ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

FACULTAD DE INGENIERÍA EN ELECTRICIDAD Y COMPUTACIÓN

CONTENIDO DE CURSO

PROGRAMACIÓN DE SISTEMAS

CCPG1008

A. IDIOMA DE ELABORACIÓN

Español

B. DESCRIPCIÓN DEL CURSO

El curso aborda el desarrollo de software de bajo nivel para interactuar directamente con el sistema operativo de una computadora o con hardware. Adicionalmente, se cubre el uso de herramientas que permiten gestionar la versión y compilación del código desarrollado para facilitar el trabajo en equipo y contribuir en proyectos de software. En el curso se utiliza “scripting” de BASH para automatizar tareas y el lenguaje C para interactuar directamente con el kernel en sistemas operativos UNIX / LINUX.

C. CONOCIMIENTOS PREVIOS DEL CURSO

Conocimientos intermedios de programación y uso del computador.

D. OBJETIVO GENERAL

Desarrollar software de bajo nivel computacional usando el lenguaje C y herramientas de gestión de código fuente para la interacción directa y eficaz con el sistema operativo y el hardware en sistemas basados en UNIX / LINUX.

E. OBJETIVOS DE APRENDIZAJE DEL CURSO

El estudiante al finalizar el curso estará en capacidad de:

1	Construir un programa simple en C usando métodos de división de capas, detección de errores y reflexión de estatus de errores para la creación de un sistema robusto y de mínimo mantenimiento.
2	Implementar programas con paralelismo computacional usando eventos, hilos, procesos y otros paradigmas de concurrencia para el uso eficiente de los recursos provistos por el hardware y el sistema operativo de un computador.
3	Implementar una aplicación cliente-servidor simple usando sockets y una interfaz de programación de aplicaciones (API) básico para la creación de un sistema escalable con clara separación de competencias.
4	Programar un sistema computacional usando un paradigma orientado a eventos para la gestión de eventos asincrónicos externos.
5	Usar herramientas de colaboración de software, depuración e integración para la gestión en equipo del desarrollo de productos de software de mediano tamaño.

F. ESTRATEGIAS DE APRENDIZAJE

Aprendizaje asistido por el profesor	✔
Aprendizaje cooperativo/colaborativo:	✔
Aprendizaje de prácticas de aplicación y experimentación:	✔
Aprendizaje autónomo:	✔

G. EVALUACIÓN DEL CURSO

Actividades de Evaluación	DIAGNÓSTICA	FORMATIVA	SUMATIVA
Exámenes			✔
Lecciones		✔	✔
Tareas			✔
Proyectos		✔	✔
Laboratorio/Experimental		✔	✔
Participación en Clase	✔	✔	
Visitas			
Otras			

H. PROGRAMA DEL CURSO

UNIDADES y SUBUNIDADES	Horas Docencia
1. Introducción a la consola (shell) de Linux y C	6
1.1. Introducción al curso y a la arquitectura de sistemas UNIX / LINUX	
1.2. Comandos básicos del shell de LINUX	
1.3. Scripts en el shell de LINUX	
1.4. Introducción al lenguaje C	
2. Compiladores y herramientas de programación	8
2.1. Introducción a las cadenas de desarrollo	
2.2. Creación de archivos Makefile	
2.3. Introducción a entornos de desarrollo integrado (IDE)	
2.4. Introducción a herramientas de versionamiento de código	
3. Representación de datos y gestión de memoria	9
3.1. Representación de tipos de datos en memoria	
3.2. Punteros	
3.3. Gestión de la memoria	
3.4. Depuración de errores	
4. Librerías	4
4.1. Librerías estáticas	
4.2. Librerías dinámicas	
4.3. Compilación de librerías	
4.4. Linking de librerías	
5. Entrada/Salida	9
5.1. Introducción a redes de computadoras	
5.2. Introducción a prótocolos de transferencia de control e internet (TCP/IP)	
5.3. API entrada/salida	
5.4. API sockets	
6. Programación en paralelo	12
6.1. Uso de señales asincrónicas	
6.2. Concurrencia con procesos	
6.3. Concurrencia con hilos	
6.4. Variables compartidas y sincronización con hilos	
6.5. Patrones de diseño para concurrencia	

I. REFERENCIAS BIBLIOGRÁFICAS

BÁSICA	1. Randal Bryant y David O'Hallaron. (2015). Computer Systems: A Programmer's Perspective. (3ra). Massachusetts, USA: Pearson. ISBN-10: 013409266X
COMPLEMENTARIA	1. Love, R. (2013). Linux System Programming: Talking Directly to the Kernel and C Library. (2da). California, USA: O'Reilly. ISBN-10: 1449339530, ISBN-13: 9781449339531

J. DESCRIPCIÓN DE UNIDADES

1. *Introducción a la consola (shell) de Linux y C*

Introducción a la unidad

En esta unidad se revisan los conceptos básicos de sistemas operativos UNIX / LINUX, del funcionamiento del interpretador de comandos de estos sistemas (conocido como shell) y del lenguaje C.

