

DESARROLLO DE FUNCIONES EN EL SISTEMA INFORMÁTICO

CICLO FORMATIVO DE GRADO SUPERIOR

ADMINISTRACIÓN DE SISTEMAS INFORMÁTICOS

PROFESORA: Rosa M^a Medina Gómez

ÍNDICE

INTRODUCCIÓN.....	3
OBJETIVOS.....	3
CAPACIDADES TERMINALES.....	3
ORGANIZACIÓN DE LOS CONTENIDOS.....	4
ESTRUCTURA DE LOS CONTENIDOS.....	4
SECUENCIA DE CONTENIDOS.....	5
TEMPORALIZACIÓN.....	22
CRITERIOS DE EVALUACIÓN.....	22
BIBLIOGRAFÍA.....	22
ACTIVIDADES COMPLEMENTARIAS Y EXTRAESCOLARES.....	22
MATERIALES Y RECURSOS DIDÁCTICOS.....	23
MEDIDAS DE ATENCIÓN A LA DIVERSIDAD.....	23

INTRODUCCIÓN

Cada vez es más frecuente personalizar el entorno de trabajo y las aplicaciones que utiliza el usuario final. Esto supone tener que ofrecer ciertas funciones que mejoren, o clarifiquen aquellas que de por sí ofrece el sistema operativo con que se trabaja. En el caso de UNIX, o de alguna de sus implementaciones (Linux) esto se consigue modificando o creando nuevas funciones escritas en C, que es el lenguaje en que están escritos estos sistemas operativos. Por esta razón, es preciso tener un amplio conocimiento del sistema operativo, y del lenguaje de programación C.

OBJETIVOS

Los objetivos del módulo son los siguientes:

- ✓ Interpretar y aportar soluciones a las necesidades y requerimientos funcionales formulados por los usuarios.
- ✓ Aplicar la metodología de desarrollo, elegir una estructura para los datos y codificar el programa en lenguajes estructurados.
- ✓ Establecer y aplicar procedimientos que aseguren la integridad, disponibilidad y confidencialidad de la información.
- ✓ Definir y proponer cambios y mejoras en el sistema y aplicaciones encaminadas a optimizar las prestaciones, manteniéndose informado de las innovaciones, tendencias, tecnología y normativa aplicable.
- ✓ Organizar y dirigir tareas colectivas cooperando en la superación de las dificultades que se presenten, con una actitud tolerante hacia las ideas de los compañeros y subordinados.
- ✓ Mantener relaciones fluidas con los miembros del grupo funcional en el que se integre, responsabilizándose de la consecución de los objetivos asignados al grupo.

Este módulo pretende conseguir las siguientes realizaciones profesionales:

- ✓ 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 introducidos en el sistema y/o aplicaciones según las normas y procedimientos establecidos.

CAPACIDADES TERMINALES

- ✓ Las capacidades terminales asociadas al módulo están determinadas en los proyectos curriculares y son las siguientes:

- 1.Desarrollar un programa ejecutable utilizando las interfaces de programación que ofrece un sistema operativo monousuario, multiusuario y de red.
- 2.Establecer procedimientos de prueba para verificar el sistema y los programas desarrollados.
- 3.Elaborar documentación sobre el sistema y los cambios realizados.
- 4.Valorar técnica y económicamente la implicación de los cambios en un sistema.

ORGANIZACIÓN DE LOS CONTENIDOS

ESTRUCTURA DE LOS CONTENIDOS

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

A su vez se han tenido en cuenta los conocimientos previos de los alumnos adquiridos en los módulos del primer curso *Fundamentos de programación y Sistemas informáticos monousuario y multiusuario*.

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

- 1.Administración del sistema. Los conceptos y las características principales de Linux.
- 2.Herramientas de desarrollo en lenguaje C más características sobre sistemas Unix.
- 3.Descripción y utilización de las llamadas al sistema para las diferentes necesidades y actividades del sistema operativo.

