



UNIVERSIDAD NACIONAL DE LA MATANZA
Departamento de Ingeniería
e Investigaciones Tecnológicas

Cátedras de:

Sistemas de Computación II (Plan 1997)

Sistemas Operativos (Plan 2009)

Jefe de Cátedra: Fabio E. Rivalta
Equipo de Docentes: Boettner F., Catalano L.,
de Lizarralde R, Villamayor A
Auxiliares docentes: Loiacono F., Hirschfeldt D.,
Rodriguez A., Piubel F., Segura L.,

PLANIFICACIÓN Y

GUÍA DE TRABAJOS PRÁCTICOS

(Primer cuatrimestre 2020)

Contenidos:

- Reglamento, Programa y Planificación.
- Comentarios Previos
- Introducción al Diseño de Software.
- Guía de Trabajos Prácticos Grupales.

DEPARTAMENTO: *Ingeniería e Investigaciones Tecnológicas***ASIGNATURAS:****Sistemas de Computación II
Sistemas Operativos****1. OBJETIVOS.****1.1. OBJETIVOS DEL CURSO.**

- Brindar los conceptos fundamentales y su respectiva actualización tecnológica sobre los Sistemas Operativos.
- Facilitar una actualización sobre las terminologías, y desarrollos tecnológicos sobre Sistemas Operativos Modernos.

1.2. OBJETIVOS DE APRENDIZAJE.

- Que el alumno adquiriera el dominio de conceptos básicos y actualizados sobre los Sistemas Operativos e introducir los lineamientos generales de nuevos desarrollos tecnológicos en estos temas.
- Generar una concepción global y un enfoque selectivo para las soluciones algorítmicas de los diferentes problemas que ocurren dentro de un computador y la correcta utilización del mismo.

1.3. META OPERATIVA:

- Se tratara que el alumno al finalizar la materia logre:
- Adquirir el vocabulario y usarlo con precisión.
- Conocer en forma amplia y general la misión y funcionamiento de los componentes de los Sistemas Operativos de un computador.
- Analizar y evaluar por sí mismo un Sistema Operativo de cualquier equipo existente en plaza.
- Desarrollar en el Alumno, el interés por la investigación, usando libros y publicaciones propuestas por el Docente.
- Crear en el Alumno, una capacidad de resolución de problemas mediante una adecuada ejercitación práctica.
- Motivar en los alumnos a proponer algunos temas de interés para desarrollar o investigar o encontrar diferentes soluciones a los mismos.

2. ALCANCES.

Los temas a tratar contemplarán básicamente los módulos temáticos que propondrá la cátedra. Su profundidad abarca la extensión de todos los temas específicamente mencionados en el Programa de SISTEMAS DE COMPUTACIÓN II (Plan 1997) y SISTEMAS OPERATIVOS (Plan 2009) vigentes.

PROGRAMA ANALÍTICO.

CONTENIDOS TEÓRICOS Y PRÁCTICOS DE LAS ASIGNATURAS: SISTEMAS DE COMPUTACIÓN II (Plan 1997) y SISTEMAS OPERATIVOS (Plan 2009) – VERSIÓN 2020

Módulo 1: Generalidades de los sistemas operativos

Conceptos de Arquitecturas CISC y RISC. Conceptos de Microprogramación. Conceptos del Lenguaje Assembler.

Interacción con el Sistema Operativo. Limitaciones del Hardware de las Computadoras.

Introducción a los SO. Clasificación. Conceptos fundamentales y conceptos básicos de SO.

Terminología, definiciones y funciones de SO.

Características comunes a todos los SO. Organización y estructura interna de los sistemas operativos.

Componentes mínimos de un SO.: El shell, los administradores del SO., el Kernel o núcleo.

Prestaciones y servicios de los SO.

Módulo 2: Procesos

Definición y concepto de proceso.

Estados de un proceso. Diagrama de estados. Ciclo de vida de un proceso. Transiciones de estado.

Las operaciones sobre un proceso: creación, manipulación y muerte de un proceso.

El control de un proceso. Estructuras de control del sistema operativo.

Tipos de procesos: los procesos pesados y livianos, hilos o hebras (Threads).

Implementación de hilos (Threads). La creación y ejecución de los Threads. Estado de los threads. Uso de los hilos. Sistemas operativos "multithreaded": aspectos del diseño e implementación de paquetes de Threads. El Concepto de Fibra (Fiber). principios de multitareas

Módulo 3: Planificación de procesadores

Objetivos. Introducción al problema de la planificación: planificación de monoprocesadores y multiprocesadores.

Niveles de planificación: extra largo plazo, largo plazo, mediano plazo y a corto plazo.

Criterios de planificación de los trabajos y de los procesos: política vs. mecanismo.

Administración y gestión de procesos y procesadores: tipos de planificadores. Algoritmos de planificación del procesador.

Algoritmos NON-PREEMPTIVE (sin reemplazo o apropiativos): FCFS (First-Come First-Served), SPF-Shortest Process First (también llamado SPN-Shortest Process Next). Planificación por prioridad.

Algoritmos PREEMPTIVE (con reemplazo en el uso del procesador), Round Robin o torneo cíclico, Menor tiempo restante (SRT Shortest Remaining Time First). Primero el de mayor tasa de respuesta (HRRN).

Planificación con colas de múltiples niveles y realimentación. Planificación con múltiples colas fijas.

Planificación con múltiples colas dinámicas. Planificación de tres niveles. Evaluación y comparación de algoritmos.

Planificación de múltiples procesadores: granularidad, planificación de múltiples procesos y de hilos.

