto. - Programación modular y estructurada: <ul style="list-style-type: none"> ▪ Funciones: Declaración. <ul style="list-style-type: none"> • Definición. • Llamada. • Pasando argumentos. Formas. - Funciones predefinidas en C: <ul style="list-style-type: none"> ▪ Funciones matemáticas. ▪ Otras funciones de interés. 	<ul style="list-style-type: none"> - Manejo e interpretación de los manuales y del material bibliográfico. - Identificación de las distintas estructuras de programación que aparecen en un listado fuente. - Interpretación del problema. - Elección de los objetos de programación necesarios para la resolución del problema. - Construcción del algoritmo utilizando tipos simples de datos. - Codificación del algoritmo. - Compilación del programa fuente. - Montaje(enlazado) del programa objeto y las librerías necesarias. - Realización de pruebas. - Corrección de los errores observados. - Documentación del programa.

UNIDAD DE TRABAJO Nº 5: ESTRUCTURAS ESTÁTICAS.

Contenidos Conceptuales	Contenidos Procedimentales
<ul style="list-style-type: none"> - Estructuras estáticas. Definiciones y características. - Tablas o arrays: <ul style="list-style-type: none"> ▪ Características: <ul style="list-style-type: none"> • Tipos de tablas. • Declaración de tablas. • Representación de tablas. • Operaciones con tablas. ▪ Paso de tablas a funciones. ▪ Cadenas de caracteres: <ul style="list-style-type: none"> • Operaciones con cadenas. • Funciones para manipular cadenas de caracteres. • Funciones para la conversión de datos. • Funciones para la conversión de caracteres. - Estructuras: <ul style="list-style-type: none"> ▪ Características: <ul style="list-style-type: none"> • Creación. • Operaciones con estructuras. ▪ Arrays de estructuras. ▪ Paso de estructuras a funciones. ▪ Arrays y estructuras dentro de estructuras. - Uniones. - Funciones: <ul style="list-style-type: none"> ▪ Prototipos. Definición. Paso de argumentos. - Punteros: <ul style="list-style-type: none"> ▪ Características: <ul style="list-style-type: none"> • Creación. • Utilización. • Operaciones con punteros. ▪ Punteros y arrays: <ul style="list-style-type: none"> • Punteros a cadenas de caracteres. • Inicialización de cadenas. ▪ Arrays de punteros. Punteros a punteros: <ul style="list-style-type: none"> • Inicialización de un array de punteros a cadenas de caracteres. ▪ Punteros a estructuras. ▪ Punteros a uniones. ▪ Punteros a funciones. 	<ul style="list-style-type: none"> - Manejo e interpretación de los manuales y del material bibliográfico. - Identificación de las distintas estructuras de datos. - Interpretación del problema. - Elección de las estructuras estáticas necesarias para la resolución del problema. - Construcción del algoritmo utilizando las estructuras estáticas elegidas. - Codificación del algoritmo. - Compilación del programa fuente. - Realización de pruebas. - Corrección de los errores observados. - Documentación del programa.

UNIDAD DE TRABAJO Nº 6: ESTRUCTURAS EXTERNAS.