SECUENCIA DE CONTENIDOS

UNIDAD 1. SISTEMAS DE FICHEROS

OBJETIVOS

- ✓ Conocer el significado que tiene el sistema de ficheros en los sistemas operativos.
 - ✓ Conocer la estructura de datos que conforma un sistema de ficheros.
 - ✓ Aprender las operaciones básicas que se pueden realizar sobre un sistema de ficheros.
 - ✓ Conocer los diferentes tipos de ficheros que soporta el sistema de ficheros de Linux; directorios, ficheros ordinarios, pipes y enlaces.
 - ✓ Conocer la estructura de directorios del sistema de ficheros de Linux.
 - ✓ Conocer las operaciones básicas de los intérpretes de comandos más comunes de Linux.
 - ✓ Conocer los diferentes sistemas que permiten compartir archivos entre Linux y Windows.
 - ✓ Conocer diferentes tipos de software que permitan la creación de imágenes para la restauración del sistema de archivos, y la clonación de discos duros.
-

CONTENIDOS

Conceptos

- ✓ Sistema de ficheros.
- ✓ Punto de montaje.
- ✓ Tipos de ficheros.
- ✓ Inodos.
- ✓ Enlaces.
- ✓ Organización del sistema de ficheros.
- ✓ Intérprete de comandos.
- ✓ Planos de ejecución.
- ✓ Fichero de proceso por lotes.
- ✓ Servidores y clientes FTP y SAMBA.
- ✓ UDPCAST, Ghost, G4L, Partimage.

Procedimientos

- ✓ Manejo e interpretación de los manuales y del material bibliográfico.

- ✓ Identificación de las diferentes características de los sistemas de archivos.
- ✓ Manejo de las operaciones sobre el sistema de archivos Linux.
- ✓ Identificación de la función del intérprete de comandos.
- ✓ Desarrollar ficheros de órdenes que permitan realizar operaciones repetitivas.
- ✓ Editar la configuración de servidores FTP y SAMBA, e iniciar los servicios.
- ✓ Utilizar los clientes FTP y SAMBA para transmitir archivos.
- ✓ Realizar clonación de discos duros.

CRITERIOS DE EVALUACIÓN

- ✓ Describir las operaciones que hay que realizar para montar una partición lógica en el sistema de ficheros de Linux.
- ✓ Describir la estructura de un inodo.
- ✓ Citar los principales comandos que se utilizan sobre la estructura de ficheros de Linux.
- ✓ Crear correctamente los diferentes tipos de enlaces.
- ✓ Realizar ficheros script sencillos.
- ✓ Intercambiar ficheros y sistemas completos utilizando la red.

UNIDAD 2. ADMINISTRACIÓN DEL SISTEMA

OBJETIVOS

- ✓ Conocer las operaciones básicas para administrar el sistema operativo Linux.
 - ✓ Conocer los métodos de seguridad que dispone Linux para el control de acceso a los ficheros.
 - ✓ Diseñar los ficheros de configuración de varios dispositivos físicos del ordenador; impresora, CD-ROM Y comunicaciones de red.
-

CONTENIDOS

Conceptos

- ✓ Administración del sistema.
- ✓ Seguridad del sistema de ficheros.
- ✓ Permisos de los ficheros.
- ✓ Ficheros de configuración.

Procedimientos

- ✓ Manejo e interpretación de los manuales y del material bibliográfico.
- ✓ Identificación de los diferentes niveles de seguridad.
- ✓ Identificación de las tareas del administrador del sistema.
- ✓ Interpretación de los permisos de accesos de los ficheros.
- ✓ Identificación de los ficheros que configuran el comportamiento de los dispositivos físicos instalados en el sistema.

CRITERIOS DE EVALUACIÓN

- ✓ Citar las principales atribuciones del administrador del sistema.
- ✓ Realizar correctamente el procedimiento de creación de nuevos grupos y usuarios del sistema.
- ✓ Utilizar correctamente las operaciones relacionadas con los permisos de ficheros.
- ✓ Justificar el uso de los ficheros de configuración de la shell.
- ✓ Realizar correctamente la instalación de un dispositivo hardware.