Ejemplos de "scheduler/dispatcher" de sistemas operativos.

Evaluación de desempeño. Detección de cuellos de botellas en el procesador.

Módulo 4: Sincronización y Comunicación entre Procesos

Conceptos de sincronización y comunicación entre procesos.

Problemas concurrentes. Grafos de precedencia. Condiciones de concurrencia (Bernstein).

Especificaciones concurrentes: Fork y Join, Cobegin y coend.

Relaciones entre procesos concurrentes y sus conflictos. Introducción al problema de la región crítica (RC.).

Condición de carrera. Solución de la concurrencia por software y hardware.

Algoritmos de sincronización con espera activa: solución simple, espera ocupada por turnos (alternancia), solución de Peterson, algoritmo de Dekker, algoritmo de Lamport o de la panadería.

Algoritmos sin espera activa: semáforos, monitores.

Mecanismos provistos por el hardware. Cola de espera, Semáforos.

Comunicaciones entre procesos: mensajes, IPC: Inter Process Communication, tipos de sincronizaciones mediante mensajes, modelo productor-consumidor, algunos algoritmos para el modelo productor-consumidor.

Deadlocks (interbloqueo, bloqueo mutuo o abrazo mortal). Condiciones necesarias y suficientes. Tipos de recursos.

Ejemplos de abrazo mortal. Prevención, detección, evasión y recuperación de abrazo mortal.

Métodos de representación: grafos y matrices. Grafo de asignación de recursos. Estrategias para tratar Deadlocks. Conflicto en la comunicación entre procesos

Módulo 5: Administración de Memoria Central

Administración de memoria central (MC). Funciones del administrador de la memoria central. Objetivos de la administración de la MC. Asignación y reasignación de direcciones. Espacio de direcciones lógico y físico.

Técnicas de administración sin swapping: Memoria dedicada (máquina desnuda sin SO.), Asignación contigua simple o monitor residente, asignación particionada simple y variable, paginación pura, segmentación simple, manejo de memoria con buddy system.

Técnicas de administración con swapping (intercambio) o sea memoria virtual: swapping, paginación por demanda o bajo solicitud.

Algoritmos de gestión de memoria virtual, sistemas mixtos: segmentación con paginación por demanda.

Módulo 6: Sistema de Gestión de Entrada / Salida

Administración de la Entrada / Salida (I/O scheduler). Funciones del administrador de E/S. Módulos de E/S y la estructura del módulo de E/S.

Función del módulo. Estructura del módulo de E/S.

Las operaciones del hardware de E/S: operación asincrónica, diferencias de velocidades. Los dispositivos y sus interfaces (el hardware de E/S): dispositivos de E/S. Controlador, adaptador o interface de E/S, procesadores de E/S (IOP), dispositivos externos, almacenamiento intermedio de E/S (Buffering), dispositivos internos.

Técnicas de E/S: E/S programada, E/S por interrupciones, E/S por DMA (Acceso Directo a Memoria).

Principios del software de E/S. Metas del software de E/S. Manejadores de interrupciones (Interrupt handler).

Drivers de dispositivos. Pasos y controles en una operación de E/S. Software de E/S independiente del dispositivo.

Software de E/S del espacio del usuario. Software de entrada. Software de salida. Procesadores de E/S y Canales de E/S

Módulo 7: Sistema de archivos y sus directorios

Introducción sistema de gestión de archivos (File System).

Concepto de archivo. Tipos de archivos. Atributos de los archivos.

Sistemas basados en cinta y en disco.

Objetivos y funciones del sistema de gestión de archivos. Conflictos. Sistema básico de archivos. La estructura de la información. Archivos mapeados a memoria. Nombres de archivos. La estructura de un archivo. Estructura interna. Descriptores de archivos.

Operaciones sobre archivos: apertura y cierre, creación, escritura, lectura, rebobinado y borrado.

Catalogación de los archivos en el soporte: Área de datos fijos, área de catálogo y área de datos.

Administración del espacio de almacenamiento: espacio libre, métodos de asignación. Sistemas de directorio: directorio de dispositivos. Operaciones sobre directorios. Estructuras de directorio.

Métodos de acceso: acceso secuencial, acceso directo, otros métodos de acceso. Métodos de implementación del sistema de archivos. Algoritmos para la administración de archivos.

Protección de archivos: nombre, contraseñas, control de acceso.

Módulo 8: Protección y seguridad

Concepto de seguridad y protección. Concepto de política y mecanismo. Política de seguridad. Principios de las políticas de seguridad. Categorías básicas de las políticas de seguridad. Objetivos de la seguridad y la protección de un sistema. Justificación de la seguridad y protección. Niveles de seguridad en informática.

Amenazas a la seguridad. Diseño: principio de los mecanismos. Tipos de seguridad. Supervisión y vigilancia.

Supervisión de riesgos de seguridad por el SO.

Seguridad a través del sistema operativo. Funciones de los sistemas de protección en el sistema operativo.

Seguridad en el kernel.

Dominios de protección: matriz de accesos. Implementación de la matriz de accesos. Cambio de dominio – switch.

Cambio de contenido de la matriz de accesos. Revocación de permisos. Sistemas basados en capacidades.

Seguridad multinivel, autenticación del usuario: validación. Los problemas de la identidad: sus puntos débiles.

Amenazas relacionadas con los programas: caballo de troya, puerta trasera, bomba lógica, desbordamiento de pila y de buffer, virus, gusanos, vulnerabilidad. Política de seguridad.

Seguridad para los datos. Seguridad de datos en bases de datos. Métodos de ocultamiento de los datos.