Contenidos Conceptuales	Contenidos Procedimentales
<ul style="list-style-type: none"> - Archivos o ficheros: <ul style="list-style-type: none"> ▪ Terminología. ▪ Características. ▪ Clasificación según su uso. ▪ Soportes. Tipos. - Organización de archivos: <ul style="list-style-type: none"> ▪ Secuencial. ▪ Aleatoria o directa. ▪ Secuencial indexada. - Forma de acceso a archivos: <ul style="list-style-type: none"> ▪ Secuencial. ▪ Directa. ▪ Dinámica. - Los archivos en C. Punteros ficheros: <ul style="list-style-type: none"> ▪ Archivos de texto. ▪ Archivos binarios. - Operaciones sobre archivos: <ul style="list-style-type: none"> ▪ Creación. ▪ Consulta. ▪ Actualización. ▪ Clasificación. ▪ Destrucción. ▪ Otros. - Procesamiento de archivos secuenciales: <ul style="list-style-type: none"> ▪ Creación. ▪ Consulta. ▪ Actualización. - Procesamiento de archivos directos: <ul style="list-style-type: none"> ▪ Clave-dirección. ▪ Colisiones. ▪ Operaciones. - Procesamiento de archivos secuenciales-indexados. Utilización de librerías específicas: <ul style="list-style-type: none"> ▪ Clave. ▪ Operaciones. - Métodos de tratamiento de archivos: <ul style="list-style-type: none"> ▪ Búsqueda : <ul style="list-style-type: none"> • Secuencial. • Binaria. • Mediante transformación de claves. Colisiones. ▪ Partición: <ul style="list-style-type: none"> • Por contenido. • En secuencias. 	<ul style="list-style-type: none"> - Manejo e interpretación de los manuales y del material bibliográfico. - Interpretación del problema. - Elección de las estructuras externas necesarias para la resolución del problema. - Construcción del algoritmo utilizando las estructuras externas elegidas. - Codificación del algoritmo. - Compilación del programa fuente. - Realización de pruebas. - Corrección de los errores observados. - Documentación del programa.

<ul style="list-style-type: none">▪ Mezcla:<ul style="list-style-type: none">• Con registro centinela.• Controlada por valor de clave máxima.• Controlada por fin de archivo.▪ Clasificación:<ul style="list-style-type: none">• Por mezcla directa.• Por mezcla equilibrada.▪ Ordenación. Métodos.	
--	--

UNIDAD DE TRABAJO Nº 7: ESTRUCTURAS DINÁMICAS.

Contenidos Conceptuales	Contenidos Procedimentales
<ul style="list-style-type: none"> - Estructuras dinámicas. Definición y características. - Punteros. - Asignación dinámica de memoria. Funciones de librería. - Arrays dinámicos. - Abstracción de datos. - Pilas: <ul style="list-style-type: none"> ▪ Terminología. ▪ Representación. ▪ Operación con pilas. - Colas: <ul style="list-style-type: none"> ▪ Terminología. ▪ Representación. ▪ Operaciones con colas. - Listas: <ul style="list-style-type: none"> ▪ Terminología. ▪ Representación. ▪ Operaciones con listas. ▪ Tipos: <ul style="list-style-type: none"> • Enlazadas. • Circulares. • Doblemente enlazadas. - Árboles: <ul style="list-style-type: none"> ▪ Terminología. ▪ Representación. ▪ Tipos. ▪ Operaciones con árboles. - Grafos: <ul style="list-style-type: none"> ▪ Terminología. ▪ Representación. 	<ul style="list-style-type: none"> - Manejo e interpretación de los manuales y del material bibliográfico. - Gestión de la memoria del sistema. - Interpretación del problema. - Elección de las estructuras dinámicas necesarias para la resolución del problema. - Construcción del algoritmo utilizando las estructuras dinámicas elegidas. - Codificación del algoritmo. - Compilación del programa fuente. - Realización de pruebas. - Corrección de los errores observados. - Documentación del programa.

UNIDAD DE TRABAJO Nº 8: PROGRAMACIÓN AVANZADA.