Meta-Lenguaje

shell, bash, scripts, C, LINUX

Subunidades

1.1.	Introducción al curso y a la arquitectura de sistemas UNIX / LINUX
1.2.	Comandos básicos del shell de LINUX
1.3.	Scripts en el shell de LINUX
1.4.	Introducción al lenguaje C

Objetivos de Aprendizaje

1.1.	Codificar un programa de complejidad media usando shell-scripts para automatizar tareas en sistemas operativos UNIX/LINUX.
1.2.	Codificar un programa simple usando el lenguaje C.

Actividades

1.1. Ejercicios de codificación de shell-scripts

Los estudiantes programarán scripts de shell para automatizar tareas simples en un sistema operativo.

1.2. Clase Magistral

Introducción al curso, UNIX, LINUX y el shell de LINUX.

1.3. Lección

Control de lectura.

1.4. Lectura

Lectura de secciones del libro guía sobre arquitectura del computador.

1.5. Ejercicio de programación

Resolver un problema computacional simple usando el lenguaje C.

2. *Compiladores y herramientas de programación*

Introducción a la unidad

En esta unidad se cubre la cadena de herramientas de C y C++, desde los entornos de desarrollo integrado (IDE) más usados y los compiladores hasta las herramientas de depuración y versionamiento.

Meta-Lenguaje

c, c++, versionamiento, compilador, depurador

Subunidades

2.1.	Introducción a las cadenas de desarrollo
2.2.	Creación de archivos Makefile
2.3.	Introducción a entornos de desarrollo integrado (IDE)

J. DESCRIPCIÓN DE UNIDADES

2.4. Introducción a herramientas de versionamiento de código

Objetivos de Aprendizaje

2.1. Desarrollar un programa simple en C usando herramientas GNU de compilación y gestión de código en sistemas UNIX / LINUX.
2.2. Gestionar el desarrollo de un proyecto de software usando herramientas de versionamiento de código para la facilitación del trabajo en equipo.

Actividades

2.1. Ejercicios de desarrollo de programas en C usando Make, GCC

Trabajo autónomo y práctico de codificación de un programa usando C en conjunto con herramientas de la cadena GNU como Make y GCC (GNU Compiler Collection). Además, el estudiante deberá usar una herramienta de versionamiento como Git para gestionar el desarrollo en equipo y vim/nano como IDE de desarrollo.

2.2. Clase Magistral

Introducción a la cadena de desarrollo de C/C++ en Linux y a la herramienta de versionamiento Git.

2.3. Lección

Control de lectura.

2.4. Lectura

Lectura de secciones del libro guía sobre los compiladores existentes de C/C++.

3. *Representación de datos y gestión de memoria*

Introducción a la unidad

En esta unidad se revisan los conceptos de gestión de memoria en el lenguaje C desde el uso básico de punteros hasta el uso de la herramienta GNU Debugger (GDB) para depuración avanzada de código fuente. El estudiante aprende los errores comunes en la gestión de memoria y técnicas para depurarlos.

Meta-Lenguaje

memoria, punteros, heap, stack, GNU Debugger

Subunidades

3.1. Representación de tipos de datos en memoria
3.2. Punteros
3.3. Gestión de la memoria
3.4. Depuración de errores

Objetivos de Aprendizaje

3.1. Gestionar eficientemente el uso de la memoria mediante el uso de punteros y GDB en un programa en lenguaje C.

J. DESCRIPCIÓN DE UNIDADES

Actividades

3.1. Ejercicios de programación usando punteros y memoria dinámica

Trabajos autonómos y en equipo en el cual los estudiantes codifican estructuras de datos con memoria dinámica como listas enlazadas o tablas de hash para resolver problemas computacionales básicos.