Algunos problemas en CRIPTOGRAFÍA.

Seguridad en telecomunicaciones o redes de computadoras. Distribución de llaves. Normas y procedimientos en un sistema de seguridad: estrategia de seguridad, plan de contingencia. Auditorías. Mecanismos y políticas de seguridad en sistemas.

Módulo 9: Sistemas distribuidos

Conceptos de sistemas cliente/servidor y sus variantes.

Conceptos de procesamiento distribuido.

Conceptos de sistemas de archivos en sistemas distribuidos.

Conceptos de control de concurrencia en sistemas distribuidos.

Conceptos de memoria compartida distribuida.

Conceptos sobre transacciones distribuidas

Módulo 10: Sistemas de alto rendimiento

Conceptos de procesadores de alta performance.

Conceptos de procesamiento paralelo.

Conceptos de arquitecturas multiprocesadores.

Generación y ajuste de un sistema operativo. Mediciones del sistema y performance.

BIBLIOGRAFÍA RECOMENDADA PARA EL CURSO (EN INGLÉS) 2020

OBRA: Operating Systems Internals and Design Principles (7th Edition)

AUTOR: Stallings, William

EDITORIAL: Prentice Hall

FECHA: 2011

OBRA: Operating Systems Concepts (9th edition)

AUTOR: Silberschatz, J.L. and Galvin P. B.

EDITORIAL: Addison Wesley

FECHA: 2012

BIBLIOGRAFÍA RECOMENDADA PARA CONSULTA (EN INGLÉS)

OBRA: UNIX Internals - A Practical Approach

AUTOR: Steve D Pate

EDITORIAL: Addison Wesley

FECHA: 1996

OBRA: Advanced programming the UNIX environment

AUTOR: Richard Stevens

EDITORIAL: Addison Wesley

FECHA: 2001

OBRA: UNIX network programming Volume 1

AUTOR: Richard Stevens

EDITORIAL: Prentice Hall

FECHA: 1998

BIBLIOGRAFÍA RECOMENDADA PARA CONSULTA (EN CASTELLANO)

OBRA: FUNDAMENTOS DE SISTEMAS OPERATIVOS

AUTOR: Gunnar Wolf, Esteban Ruiz, Federico Bergero y Erwin Meza

EDITORIAL: UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FECHA: 2015

OBSER.: Libro de referencia https://sistop.org/pdf/sistemas_operativos.pdf

OBRA: Notas sobre Sistemas Operativos - Manual del Alumno - 2 tomos

AUTOR: La Cátedra

EDITORIAL: Ghia

FECHA: 2006

OBSER.: Libro de referencia para el seguimiento de las clases

OBRA: Apuntes de Sistemas Operativos Distribuidos

AUTOR: La Cátedra

EDITORIAL: Ghia

FECHA: 2007

OBSER.: Libro de referencia para el seguimiento de las clases

OBRA: Sistemas Operativos Principios de diseño (Desde la Fifth Edition)

AUTOR: Stallings, William

EDITORIAL: Prentice Hall

FECHA: 2006

OBRA: Sistemas Distribuidos – Conceptos y Diseño (Desde la 3° Edition)

AUTOR: George Coulouris / Jean Dollimore / Tim Kindberg

EDITORIAL: Addison Wesley

FECHA: 2001

OBRA: Sistemas Operativos (Desde la 7ta. edición)

AUTOR: Silberschatz, Galvin & Gagne

EDITORIAL: Mc Graw Hill

FECHA: 2006

METODOLOGÍA DE ENSEÑANZA.

El dictado del curso será del tipo explicativo, participativo e informativo, basado en la discusión de los tópicos desarrollados en el transcurso de las diferentes clases mediante su tratamiento teórico y de ejemplos de aplicaciones prácticas.

La introducción de un tema, generalmente es precedida por un diálogo dirigido, con preguntas orientadas hacia el tema a tratar, lo que induce a la participación de todo el grupo.

A partir de esto se desarrolla la exposición teórica con ejercitación práctica en el aula, esta exposición puede ser apoyada por una lectura previa recomendada a los alumnos. Los conceptos impartidos son reforzados y puestos en práctica con los ejercicios propuestos en la Guía de Ejercicios confeccionada por la cátedra. Esa ejercitación permite al alumno confrontar los nuevos conocimientos con los previamente adquiridos y aplicar los conceptos vistos teóricamente, a nuevas situaciones. Algunos ejercicios son presentados, discutidos y resueltos en el aula por el docente

Además la cátedra dispone de una guía de trabajos prácticos para ser desarrollados por los alumnos en forma grupal en el laboratorio de Sistemas Abierto que dispone la cátedra. Cada tema propuesto se verifica en tiempo y forma en el laboratorio observando su correcto funcionamiento en la computadora. Dentro de este ámbito el alumno dispone de atención permanente de docentes para aclarar todas sus consultas. Este estilo de trabajo es abordado durante todas las clases. Cada Trabajo práctico Grupal se deberá defender en forma individual para su aprobación.

Se utilizará material audiovisual cuando las circunstancias así lo requieran.

6.- DESCRIPCIÓN DE LA ACTIVIDAD CURRICULAR

Por parte del Profesor:

- Desarrollo de clases de exposición de temas teóricos Utilizando Pizarrón y en ocasiones con presentaciones por computadora.
- Desarrollo de clases prácticas de resolución de ejercicios de aplicación de los conceptos teóricos.
- Desarrollo de clases prácticas en laboratorio, mostrando ejemplos significativos de la teoría y demostraciones prácticas que deben realizar los alumnos.
- Actualización de contenidos en la página de la cátedra donde se encuentra toda la documentación de la asignatura y medio de comunicación para envío de noticias, material y dar respuesta a los requerimientos de los alumnos (acceso vía e-mail de y a los alumnos).