Contenidos Conceptuales	Contenidos Procedimentales
<ul style="list-style-type: none"> - Completar y extrapolar el conocimiento de las estructuras de datos vistas hasta ahora. - Recursividad. - El preprocesador de C. - Utilidades que acompañan al compilador. El entorno de desarrollo. - Librerías: <ul style="list-style-type: none"> ▪ Librerías que acompañan al compilador. ▪ Librerías de usuario. Técnicas a emplear para la creación y modificación de librerías. - Comunicaciones. - Servicios del DOS y del BIOS: <ul style="list-style-type: none"> ▪ Operadores de bits. ▪ Campos de bits. ▪ Funciones para llamar al DOS. - Utilización y acceso memoria de la pantalla. Acceso a color y gráficos. <ul style="list-style-type: none"> ▪ Funciones de gráficos. ▪ Librerías de gráficos. - C y DOS: <ul style="list-style-type: none"> ▪ Directorios y caminos. ▪ Definiciones generales. ▪ Redirección de la salida. ▪ Redirección de entrada. ▪ Interconexión de entradas y salidas estándar. ▪ Operaciones con directorios. ▪ Funciones para el control de directorios. - La programación en entornos de teleprocesos. - C y otros lenguajes. 	<ul style="list-style-type: none"> - Manejo e interpretación de los manuales y del material bibliográfico. - Resolución de problemas que requieren necesariamente el uso de varios tipos de estructuras de datos. - Interpretación del problema. - Análisis de los niveles de abstracción de datos. - Elección de las estructuras necesarias para la resolución del problema. - Construcción del algoritmo utilizando las estructuras elegidas. - Codificación del algoritmo. - Compilación del programa fuente. - Realización de pruebas. - Corrección de errores observados. - Documentación del programa. - Creación de funciones. - Creación de ficheros de cabeceras. - Creación de librerías.

UNIDAD DE TRABAJO Nº 9:**ADAPTACIÓN Y/O CREACIÓN DE APLICACIONES Y/O FUNCIONES
SENCILLAS PARA EL SISTEMA.**

Contenidos Conceptuales	Contenidos Procedimentales
<ul style="list-style-type: none"> - Utilización de los conocimientos adquiridos anteriormente. - Técnicas de programación. <ul style="list-style-type: none"> ▪ Programación convencional. ▪ Programación estructurada: <ul style="list-style-type: none"> • Teorema. • Herramientas. ▪ Programación modular: <ul style="list-style-type: none"> • Subprogramas. • Procedimientos. • Funciones. • Recursividad. - Abstracción de datos. - Creación de funciones. - Ficheros de cabecera. - Librerías de funciones. 	<ul style="list-style-type: none"> - Manejo e interpretación de los manuales y del material bibliográfico. - Reconocimiento de las soluciones que se deben obtener o que, en su caso, se obtienen con el programa que se desea modificar. - Interpretación del problema o, en su caso, del código fuente del programa, o módulo que se desea modificar. - Interpretación de los cambios y adaptación que se pretenden realizar sobre el programa existente, sí como su viabilidad. - Diseño del algoritmo de la nueva aplicación o, en su caso, reconocimiento de las modificaciones a realizar sobre el algoritmo inicial y sus posibles consecuencias. - Elección de las estructuras necesarias para la resolución del nuevo problema o, en su caso, de los cambios planteados. - Modificación de los algoritmos afectados por los cambios, utilizando las estructuras elegidas. - Codificación de la nueva aplicación o, en su caso, de los módulos afectados por la modificación. - Compilación de los programas fuente. - Creación de los ficheros cabecera y de librerías necesarios. - Realización de pruebas con los módulo modificados. - Corrección de errores observados - Documentación, en su caso, de los cambios realizados y de su repercusión en el resto del programa. - Documentación del programa.

4. METODOLOGÍA DIDÁCTICA

El módulo **Fundamentos de programación** es eminentemente **procedimental** y tanto los contenidos soporte (conceptos) como los contenidos organizadores (procedimientos), están interrelacionados en todas las unidades de trabajo del mismo. Teniendo esto en cuenta, se aplicarán los siguientes aspectos en cuanto a metodología didáctica:

