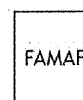


Universidad  
Nacional  
de Córdoba



**FAMAF**  
Facultad de Matemática,  
Astronomía y Física

EXP-UNC: 49517/2015

Resolución CD N° 361/2015

**PROGRAMA DE ASIGNATURA**

<b>ASIGNATURA:</b> Sistemas Operativos		<b>AÑO:</b> 2015
<b>CARÁCTER:</b> Obligatoria		
<b>CARRERA:</b> Licenciatura en Ciencias de la Computación		
<b>RÉGIMEN:</b> Cuatrimestral	<b>CARGA HORARIA:</b> 120 hs.	
<b>UBICACIÓN en la CARRERA:</b> 2do Año - 2do Cuatrimestre		

**FUNDAMENTACIÓN Y OBJETIVOS**

**Fundamentación:**

El sistema operativo es un programa fundamental dentro de toda la pila de software y hardware que compone una computadora moderna. Esto es así no solamente porque asila a los programas de usuario de los detalles del hardware subyacente, sino que además provee fuertes abstracciones que han perdurado a lo largo de las décadas: procesos, memoria (virtual) y sistema de archivos.

La necesidad de aprovechar mejor el hardware hizo que apareciera el concepto de multiprogramación, donde el no-determinismo de los programas secuenciales se presenta de manera concreta, además de presentar el Área de la Teoría y la Práctica de la Concurrencia.

**Objetivos:**

**Teórico**

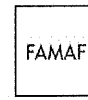
- Comprender las abstracciones principales de un Sistema Operativo: procesos, memoria, sistema de archivos.
- Resolver problemas simples que se plantean en la práctica para estas abstracciones.
- Comprender, reparar y programar algoritmos concurrentes de baja complejidad.
- Entender la problemática de la seguridad en general, y para los Sistemas Operativos en particular.
- Resolver problemas sencillos que involucren algunos de los aspectos sobresalientes de la seguridad y la entrada/salida en sistemas operativos.
- Entender las relaciones de compromiso de los algoritmos y estructuras de datos internas del Sistema Operativo. Comprender como algunos cambios tecnológicos afectan fuertemente estas relaciones de compromiso.
- Comprender la relación entre algunas partes del diseño de la arquitectura del microprocesador con el Sistema Operativo.
- Poder asimilar los conceptos utilizando ejemplos concretos de Sistemas Operativos.

**Laboratorio**

- Avanzar en la práctica de la programación en general.
- Trabajar en grupo tanto en objetivos individuales como en objetivos grupales.
- Ser capaz de leer, modificar y comprobar código dentro de Sistemas Operativos completos y funcionales.
- Utilizar herramientas de apoyo para el desarrollo del software: editores, detectores de errores en código estático, debuggers, chequeadores de memoria, etc.
- Utilizar herramientas de desarrollo colaborativo de proyectos.
- Generar independencia para la búsqueda de soluciones técnicas en el proceso de desarrollo y/o modificación de código.
- Realizar entregas de proyectos dentro de límites de tiempo prefijados.
- Programar abstracciones de dispositivos de bajo nivel a partir de la especificación dada por su hoja de datos.



Universidad  
Nacional  
de Córdoba



**FAMAF**  
Facultad de Matemática,  
Astronomía y Física

EXP-UNC: 49517/2015

Resolución CD N° 361/2015

- Realizar modificaciones a partes fundamentales del Sistema Operativo: procesos, memoria virtual y sistema de archivos.
- Comprender en general la problemática del desarrollo del software dentro del núcleo del Sistema Operativo.

## CONTENIDO

### Unidad I: Introducción.

- (1) Historia de la Computación.
- (2) ¿Qué es un sistema operativo?
- (3) Historia de los sistemas operativos.
- (4) Variedad de sistemas operativos: de tiempo real, embebidos, distribuidos.
- (5) Conceptos de sistemas operativos.
- (6) Llamadas a sistemas.
- (7) Estructura de los sistemas operativos.

### Unidad II: Procesos e Hilos.

- (1) Concepto de proceso.
- (2) Hilos.
- (3) Comunicación entre procesos.
- (4) Problemas clásicos de la comunicación entre procesos.
- (5) Planificación de procesos (Scheduling).

### Unidad III: Concurrencia de ejecución. Interbloqueos (deadlocks).

- (1) Introducción a deadlocks y recursos.
- (2) Detección y recuperación de deadlocks.
- (3) Técnicas para evitar los deadlocks.
- (4) Técnicas para prevenir los deadlocks.

### Unidad IV: Sistema de Archivos:

- (1) Archivos.
- (2) Directorios.
- (3) Protección.
- (4) Implementación de los sistemas de archivos.
- (5) Ejemplos de sistemas de archivos.