Por parte de los alumnos:

- Resolución individual de ejercicios de aplicación de la teoría propuestos por el profesor.
- Desarrollo e implementación grupal de una a serie de trabajos Prácticos diseñados especialmente para que el grupo de alumnos los programe.

Material Didáctico:

- Diapositivas Power Point de Clase sobre temas teóricos
 - Guías de Trabajos Prácticos y Ejemplos de Resolución de Ejercicios desarrollados por la cátedra
 - Procedimientos escritos para diversos procesos.
 - Bibliografía básica y avanzada.
 - Uso del sitio de la Asignatura.
- Envío y atención de mail para consultas

7.- EXPERIENCIAS DE LABORATORIO, TALLER O TRABAJOS DE CAMPO

NOTA: Todos los trabajos prácticos que se realicen en ésta materia serán corregidos en un entorno GNU/Linux. Para mayor detalle de las versiones utilizadas concurrir al laboratorio 266.

En caso de que el alumnado decida realizar los trabajos en otra plataforma, o en otra distribución, deberá tomar las precauciones necesarias para que el producto entregado pueda ser ejecutado en dicho ambiente. En caso de que el producto entregado no cumpla con éstas indicaciones, el trabajo práctico será considerado como no entregado.

Trabajo Práctico Nro. 1:

Scripting

Trabajo Práctico Nro. 2:

Scripting

Trabajo Práctico Nro. 3:

Programación de comunicación y procesos

Defensa de TP's (Coloquio) y Recuperatorios

Todos los Trabajos prácticos serán probados por los docentes y defendidos por los alumnos en el Laboratorio. En las defensas podrán estar presentes además de los docentes del curso, el jefe de cátedra, y el jefe de trabajos prácticos.

8.- USO DE COMPUTADORAS

En la asignatura se utilizan profusamente computadoras en experiencias de simulación y de programación de Sistemas como complemento práctico de la Teoría. Para ello se recurrirá al Laboratorio específico de Sistemas Operativos (aula 266).

9.- METODOLOGÍA DE EVALUACIÓN

- Esta asignaturas se evaluará de acuerdo a la reglamentación vigente en la Universidad y la que se detalla en el "Reglamento de cursado y aprobación de la materia".
- Se efectuarán dos evaluaciones parciales: El primero al promediar el dictado del curso y el segundo al finalizar el mismo según el Calendario Académico.
- Asimismo y como **condición necesaria para la aprobación del curso** se examinará al alumnado mediante una Guía de Trabajos Prácticos que se desarrollará durante el transcurso del mismo, además de las exposiciones orales que efectuarán los alumnos sobre los T.P., cuestionarios o problemas teóricos planteados.
- Además se requiere una asistencia a clase no inferior al 75%, se hace un seguimiento del trabajo realizado por cada integrante en cada clase.

El conjunto formado por los Trabajos Prácticos y las evaluaciones parciales serán el instrumento para medir el rendimiento y la aprobación de la cursada.

Aprobación y cursada

A los fines de la aprobación de la materia, se considera "la última nota obtenida" en cada uno de los exámenes rendidos (en primera instancia ó recuperatorios).

- a) Por régimen de promoción, sin examen final, se considera la materia **aprobada**, cuando la calificación es igual o superior a 7 (siete) a través de exámenes parciales y recuperatorios, en las fechas indicadas en el cronograma.
- b) Si el alumno no llena los requisitos para promover (calificación superior o igual a 4 pero inferior a 7 puntos), queda en condición de **cursada**. Para su aprobación definitiva tiene que rendir posteriormente un examen final. La validez de la cursada será de 5 turnos consecutivos de examen final. Dichos turnos serán contados a partir del turno inmediato siguiente al periodo de cursada.
- c) El alumno que tenga 1 (un) aplazo en las evaluaciones y/o recuperatorios, y haya estado presente en todas las instancias evaluativas, pierde la materia y se considera **desaprobado**.
- d) Aquel alumno que tenga al menos 1 (un) examen cuya evaluación final sea ausente (considerando parcial y recuperatorio), se considera **ausente**.

Régimen de Trabajos Prácticos

- 1- En fecha de entrega del TP, se hará una corrección grupal de ejercicios en forma arbitraria para cada grupo.
- 2- La NO presentación del TP o estar incompleto en la fecha propuesta significa su desaprobación en primera instancia (es para todos los integrantes del grupo).
- 3- La realización de cada TP será grupal pero su evaluación individual a través de un examen escrito u oral en todos los trabajos prácticos.
- 4- El alumno que no apruebe la evaluación del TP o NO se entregó en fecha establecida, tendrá una nueva fecha que es a la semana siguiente del establecido inicialmente en el cronograma.
- 5- El alumno que desapruebe 2(dos) TP en segundas instancias (por ausencia, estar incorrecto, y /o incompleto o no responder correctamente el coloquio) desaprobará la materia.

Examen Final

- Los alumnos pueden rendir examen final bajo dos modalidades **regular** o **libre**.
- Para rendir examen como **regular** deberá tener la materia cursada y no haberse operado el vencimiento de la misma.
- Deberán rendir como regular los que obtengan entre cuatro y seis en los parciales o sus recuperatorios.
- Para rendir examen como **libre** tendrán que ajustarse a la reglamentación vigente.
- La mesa examinadora considerará válidas las inscripciones que consten en las actas proporcionadas por la oficina de alumnos.
- Cada alumno rendirá el final con el programa vigente.