3.2. Clase Magistral

Estudio de gestión de memoria, así como de los errores comunes de gestion de memoria. Revisión básica de respresentación de datos.

3.3. Lección

Control de lectura.

3.4. Lectura

Lectura de secciones del libro guía y material complementario sobre gestión de memoria.

4. *Librerías*

Introducción a la unidad

En esta unidad se revisa el uso y creación de librerías estáticas y dinámicas en el lenguaje C. Se enfatiza además el uso de librerías como una forma para distribuir y compartir soluciones computacionales a terceros.

Meta-Lenguaje

librería estática, librería dinámica, linking

Subunidades

4.1. Librerías estáticas
4.2. Librerías dinámicas
4.3. Compilación de librerías
4.4. Linking de librerías

Objetivos de Aprendizaje

4.1. Crear una librería usando el lenguaje C y Make para el empaquetamiento y distribución de algoritmos computacionales.

Actividades

4.1. Ejercicio de creación de una libería

Trabajo autónomo y en equipo donde el estudiante debe de crear una librería en versión dinámica y estática para empaquetar alguna funcionalidad específica.

4.2. Clase Magistral

Creación de librerías estáticas y dinámicas usando C y Make. Ventajas y desventajas de los diferentes tipos de librerías.

4.3. Lección

Control de lectura.

4.4. Lectura

Lectura de secciones del libro sobre el uso de librerías dinámicas y estáticas.

5. *Entrada/Salida*

Introducción a la unidad

J. DESCRIPCIÓN DE UNIDADES

En esta unidad se revisa la arquitectura cliente - servidor como un patrón de diseño básico para aplicaciones distribuidas. Se utilizan llamadas del sistema de entrada/salida para comunicación entre procesos usando sockets y archivos, creando un sistema distribuido.

Meta-Lenguaje

archivos, sockets, sistema distribuido, Input/Output (I/O)

Subunidades

5.1. Introducción a redes de computadoras
5.2. Introducción a prótocolos de transferencia de control e internet (TCP/IP)
5.3. API entrada/salida
5.4. API sockets

Objetivos de Aprendizaje

5.1. Desarrollar una aplicación distribuida con modelo cliente - servidor usando APIs de entrada/salida y sockets.

Actividades

5.1. Ejercicios de programación usando modelo cliente - servidor

Trabajo autónomo y en equipo donde el estudiante desarrolla aplicaciones distribuidas usando comunicación entre procesos con sockets.

5.2. Lección

Control de lectura.

5.3. Charla Magistral

Revisión básica de las redes de computadoras y TCP/IP y estudio de los APIs de entrada/salida.

5.4. Lectura y discusión de literatura

Lectura de secciones del libro guía sobre TCP/IP.

6. *Programación en paralelo*

Introducción a la unidad

En esta unidad se revisan los conceptos de programación que usan concurrencia. Específicamente se exploran tres paradigmas de concurrencia a bajo nivel computacional: eventos, procesos e hilos. Además se utlizan patrones de diseño que permiten explotar el paralelismo computacional de una manera robusta y eficiente.

Meta-Lenguaje

hilos, concurrencia, paralelismo, procesos, eventos, sincronización, semáforos

Subunidades

6.1. Uso de señales asincrónicas
6.2. Concurrencia con procesos
6.3. Concurrencia con hilos
6.4. Variables compartidas y sincronización con hilos
6.5. Patrones de diseño para concurrencia

Objetivos de Aprendizaje

6.1. Implementar un programa robusto de complejidad baja usando paradigmas de concurrencia para explotar eficientemente los recursos del sistema.

J. DESCRIPCIÓN DE UNIDADES

Actividades

6.1. Lectura y discusión de literatura

Lectura y discusión en grupo de capítulos del libro guía y literatura que trate sobre diversos temas de concurrencia.

6.2. Lección

Control de lectura.

6.3. Charla Magistral

Estudio de paradigmas de concurrencia basados en procesos, hilos y señales, así como temas de sincronización.

6.4. Ejercicios de programación usando varios modelos de concurrencia

Trabajos de programación usando los diversos paradigmas de concurrencia.

K. RESPONSABLES DE LA ELABORACIÓN DEL CONTENIDO DE CURSO

Profesor	Correo	Participación
DOMINGUEZ BONINI FEDERICO XAVIER	fexadomi@espol.edu.ec	Coordinador de materia
MURILLO BAJAÑA EDUARDO WENCESLAO	emurillo@espol.edu.ec	Colaborador