UNIDAD 3. EL LENGUAJE DE PROGRAMACIÓN C++

OBJETIVOS

- ✓ Desarrollar pequeños programas en C++ utilizando simples instrucciones de entrada/salida.
 - ✓ Aprender a utilizar el tipo cadena de C++.
-

CONTENIDOS

Conceptos

- ✓ Clases.
- ✓ Sobrecarga de funciones.
- ✓ Sobrecarga de operadores.
- ✓ Cadenas de C++.

Procedimientos

- ✓ Manejo e interpretación de los manuales y del material bibliográfico.
- ✓ Identificación de las diferencias entre C y C++.
- ✓ Construcción de un proyecto simple con C++ y un entorno de desarrollo bajo el sistema operativo Linux.

CRITERIOS DE EVALUACIÓN

- ✓ Citar las principales características de C++.
- ✓ Identificar correctamente los ficheros que forman parte de un proyecto de C++.
- ✓ Utilizar correctamente el entorno de desarrollo Kdevelop.

UNIDAD 4. PROCESOS

OBJETIVOS

- ✓ Conocer la unidad fundamental de ejecución del sistema operativo Linux, el proceso.
 - ✓ Conocer la estructura de los procesos de Linux.
 - ✓ Conocer la utilidad de las señales más importantes.
 - ✓ Aprender a realizar un seguimiento del estado de ejecución un proceso.
-

CONTENIDOS

Conceptos

- ✓ Proceso.
- ✓ Atributos de los procesos.
- ✓ Estado de ejecución de un proceso.
- ✓ Señales.
- ✓ Procesos del sistema, demonios.

Procedimientos

- ✓ Manejo e interpretación de los manuales y del material bibliográfico.
- ✓ Identificación de los atributos de un proceso.
- ✓ Identificación del estado de ejecución de un proceso.
- ✓ Interpretación de la información mostrada en la monitorización de los procesos.

CRITERIOS DE EVALUACIÓN

- ✓ Describir el procedimiento de creación de un nuevo proceso.
- ✓ Citar los principales atributos de un proceso.
- ✓ Citar los estados posibles de ejecución de un proceso.
- ✓ Utilizar correctamente los comandos que envían señales a los procesos.
- ✓ Utilizar correctamente los comandos que muestran el estado de ejecución de los procesos.
- ✓ Generar procesos que inicien su ejecución utilizando las alarmas del sistema.

UNIDAD 5. COMPILACIÓN DEL KERNEL

OBJETIVOS

- ✓Adecuar el Kernel de Linux a las características del sistema.
-

CONTENIDOS

Conceptos

- ✓Compilación del Kernel.

Procedimientos

- ✓Manejo e interpretación de los manuales y del material bibliográfico.
- ✓Identificación de los componentes físicos instalados en el sistema.

CRITERIOS DE EVALUACIÓN

- ✓Recompilar el kernel de Linux.

UNIDAD 6. ENTORNO DE PROGRAMACIÓN

OBJETIVOS

- ✓ Editar código en lenguaje C utilizando alguno de los editores soportados.
 - ✓ Conocer las características de la compilación de programas en C sobre Linux.
 - ✓ Utilizar los compiladores GNU gcc y g++ y conocer sus opciones.
 - ✓ Aprender a generar ficheros para la construcción de software con make.
 - ✓ Construir ficheros de bibliotecas.
 - ✓ Depurar programas con GNU gdb.
-

CONTENIDOS

Conceptos

- ✓ Editores de texto.
- ✓ El compilador gcc. Las bibliotecas.
- ✓ GNU make.
- ✓ GNU gdb.