Condiciones para rendir EXAMEN LIBRE

Por Resolución N° 142 del H.C.S., se autoriza a rendir “**exámenes libres**” de todas las asignaturas, a los alumnos de las tres Carreras pertenecientes al Departamento de Ingeniería e Investigaciones Tecnológicas.

Todos los alumnos estarán en condiciones de rendir exámenes libres, siempre y cuando hayan aprobado las materias correlativas correspondientes.

Dicha instancia examinadora se deberá llevar a cabo en una de las fechas de convocatoria a exámenes finales.

Para mayor detalle sobre la forma de rendir exámenes libres y los requerimientos a cumplir antes de presentarse en la llamada correspondiente ver la sección REGLAMENTO DE CURSADA LIBRE DE LA CÁTEDRA SISTEMAS DE COMPUTACIÓN II (PLAN 1997) / SISTEMAS OPERATIVOS (PLAN 2009) más adelante en el presente documento

CALENDARIO DE ACTIVIDADES (modalidad presencial)

PLANIFICACIÓN DOCENTE PARA EL AÑO 2020

DURACIÓN DE CADA CURSO:

- Teórica: Aprox. 8 clases de 4 horas. (32 horas)
- Práctica: Aprox. 9 clases de 4 horas. (36 horas)
- Laboratorio: Aprox. 13 clases de 4 horas. (52 horas)

HORARIO: Según el fijado para cada curso (turno mañana 8 a 12 / turno noche 19 a 23)

CRONOGRAMA DE ACTIVIDADES DE LA PLANIFICACIÓN POR CURSO

Clase 1: Introductoria-Práctica. Presupuesto de tiempo: 4 Hs.

Fecha: Comienzo del ciclo lectivo 2020- (06/04/2020)

Objetivos:

- Definir la metodología para el futuro desarrollo del curso y dar los lineamientos introductorios al curso. Explicar la metodología de evaluación de TPs y exámenes parciales
- Introducción al sistema operativo GNU/Linux

Tipo de conocimiento:

- Práctico

Evaluación del Módulo:

- Primer parcial

MÓDULO 1: Presupuesto de tiempo: 2 Hs.

Objetivos:

- Que el alumno incorpore un enfoque introductorio sobre los Sistemas Operativos, sus interfaces, los servicios que brinda, su funcionamiento y conozca la terminología básica y conceptual de los S.O. y sus ambientes de trabajo.

Tipo de conocimiento:

- Teórico

Evaluación del Módulo:

- Primer parcial

MÓDULO 2: Presupuesto de tiempo: 2 Hs.

Objetivos:

- Que el alumno adquiera los conocimientos sobre las distintas modalidades de procesamiento e incorpore los conceptos fundamentales sobre organización del ambiente de ejecución.

Tipo de conocimiento:

- Teórico

Evaluación del Módulo:

- Primer parcial

PRÁCTICO 1: Presupuesto de tiempo: 8 Hs.

Objetivos:

- Que el alumno incorpore los conocimientos para la codificación de scripts básicos y con las herramientas awk y sed

Tipo de conocimiento:

- Práctica de laboratorio

Evaluación del Módulo:

- Coloquio del TP

EVALUACIÓN PRÁCTICO 1: Presupuesto de tiempo: 1 Hs.

Objetivos:

- Evaluar a los alumnos sobre los conocimientos de GNU/Linux adquiridos durante el trabajo práctico 1

MÓDULO 3: Presupuesto de tiempo: 8 Hs.

Objetivos:

- Que el alumno se familiarice con los conceptos y los medios de la planificación del procesador y de los procesos, en especial en el largo, mediano y corto plazo.

Tipo de conocimiento:

- Teórico y práctico

Evaluación del Módulo:

- Primer parcial

MÓDULO 4: Presupuesto de tiempo: 10 Hs.

Objetivos:

- Que el alumno integre los conceptos fundamentales sobre los recursos compartidos, sincronización y comunicación entre procesos.

Tipo de conocimiento:

- Teórico y práctico

Evaluación del Módulo:

- Primer parcial

PRÁCTICO 2: Presupuesto de tiempo: 6 Hs.

Objetivos:

- Que el alumno incorpore los conocimientos en PowerSell necesarios para los trabajos prácticos

Tipo de conocimiento:

- Práctica de laboratorio

Evaluación del Módulo:

- Coloquio del TP

EVALUACIÓN PRÁCTICO 2: Presupuesto de tiempo: 1 Hs.

Objetivos:

- Evaluar a los alumnos sobre los conocimientos de GNU/Linux adquiridos durante el trabajo práctico 2

EVALUACIÓN 1: Presupuesto de tiempo: 4 Hs.

Objetivos:

- Evaluar a los alumnos sobre los conocimientos teóricos y prácticos.

MÓDULO 5: Presupuesto de tiempo: 10 Hs.

Objetivos:

- Que el alumno concrete los conceptos sobre la administración de la Memoria Central, en especial las particiones y los conceptos de asignación, paginación y segmentación.

Tipo de conocimiento:

- Teórico y práctico

Evaluación del Módulo:

- Segundo parcial

MÓDULO 6: Presupuesto de tiempo: 6 Hs.

Objetivos:

- Que el alumno incorpore los conceptos sobre la administración de los dispositivos de Entrada - Salida.

Tipo de conocimiento:

- Teórico y práctico

Evaluación del Módulo:

- Segundo parcial

MÓDULO 7: Presupuesto de tiempo: 6 Hs.