- Proponer constantemente la obtención de algoritmos y programas que resuelvan problemas, aplicando alternativamente las siguientes estrategias:
 1.
 - Explicación teórica de los conceptos básicos necesarios.
 - Presentación de un problema a resolver.
 - Explicación de posibles soluciones.
 - Ampliación de conocimientos mediante trabajos de investigación, utilizando distintos recursos y materiales.
 - Evaluación y reflexión sobre los resultados obtenidos.
 2.
 - Presentación del problema a resolver.
 - Identificación de los conocimientos necesarios.
 - Adquisición de dichos conocimientos utilizando los materiales convenientes.
 - Evaluación y reflexión sobre los resultados obtenidos.
- Aplicar una metodología personalizada en la que el profesor ajuste la ayuda pedagógica a las diferentes necesidades del alumno y facilite recursos que permitan dar respuestas a las diversas motivaciones, interés y capacidades que presentan los alumnos en estas edades. Se plantearán actividades de recuperación tras cada Unidad de Trabajo para los alumnos que las necesiten, ya que difícilmente podrán superar una unidad sin haberlo hecho de las precedentes.
- Contemplar contenidos que resulten **relevantes** para los alumnos y que abarquen diversas áreas de la actividad empresarial o institucional.
- Potenciar como fuentes de aprendizaje los propios compañeros del alumno, mediante los trabajos en grupo, planteamiento de debates, acoger inquietudes, etc.
- Realizar periódicamente, por ejemplo al concluir cada unidad de trabajo, actividades de análisis y estudio que permitan relacionar conceptos previos y conocimientos aprendidos.
- Selección de tipos de contenidos de interés para los alumnos, estableciendo actividades de descubrimiento o de adquisición de destrezas básicas.

5. PROCEDIMIENTOS DE EVALUACIÓN DEL APRENDIZAJE DE LOS ALUMNOS

La evaluación será continua, formativa y sumativa, considerándose además de las pruebas objetivas, el trabajo en clase, el progreso, el interés por el módulo, la atención, etc.

EVALUACIÓN INICIAL

Al comienzo del curso se realizará un cuestionario que permita detectar los conocimientos, intereses profesionales, experiencia laboral, etc, relacionados con el módulo.

EVALUACIÓN FORMATIVA

Utilizando la observación y el análisis de los trabajos desarrollados, se evaluarán las siguientes actividades y actitudes de los alumnos:

- La colaboración en el trabajo en grupo
- La asistencia regular a clase
- La puntualidad
- La correcta utilización del material y equipos
- Actividades extraescolares
- Iniciativas del alumno
- Participación en clase
- Realización y presentación de los trabajos solicitados por el profesor

EVALUACIÓN SUMATIVA

Al final de cada Unidad de Trabajo o de ciertos bloques de contenidos, fundamentales para proseguir el desarrollo del módulo, se realizarán pruebas específicas de evaluación, orales o escritas llevadas a cabo por el alumno de forma individual.

La Unidad de Trabajo nº 10 consiste en la realización de un proyecto que englobe la materia más importante del módulo.

6. CRITERIOS DE CALIFICACIÓN

Se aplicarán los establecidos en el Proyecto curricular de ciclo:

En cada una de las evaluaciones se calificarán los siguientes conceptos:

Módulo: Fundamentos de Programación

Dept. Informática

- Actividades de enseñanza-aprendizaje (trabajo del alumno durante la evaluación), asistencia y actitud 20%.
- Actividades específicas de evaluación 80%

Además para poder **superar el módulo** es necesario:

1. Que las ausencias a clase no superen las establecidas en las normas de convivencia del instituto .
2. Cumplir los planteamientos programados en las actividades de enseñanza aprendizaje.
3. Que la actitud hacia el profesor y los compañeros sea correcta.
4. Si hay proyecto globalizador, obtener en éste al menos la calificación de 5 sobre 10.
5. En las actividades específicas de evaluación , obtener al menos una calificación de 5 sobre 10.

7. PLAN DE RECUPERACIÓN

Se trata de una evaluación continua como se desprende de la metodología empleada, y por tanto, no existirán pruebas de recuperación específicas (a no ser que el profesor lo considere necesario).

Hay que señalar que no se llevará a cabo evaluación extraordinaria, por tener una suma horaria superior al 25% de la duración del total de los módulos de primer curso. Los alumnos que no superen el módulo repetirán curso con este módulo.