Procedimientos

- ✓ Manejo e interpretación de los manuales, y del material bibliográfico.
- ✓ Utilización de los recursos del sistema. Editores, compilador, bibliotecas, construcción de ejecutables, depuradores, etc.
- ✓ Edición de un programa, con alguno de los editores incluidos con Linux, a partir de su listado fuente.
- ✓ Utilización del compilador de C gcc y de C++ g++.
- ✓ Construcción de Makefiles con make.
- ✓ Realización de pruebas.
- ✓ Corrección de los errores observados, con gdb.
- ✓ Documentación del programa, utilizando el correspondiente editor.

CRITERIOS DE EVALUACIÓN

- ✓ Citar las características fundamentales en que se basa la programación bajo Linux.
- ✓ Justificar la elección de un determinado entorno de programación.
- ✓ Describir el entorno de desarrollo del lenguaje, los recursos que se utilizan y el procedimiento práctico de desarrollo de programas.

- ✓ Describir la utilidad de las bibliotecas y de los enlazadores de los sistemas operativos y depuradores, así como su forma de empleo.
- ✓ Utilizar las bibliotecas y los enlazadores de los sistemas operativos así como los depuradores.
- ✓ Reconocer las diferencias entre la programación bajo MS-DOS y la programación bajo Linux.

UNIDAD 7. GESTIÓN DE FICHEROS CON C y C++

OBJETIVOS

- ✓ Identificar las distintas formas en que se pueden realizar las entradas/salidas de datos sobre ficheros.
 - ✓ Conocer y utilizar las funciones de la biblioteca estándar de C y C++.
 - ✓ Identificar las llamadas al sistema para la gestión de ficheros.
 - ✓ Utilizar las llamadas al sistema para las operaciones de entrada/salida sobre ficheros.
 - ✓ Aprender a realizar operaciones sobre directorios.
 - ✓ Realizar operaciones sobre dispositivos.
 - ✓ Aplicar llamadas al sistema a la administración del sistema de ficheros.
-

CONTENIDOS

Conceptos

- ✓ Conceptos de entrada/salida.
- ✓ Biblioteca estándar de entrada/salida.
- ✓ Primitivas de entrada/salida o funciones de entrada/salida de bajo nivel.
- ✓ Directorios.
- ✓ Atributos de los ficheros.
- ✓ Ficheros especiales. Dispositivos.
- ✓ Sistemas de ficheros.

Procedimientos

- ✓ Manejo e interpretación de los manuales, y del material bibliográfico.
- ✓ Utilización de los recursos del sistema.
- ✓ Interpretación del problema propuesto de aplicación y utilización básica de la interfaz.
- ✓ Elección de las llamadas al sistema necesarias para la resolución del problema.
- ✓ Construcción del algoritmo.
- ✓ Utilización de las correspondientes llamadas al sistema.
- ✓ Obtención de un programa ejecutable.
- ✓ Realización de pruebas.
- ✓ Corrección de los errores observados.

- ✓ Documentación del programa.
- ✓ Documentación de los cambios efectuados en el sistema.

CRITERIOS DE EVALUACIÓN

- ✓ Clasificar las instrucciones típicas de los lenguajes estructurados según su función.
- ✓ Describir las características de la interfaz de las llamadas al sistema.
- ✓ Explicar y utilizar los modelos de interfaz de programación que ofrecen los sistemas y su procedimiento de aplicación desde un programa.
- ✓ Identificar y utilizar funciones o servicios de llamada al sistema.
- ✓ Diseñar y codificar programas que pongan en evidencia el uso adecuado de los recursos de los lenguajes C y C++.
- ✓ Creación de aplicaciones, siguiendo las distintas fases del diseño.
- ✓ Utilizar correctamente las distintas utilidades integradas en Linux.
- ✓ Documentar el diseño y las opciones elegidas para la aplicación.