Objetivos:

- Que el alumno conozca los métodos de acceso para el almacenamiento y la recuperación de la información en los soportes como también la administración de la misma.

Tipo de conocimiento:

- Teórico y práctico

Evaluación del Módulo:

- Tercer parcial

MÓDULO 8: Presupuesto de tiempo: 1 Hs.**Objetivos:**

- Que el alumno conozca los fundamentos y los conceptos sobre el manejo de la protección y la seguridad de un centro de cómputo y el S.O.

Tipo de conocimiento:

- Teórico

Evaluación del Módulo:

- Segundo parcial

PRÁCTICO 3: Presupuesto de tiempo: 10 Hs.**Objetivos:**

- Que el alumno incorpore los conocimientos necesarios para realizar procesos concurrentes en Linux

Tipo de conocimiento:

- Práctica de laboratorio

Evaluación del Módulo:

- Coloquio del TP

EVALUACIÓN PRÁCTICO 3: Presupuesto de tiempo: 1 Hs.**Objetivos:**

- Evaluar a los alumnos sobre los conocimientos de GNU/Linux adquiridos durante el trabajo práctico 3

EVALUACIÓN 2: Presupuesto de tiempo: 4 Hs.**Objetivos:**

- Evaluar a los alumnos sobre los conocimientos teóricos y prácticos.

MÓDULO 9: Presupuesto de tiempo: 1 Hs.**Objetivos:**

- Que el alumno adquiera los conceptos básicos sobre métrica de sistemas

Tipo de conocimiento:

- Teórico

Evaluación del Módulo:

- Sin evaluación

MÓDULO 10: Presupuesto de tiempo: 6 Hs.**Objetivos:**

- Que el alumno adquiera los conceptos básicos sobre los sistemas operativos distribuidos, sus problemáticas y la forma de implementar las soluciones

Tipo de conocimiento:

- Teórico

Evaluación del Módulo:

- Sin evaluación

RECUPERACIÓN 1: Presupuesto de tiempo: 4 Hs.**Objetivos:**

- Que los alumnos tengan la posibilidad de recuperar los exámenes que tengan aplazados o con notas menores a 7

REGLAMENTO DE PROMOCIÓN y NORMAS DE LA CÁTEDRA

(Cursada normal)

1. La aprobación de la asignatura *Sistemas de Computación II (Plan 1997) / Sistemas Operativos (Plan 2009)* en el período lectivo **2020** se basará en:
 - 1.1. La Normativa vigente en la Universidad,
 - 1.2. Las Normas Básicas de la Cátedra *Sistemas de Computación II (Plan 1997) / Sistemas Operativos (Plan 2009)* que se detallan a continuación en tanto reglen aspectos no normados por los elementos anteriores.
 - 1.3. El Reglamento Interno de la *Sistemas de Computación II (Plan 1997) / Sistemas Operativos (Plan 2009)* que se detalla a continuación, en este documento.

NORMAS BÁSICAS DE LA CÁTEDRA

1. El dictado de la materia se dividirá en aproximadamente 32 clases teóricas y clases prácticas, según calendario adjunto, las que incluirán clases teóricas, práctica, de evaluación y de recuperación.
2. En las clases de contenido teórico se desarrollarán los temas teóricos establecidos en el programa analítico adjunto. En las clases prácticas los alumnos, orientados por los docentes a cargo de las mismas, resolverán problemas y ejercicios de aplicación de los temas vistos en clase y los de la presente guía de trabajos prácticos.
3. La aprobación de los trabajos prácticos (Firma de Libreta), se obtendrá a través de:
 - 3.1. La presentación y aprobación de los trabajos prácticos, según lo detallado en la presente.
 - 3.2. La aprobación de, al menos, dos exámenes parciales en las fechas y condiciones establecidas en el calendario adjunto y en el régimen de cursado y aprobación de ambas asignaturas siguiente:

Régimen de cursado y aprobación de la asignatura *Sistemas de Computación II (Plan 1997) / Sistemas Operativos (Plan 2009)*

1. ASISTENCIAS:

- Se requiere una asistencia a clases no inferior al 75% (setenta y cinco %) tanto para la modalidad presencia como la semipresencial. El incumplimiento de este requisito coloca al alumno en condición de "ausente" o "desaprobado".

2.- RÉGIMEN DE PROMOCIÓN POR EXÁMENES PARCIALES Y RECUPERATORIOS:

- La asignatura se aprueba por régimen de promoción por exámenes parciales y/o recuperatorios.
- Para esta asignatura el cursado es cuatrimestral y habrá 2 (dos) evaluaciones parciales: uno por bimestre. Las instancias recuperatorias serán 1 (una). En esta asignatura se entiende "**ausente**" al alumno que no posea 2 (dos) evaluación parciales rendidas y no haya entregado más de 2 (dos) trabajos prácticos.
- Los exámenes parciales (y sus recuperatorios) se entenderán "**aprobados**" cuando la calificación asignada, en una escala de 0 a 10 puntos, resulte superior o igual a 7 (siete) puntos.
- La asignatura se entenderá "**aprobada**" cuando se aprueben todos los exámenes parciales (en primera instancia o por recuperatorio). La calificación asignada al examen recuperatorio (cualquiera sea el resultado), **anula y reemplaza**, a todos los efectos, la obtenida en el examen parcial que se recupera. La calificación final se calculará como promedio de los exámenes parciales o el último recuperatorio de cada parcial, rendidos y aprobados. Es importante mencionar que en caso de tener una nota entre 4 (cauto) y 6 (seis) en el parcial y sacarse un 2 (dos) en el recuperatorio, la materia se considerará como "**desaprobada**".
- De esta manera la calificación final necesaria para que la asignatura resulte "**aprobada**" deberá ser superior o igual a 7 (siete) puntos, pero no se podrán poseer notas menores a 7 (siete).