8. MATERIALES Y RECURSOS DIDÁCTICOS

Los recursos materiales y didácticos que se utilizarán en este módulo son:

Aula específica con:

- Pizarra.
- Retroproyector y pantalla.
- Ordenadores
- Impresoras.
- Borlandc C++
- Manuales de Borlandc C++
- Revistas especializadas en Informática: Pc World, Programación Actual
- Material fungible:
 - Disquetes
 - Papel

Bibliografía

- Para los alumnos:
 - Apuntes elaborados por el profesor
 - Los siguientes libros de consulta:
 - ✘ Metodología de la programación (2ª Edición)
E. Alcalde / M. García
McGraw-Hill
 - ✘ Metodología de la Programación: programación estructurada
Mª D. Alonso / S. Rumeu
Paraninfo
 - ✘ Fundamentos de Programación. Algoritmos y estructuras de datos
Luis Joyanes Aguilar
McGraw-Hill
 - ✘ Programación en Lenguajes Estructurados
Miguel A. Sutil / Antonio Garrido
Síntesis
 - ✘ Programación estructurada y lenguajes de programación
Mª Ángeles Sanchez/Félix Chamorro/José Manuel Molina/ Vicente Matellán
McGraw-Hill
 - ✘ Curso de Programación C/C++
Fco. Javier Ceballos
Rama

Módulo: Fundamentos de Programación

Dept. Informática

✘ C guía de autoenseñanza

Herbert Schildt

MCGraw-Hill

✘ C++ guía de autoenseñanza

Herbert Schildt

MCGraw-Hill

- Para profesores de la especialidad, además de los libros anteriores, los siguientes:

✘ C++ Manual de referencia

Herbert Schildt

McGraw-Hill

✘ Manual de Borland C++ 4.0

Chirs H. Pappas/William H. Murray, III

McGraw-Hill

✘ Programación orientada a objetos con C++

Fco. Javier Ceballos

Rama

✘ Programación de aplicaciones Windows con Borland C++ y Objectwindows

Angel Franco García

McGraw-Hill

✘ Lenguaje C y Estructuras de datos

Juan F. garcía de Sola/Vicente Garcerán Hdez

McGraw-Hill

✘ Programación orientada a objetos. teoría y técnicas OOP para desarrollo de software

Manuel Alfonseca / Alfonso Alcalá

Anaya

9. TEMAS TRANSVERSALES.

En el desarrollo de las Unidades de Trabajo del módulo se irán incorporando, en cada una de las actividades de enseñanza-aprendizaje, los temas transversales, principalmente en los contenidos conceptuales.

Los temas transversales a desarrollar son:

- La paz, la solidaridad y la tolerancia.
- El respeto por el medio ambiente y la ecología.
- El derecho a la libertad y el respeto por los demás, aceptar la crítica y el error.
- Conductas de seguridad e higiene en el trabajo.
- Conductas de educación y buenos modales.

- Educación para la igualdad.
- Educación para el consumidor.
- Educación moral y cívica.
- Educación vial.

10.MEDIDAS DE ATENCIÓN A LA DIVERSIDAD Y ADAPTACIONES CURRICULARES.

Se prestara especial atención a las medidas dirigidas al desarrollo de la personalidad, al desarrollo social y al desarrollo intelectual del alumno.

- Desarrollo de la personalidad:
 - Debido a la programación en lenguaje el lenguaje C de las funciones y programas que se realizarán en clase, el alumno incrementará fácilmente su autoestima ya que podrá observar inmediatamente el fruto de su trabajo y esfuerzo, si a esto se le une que las funciones que tenga que implementar tienen una finalidad concreta para que así pueda comprobar como su trabajo tiene una función determinada.
- Desarrollo social:
 - Las prácticas se enfocarán para, que aprovechando la capacidad de modularidad que posee el lenguaje C, el grupo pueda dividir el trabajo que se plantea en otros más pequeños y sencillos, siempre bajo el control del profesor, y encargarse cada pareja del desarrollo de una parte, para que después de la unión de todos los trabajos obtengamos un programa que realice la función pedida, con lo que obtendremos que los alumnos se acostumbren al trabajo en grupo y colaborar entre todos para conseguir un fin común.
- Desarrollo intelectual:
 - Debido a la programación en un lenguaje estructurado el alumno también desarrollará de forma automática el pensamiento abstracto.

Es importante tener en cuenta que al tratarse de un ciclo de grado superior no es susceptible de adaptaciones curriculares.