UNIDAD 8. EJECUCIÓN DE PROCESOS

OBJETIVOS

- ✓ Identificar los distintos elementos de un proceso.
 - ✓ Utilizar las funciones de la biblioteca estándar de C.
 - ✓ Identificar procesos.
 - ✓ Crear procesos.
 - ✓ Terminar procesos.
 - ✓ Controlar la ejecución y terminación de un proceso.
 - ✓ Utilizar las llamadas al sistema para crear funciones de usuario.
-

CONTENIDOS

Conceptos

- ✓ Conceptos previos sobre procesos.
- ✓ Ejecución simple de programas.
- ✓ Creación de un nuevo proceso.
- ✓ Ejecutando un programa.
- ✓ Terminación de procesos.
- ✓ Atributos de un proceso.
- ✓ Recursos utilizados por un proceso.

Procedimientos

- ✓ Interpretación del problema propuesto de aplicación y utilización básica de la interfaz.
- ✓ Elección de los objetos de programación necesarios para la resolución del problema.
- ✓ Construcción del algoritmo.
- ✓ Utilización de las correspondientes llamadas al sistema.
- ✓ Obtención de un programa ejecutable:
 - Codificación del algoritmo.
 - Compilación del programa fuente.
 - Montaje (enlazado) del programa objeto y las bibliotecas necesarias.
- ✓ Realización de pruebas, y corrección de los errores observados.
- ✓ Elaboración de la documentación del programa.

- ✓ Documentación de los cambios efectuados en el sistema.

CRITERIOS DE EVALUACIÓN

- ✓ Describir el funcionamiento de los procesos.
- ✓ Distinguir los distintos estados de un proceso.
- ✓ Identificar y utilizar funciones o servicios de llamada al sistema para la creación, gestión, control y parada de procesos.
- ✓ Describir las características de la interfaz de las llamadas al sistema.
- ✓ Justificar la necesidad de la prueba sistemática de los cambios introducidos en un sistema y sus aplicaciones.
- ✓ Comprobar que la utilización de recursos del sistema (procesador, memoria, periféricos) permiten que la integración y el enlace de programas sea ejecutable.
- ✓ Documentar los programas y las funciones realizados.
- ✓ Realizar pruebas sistemáticas sobre los programas desarrollados.

UNIDAD 9. SEÑALES

OBJETIVOS

- ✓ Comprender los mecanismos de comunicación basados en señales.
 - ✓ Conocer todos los tipos de señales definidos en Linux.
 - ✓ Aprender a definir gestores de señales en un programa.
 - ✓ Aprender a enviar señales desde un proceso a otro.
 - ✓ Desarrollar aplicaciones que estén sincronizadas temporalmente con el kernel del sistema operativo.
 - ✓ Desarrollar programas que requieran de una ejecución en paralelo sincronizada.
-

CONTENIDOS

Conceptos

- ✓ Diseño de aplicaciones que usan señales.
- ✓ Generación de una aplicación básica que captura señales del sistema operativo.
- ✓ Análisis de las distintas posibilidades de comunicación mediante señales.
- ✓ Manejo de las señales en una aplicación.
- ✓ Utilización de máscaras.
- ✓ Comunicación entre programas mediante señales.
- ✓ Fases de desarrollo de una aplicación en Linux que utilice señales.

Procedimientos

- ✓ Manejo e interpretación de los manuales y del material bibliográfico y la ayuda de Linux.
- ✓ Identificación de las distintas señales que se pueden producir en Linux.
- ✓ Identificación de los distintos métodos para trabajar con señales que intervienen en el diseño de una aplicación que utiliza señales.
- ✓ Interpretación de las distintas estructuras y funciones que son necesarias en un programa que utiliza señales.
- ✓ Interpretación de las distintas estructuras y funciones que son necesarias en un programa que utiliza señales en Linux.

CRITERIOS DE EVALUACIÓN

- ✓ Describir todos los tipos de señales fundamentales.
- ✓ Citar las opciones que se deben utilizar para gestionar las señales de una u otra manera.

- ✓ Justificar la elección de qué señales gestionar y del método empleado.
- ✓ Especificar correctamente qué máscara es necesaria y cómo se realizaría su implementación.
- ✓ Desarrollar de forma correcta un gestor de señales para la máscara que se ha implementado.
- ✓ Generar aplicaciones que contengan distintos gestores de señales, permitiendo la gestión de varias señales.
- ✓ Diseño de máscaras y modificaciones de las mismas para que la gestión de señales se adapte a distintas partes del programa.
- ✓ Realizar pruebas básicas de la aplicación.
- ✓ Desarrollar programas que se comuniquen mediante señales.