3. RÉGIMEN NO PROMOCIONADO DE PARCIALES Y SUS RECUPERATORIOS:

- En la asignatura, los exámenes parciales (y sus recuperatorios) calificados con 3 (tres) o menos puntos se entenderán "**aplazados**" y podrán ser recuperados.
- Si el alumno al finalizar la cursada tiene algún parcial (o recuperatorio) y/o trabajo práctico calificados con aplazo, se considerará "**aplazada**" la materia y deberá ser cursada en otro cuatrimestre.
- Los exámenes parciales calificados con 4 (cuatro), 5 (cinco) ó 6 (seis) puntos, se entenderán "**aprobados**" y podrán ser recuperados (en caso de que el alumno desee la promoción de la materia), pero no se podrá con estas notas conseguir la calificación final de "**Aprobado**", teniendo en este caso la condición final de "**Cursada**".
- La asignatura con calificación final, calculada como promedio de los exámenes parciales (o los recuperatorios correspondientes) rendidos y no aplazados, de 4 (cuatro), 5 (cinco) o 6 (seis) puntos, se entenderán "**cursada**" y podrá ser aprobada a través de un "**examen final**".
- La calificación necesaria para aprobar el "**examen final**" será de 4 o más puntos

4.- LA VALIDEZ DE LA ASIGNATURA "CURSADA**"**

- La validez de la asignatura "**cursada**" se rige bajo las normas de la facultad, por lo que se deberá consultar con las autoridades pertinentes.

5.- PREREQUISITO CONDICIONANTE PARA RENDIR LOS EXAMENES PARCIALES:

- Esta asignatura requiere que el alumno tenga el porcentaje de asistencia correspondiente para poder rendir los exámenes parciales y recuperatorios

REGLAMENTO INTERNO DE LA CÁTEDRA SISTEMAS DE COMPUTACIÓN II (PLAN 1997) / SISTEMAS OPERATIVOS (PLAN 2009)

1. OBJETIVO:

- Dar las bases normativas por las que se regirá el funcionamiento y el desarrollo operativo de la cátedra.

2. ALCANCES:

- El presente Reglamento **NO EXCLUYE** a la reglamentación vigente, sino todo lo contrario, pretende complementarla para lograr las metas operativas propuestas para cada curso en particular.

3. CONTENIDO:

a) **DE LOS PROGRAMAS:** El contenido es el indicado en el programa analítico de la materia.

b) **DEL CRONOGRAMA DE ACTIVIDADES:** Se ajustará de acuerdo al presupuesto de tiempo previsto en la planificación docente y se formalizará el primer día de clase en cada curso en particular.

c) **DE LA ASISTENCIA:** Es de recalcar que la asistencia, en el caso específico de ésta materia, juega un rol importante debido al intenso ritmo que se impartirá al dictado de las clases teóricas, por lo tanto se recomienda al alumno concurrir a dichas clases, siendo de su exclusiva responsabilidad cumplir con este requisito. Pero se aplicará el punto 1 del Régimen de cursado y aprobación de las asignaturas *Sistemas de Computación II (Plan 1997) / Sistemas Operativos (Plan 2009)*.

d) **DEL HORARIO:** En el inicio de la clase, la puntualidad es importante a los fines de constituir un ambiente ordenado. Se recomienda al alumnado el cumplimiento de este requerimiento. En particular también se recomienda la permanencia dentro del aula mientras se desarrollan las clases.

e) **DE LAS CLASES TEÓRICAS:** El Docente y sus Ayudantes dictarán la materia tratando de seguir la secuencia estricta de los módulos y la Planificación propuesta. El desarrollo tendrá un carácter ampliamente comunicativo que permita la participación del alumnado. El método a aplicar será explicativo-inductivo-deductivo. Tanto el docente o sus Ayudantes evacuarán las dudas que surjan durante el dictado de las clases o de los T.P.

f) **DE LOS TRABAJOS PRÁCTICOS:** Los alumnos confeccionarán una serie de Trabajos Prácticos (TPs.), para ello se dispondrá de una Guía de T.P. (adjunta al presente documento).

Cada Guía deberá ser completada en la fecha establecida por la Cátedra o el docente a cargo del curso. La totalidad de las guías formarán una “**carpeta de T.P.**”

Los T.P. se dividirán en dos categorías: 1) Optativos y 2) Obligatorios.

Cada guía deberá ser entregada, por el alumno o el grupo de alumnos, **en la fecha planificada a los efectos de ser corregida.**

OBSERVACIÓN: LOS T.P. NO ENTREGADOS EN FECHA SE CONSIDERAN NO APROBADOS.

La guía corregida por la cátedra será devuelta con las observaciones correspondientes para que los alumnos procedan a rectificar lo solicitado. Una vez cumplimentado por los alumnos, en el plazo fijado, los T.P. serán entregados a la cátedra para su aprobación. La cátedra firmará la aprobación parcial de cada Guía y devolverá el original para que cada alumno pueda disponer de una constancia de la aprobación, la que integrará una “carpeta de TPs”. Todos los TPs originales aprobados formarán una “carpeta de T.P. originales” que deberá ser presentada al final de la cursada.

g) **DE LA PRESENTACIÓN DE LOS T.P.:** La presentación se deberá realizar en dos soportes: Hojas de papel y medio electrónico (ver reglamento de entrega más adelante en esta guía).