### Unidad V: Administración de Memoria.

- (1) Administración básica de memoria.
- (2) Swapping.
- (3) Memoria Virtual.
- (4) Algoritmos para reemplazo de página.
- (5) Tópicos de diseño para sistemas de paginado.
- (6) Tópicos de implementación.
- (7) Segmentado.
- (8) Memoria compartida distribuida.

### Unidad VI: Seguridad.

- (1) Amenazas: Confidencialidad, Integridad y Disponibilidad.
- (2) Ataques desde fuera y dentro del sistema.
- (3) Mecanismos de protección.
- (4) Sistemas Confiables.
- (5) Intrusos.
- (6) Conceptos básicos de Criptografía.
- (7) Autenticación de usuarios.

### Unidad VII: Entrada/Salida.

- (1) Principios del hardware de entrada/salida.
- (2) Principios del software de entrada/salida.
- (3) Organización en capas del software de entrada/salida.
- (4) Discos.
- (5) Otros dispositivos.

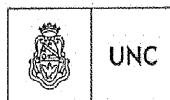
### Unidad VIII: Sistemas operativos de usos específicos.

- (1) Sistemas operativos para sistemas de tiempo real.
- (2) Planificación para sistemas de tiempo real.
- (3) Prioridades.
- (4) Sistemas operativos para sistemas embebidos.
- (5) Sistemas operativos para dispositivos móviles.

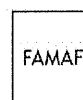
## BIBLIOGRAFIA

### BIBLIOGRAFÍA BÁSICA

- [1] Gunnar Wolf, Esteban Ruiz, Federico Bergero, Erwin Meza. Fundamentos de Sistemas



Universidad  
Nacional  
de Córdoba



**FAMAF**  
Facultad de Matemática,  
Astronomía y Física

EXP-UNC: 49517/2015

Resolución CD N° 361/2015

Operativos, 2015.

[2] Andrew S. Tanenbaum. Sistemas Operativos Modernos, Tercera Edición. Prentice Hall, 2009.

[3] Abraham Silberschatz. Operating System Concepts, Sixth Edition. John Wiley & Sons, 2001.

### **BIBLIOGRAFÍA COMPLEMENTARIA**

[4] Russ Cox , Frans Kaashoek , Robert Morris, xv6 a simple, Unix-like teaching operating system , MIT, draft 2012.

[5] Raphael Finkel. An operating systems Vade Mecum, Segunda Edición. Prentice Hall, 1988.

[6] Jonathan Corbet, Alessandro Rubini, and Greg Kroah-Hartman. Linux Device Drivers, Third Edition. O'Reilly, 2005.

[7] Michael Kerrisk, The Linux Programming Interface, No Starch Press, 2010.

### **METODOLOGÍA DE TRABAJO**

- Dictado de clases teóricas interactivas.
- Trabajo individual sobre:
  - Resolución de guías de estudios
  - Resolución de problemas teórico-prácticos
- Trabajo grupal (hasta 3 estudiantes) en el Laboratorio:
  - Realización de un (1) proyecto individual de modificación de código, cuatro (4) proyectos grupales de desarrollo en Laboratorio.
  - Seguimiento, corrección y asistencia continuas sobre estos proyectos por parte de los docentes.
- Uso del Aula Virtual para :
  - Hacer disponible electrónicamente toda la información de la materia incluyendo material didáctico.
  - Uso activo de Foros de Discusión para información y consulta de dudas a los docentes y entre pares.

### **EVALUACIÓN**

#### **FORMAS DE EVALUACIÓN**

- Dos parciales teórico-práctico.
- Un (1) proyecto de Laboratorio individual, cuatro (4) proyectos grupales. Todos los proyectos permiten re-entrega mientras sea dentro del tiempo de cursado.
- Régimen de promoción.
- Examen final teórico-práctico, más coloquio sobre los Laboratorios si el alumno está libre.

#### **CONDICIONES PARA OBTENER LA REGULARIDAD**

Aprobación de los cinco (5) proyectos de laboratorio (100%), donde en todos se permite re-entrega a lo largo del cursado.

#### **CONDICIONES PARA OBTENER LA PROMOCIÓN**

Para promocionar la materia se necesita:

- Aprobar los dos parciales con un promedio de 8 (cada parcial con más de 6).
- Aprobar todos los laboratorios con 6 o más.

En caso de promocionar la calificación final será:  $0.4 * p1 + 0.4 * p2 + 0.2 * lab$  (donde  $p1$  y  $p2$  son las calificaciones del primer y segundo parcial respectivamente, y  $lab$  es la calificación del desempeño en los laboratorios).