UNIDAD 10. COMUNICACIÓN ENTRE PROCESOS

OBJETIVOS

- ✓ Comprender los mecanismos de comunicación basados en tuberías, memoria compartida y colas de mensajes.
 - ✓ Conocer los distintos tipos de pipes y cómo utilizarlos.
 - ✓ Aprender a definir estructuras para determinar el procedimiento de comunicación.
 - ✓ Aprender a enviar datos entre dos aplicaciones.
 - ✓ Desarrollar aplicaciones que se transmitan información de forma asíncrona.
 - ✓ Desarrollar programas que requieran de una ejecución en paralelo compartiendo ciertos datos o donde los resultados de uno son necesarios para el otro.
-

CONTENIDOS

Conceptos

- ✓ Diseño de aplicaciones mediante pipes.
- ✓ Generación de una aplicación básica con pipes.
- ✓ Posibilidades de comunicación con pipes.
- ✓ Diseño y generación de aplicaciones mediante colas de mensaje y semáforos.
- ✓ Diseño y generación de una aplicación básica con memoria compartida.

Procedimientos

- ✓ Manejo e interpretación de los manuales y del material bibliográfico y de la ayuda de Linux.
- ✓ Identificación de los distintos tipos de procedimientos de comunicación.
- ✓ Identificación de las distintas estructuras que intervienen en el diseño de aplicaciones que hacen uso de pipes, mensajes y memoria compartida.
- ✓ Interpretación de las distintas clases de comunicación y en qué casos debe utilizarse cada una de ellas.

CRITERIOS DE EVALUACIÓN

- ✓ Descripción de cada una de las posibilidades de comunicación presentadas en el capítulo, analizando sus capacidades para transmisión de información.
- ✓ Citar las opciones que deben utilizar los distintos tipos de comunicación que se han presentado.
- ✓ Justificar la elección del tipo en función de la información que es necesaria transmitir y la complejidad del método de comunicación elegido.

- ✓ Integrar correctamente el método de comunicación elegido en la aplicación. Analizar la carga que debe soportar el programador a partir de la especificación de las estructuras y las funciones.
- ✓ Utilizar correctamente las distintas estructuras y funciones.
- ✓ Generar aplicaciones básicas con distintos tipos de comunicación.
- ✓ Documentar el diseño y las opciones elegidas para la aplicación.
- ✓ Realizar pruebas básicas de la aplicación.

UNIDAD 11. APLICACIONES CLIENTE/SERVIDOR MEDIANTE SOCKETS

OBJETIVOS

- ✓ Conocer las funciones estándar para trabajar con *sockets*.
 - ✓ Diseñar distintos sistemas de conexión entre aplicaciones.
 - ✓ Conocer las diferencias entre un servidor y un cliente.
 - ✓ Diseñar protocolos a nivel de aplicación.
 - ✓ Desarrollar servidores que proporcionen un servicio a un conjunto de clientes
 - ✓ Desarrollar un conjunto de clases en C++ que permita solucionar los problemas de desarrollo con *sockets*.
-

CONTENIDOS

Conceptos

- ✓ Diseño de aplicaciones utilizando *sockets*.
- ✓ Generación de un cliente básico mediante *sockets*.
- ✓ Manejo de los *sockets* para desarrollar aplicaciones que se conectan mediante *sockets*.
- ✓ Manejo de los *sockets* en desarrollo de servidores.
- ✓ Desarrollo de aplicaciones complejas que permiten el funcionamiento de aplicaciones altamente comunicadas.
- ✓ Fases de desarrollo de una aplicación que utiliza *sockets*.

Procedimientos

- ✓ Manejo e interpretación de los manuales y del material bibliográfico y de la ayuda de Linux.
- ✓ Identificación de los distintos componentes de un *socket*.
- ✓ Identificación de las distintas estructuras que intervienen en el diseño de una aplicación utilizando *sockets*.
- ✓ Interpretación de los distintos tipos de programas que pueden desarrollarse mediante *sockets*.

CRITERIOS DE EVALUACIÓN

- ✓ Describir de la funcionalidad de la arquitectura cliente/servidor.
- ✓ Describir los tipos de aplicaciones que pueden hacer uso de la arquitectura cliente/servidor así como del proceso de comunicación que siguen.
- ✓ Citar las posibilidades de diseño que tiene el programador a la hora de comunicarse mediante *sockets*.

- ✓ Justificar la elección de un tipo de cliente y de servidor para la generación del esqueleto base de una aplicación distribuida.
- ✓ Integrar correctamente las distintas posibilidades de comunicación.
- ✓ Generar aplicaciones con distintos tipos de servidor y cliente.
- ✓ Documentar el diseño de la aplicación.
- ✓ Realizar pruebas básicas de la aplicación.

TEMPORALIZACIÓN

PRIMERA EVALUACIÓN: Unidades 1, 2, 3, 4, 5, 6 y 7.

SEGUNDA EVALUACIÓN: Unidades 8, 9, 10 y 11.

CRITERIOS DE EVALUACIÓN

La evaluación se llevará a cabo mediante los siguientes recursos y con el baremo indicado:

- Pruebas escritas (50%). Se realizarán exámenes periódicos en los que el alumno deberá desarrollar pequeñas aplicaciones en lenguaje de programación C y C++. Las pruebas se realizarán escribiendo los programas en papel, o probando los resultados en un ordenador mediante entorno de desarrollo y sistema operativo Linux.
- Control de asistencia y actitud (10%)
- Control del trabajo realizado durante las clases (40%). A lo largo del curso, se desarrollarán prácticas relacionadas el temario. Durante las clases se darán las indicaciones básicas para su desarrollo, las prácticas se idearán para poder realizarlas en clase.

NOTA: Para realizar la media de cada evaluación es necesario aprobar cada uno de los exámenes de la parte escrita.

BIBLIOGRAFÍA

- ✓ SANCHIS DE MIGUEL, Araceli. *Desarrollo de funciones en el sistema informático con LINUX*. Editorial: McGraw-Hill
- ✓ Apuntes descargados de Internet.

ACTIVIDADES COMPLEMENTARIAS Y EXTRAESCOLARES.

- ✓ Sería conveniente, si existiera la posibilidad, visitar las instalaciones de alguna empresa de desarrollo de software o de cualquier empresa que tenga un sistema informatizado y exista la figura del Administrador del sistema.

MATERIALES Y RECURSOS DIDÁCTICOS.

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

HARDWARE:

- ✓ Un servidor Pentium IV.
- ✓ Veinte estaciones de trabajo Pentium IV conectadas en red.
- ✓ Una impresora láser y un escáner de página completa.
- ✓ Un switch de 24 puertos.
- ✓ Un sistema de proyección (proyector SVGA y pantalla).
- ✓ Conexión a Internet ADSL.

SOFTWARE:

- ✓ Sistemas operativos en red: Windows XP Professional, Linux Fedora Core 5, Knoppix 3.7.

MEDIDAS DE ATENCIÓN A LA DIVERSIDAD

Respecto a la atención a la diversidad se considera que será necesario realizar un esfuerzo extra por parte del profesor para que el proceso de enseñanza aprendizaje sea el adecuado en todos los casos ya que la procedencia del alumnado es bastante heterogénea. Así, hay alumnos con una buena base de conocimientos informáticos mientras que otros carecen de dicha base. Por tanto, habrá que facilitar recursos y materiales de apoyo, se deberá de prestar atención individualizada repitiendo los aspectos tratados en clase, será necesario proponer actividades alternativas que hagan que los alumnos menos motivados descubran alicientes en su formación, etc.