La presentación que se realiza en hojas de papel, deberá ser normalizada en papel A-4, o carta (no se aceptarán entregas en papel oficio), y los contenidos impresos se ajustarán a las “**Normas para la presentación escrita de los Trabajos Prácticos**” que figuran en la guía de T.P.

h) **DE LA EVALUACIÓN DE LOS T.P.:** Todos los puntos se evaluarán mediante las consideraciones en particular de cada ítem siguiente:

- Desarrollo por temas (extensión).
- Contenidos (Calidad y en el caso de programas: funcionamiento).
- Criterios.
- Síntesis.
- Definiciones (acotaciones).
- Alcances.
- Investigación Bibliográfica.
- Presentación.

El conjunto de notas dará como resultado la aprobación o desaprobación del T.P. en particular.

i) **DE LA REGULARIZACIÓN DE LA MATERIA:** Para la firma de la Libreta, el alumno deberá presentar:

- La Libreta Universitaria.
- Haber aprobado los T.P. realizados durante el curso ya sean grupales o individuales.

- Tener todos los parciales aprobados y cumplir con lo dispuesto en el Régimen de cursado y aprobación de la asignatura Sistemas de Computación II (Plan 1997) / Sistemas Operativos (Plan 2009).
- Ser alumno regular.
- Realizar el "POSTEST" que propondrá la Cátedra y la encuesta.

j) **DE LAS EVALUACIONES DURANTE EL CURSO:** Habrá dos evaluaciones parciales durante el curso. El docente fijará con cada curso fecha de cada uno de esos parciales y la del recuperatorio. Habrá un recuperatorio en el que podrá rendirse uno de los dos parciales según lo especificado en el régimen de aprobación de la materia.

Los T.P. grupales serán expuestos en el pizarrón o en una reunión grupal con el Jefe de Trabajos Prácticos o docente del curso, por cada integrante del grupo a los fines de examinar su participación en el desarrollo del T.P. y que dará lugar a una evaluación de cada presentación individual. Además de considerar una nota única por cada T.P. grupal.

k) **DE LAS EVALUACIONES FINALES:** Los mismos pueden ser Teórico-prácticos y en forma escrita y/u oral, según lo aconsejen las circunstancias.

La examinación se hará a través de un Tribunal Examinador. Para poder rendir el examen final los alumnos deberán tener regularizada la materia y la correlatividades respectivas de esta materia.

REGLAMENTO DE CURSADA LIBRE DE LA CÁTEDRA SISTEMAS DE COMPUTACIÓN II (PLAN 1997) / SISTEMAS OPERATIVOS (PLAN 2009)

1. OBJETIVO:

- Dar las bases normativas por las que se implementará la aprobación de la materia a través de exámenes libres para conseguir la aprobación de la cátedra.

2. ALCANCES:

- El presente Reglamento **NO EXCLUYE** a la reglamentación vigente, sino todo lo contrario, pretende complementarla para lograr las metas operativas propuestas para cada curso en particular.

3. CONTENIDO DEL EXAMEN LIBRE:

- a) **DE LOS TRABAJOS PRÁCTICOS:** El alumno que desee rendir la materia en condición de libre deberá efectuar **TODOS** los trabajos prácticos que la cátedra haya dispuesto para el cuatrimestre en curso vigente en la guía de trabajos prácticos que suministra la materia, confeccionándolos y teniéndolos que presentar con 15 días de anticipación a la fecha de rendir el examen libre. La vigencia de los trabajos prácticos para el examen libre será desde el comienzo del cuatrimestre correspondiente al que se quiere rendir el examen libre hasta el comienzo del próximo cuatrimestre. Esto quiere decir que para rendir exámenes libres se deberán tener los trabajos prácticos ya vencidos aprobados:

Llamada	TPs aprobados
Julio 2020	Segundo cuatrimestre 2019
Diciembre 2020	Primer cuatrimestre 2020
Febrero / Marzo 2021	Segundo cuatrimestre 2020
Julio 2021	Segundo cuatrimestre 2020

Nota: En las fechas adicionales no se puede rendir en modalidad libre según reglamento académico.

- b) **DE LA EVALUACIÓN DE LOS T.P.:** Los trabajos prácticos entregados por el alumno que rinde el examen libre serán evaluados en los 15 días que hay hasta la fecha del examen final por los docentes de la cátedra y en caso de estar bien, el alumno deberá rendir un coloquio como primera parte del examen final. Los puntos se evaluarán mediante las consideraciones en particular de cada ítem siguiente:

- Desarrollo por temas (extensión).
- Contenidos (Calidad y en el caso de programas: funcionamiento).
- Criterios.
- Síntesis.
- Creatividad.
- Definiciones (acotaciones).
- Alcances.
- Investigación Bibliográfica.
- Presentación.

El conjunto de notas dará como resultado la aprobación o desaprobación de los trabajos prácticos.

- c) **DE LA EVALUACIÓN FINAL:** En caso de aprobar los trabajos prácticos (tanto la presentación, como el coloquio), el alumno deberá rendir un examen final para la condición de libre, que tendrá una primera parte práctica escrita (conteniendo ejercicios tanto de la práctica de clases como de la práctica de laboratorio). En caso de aprobar dicho examen deberá pasar un examen teórico con carácter oral que incluirá todo el contenido de la materia que se indica en el PROGRAMA ANÁLITICO de la materia.

“Certifico que el presente programa de estudios de la asignatura *Sistemas de Computación II (Plan 1997) / Sistemas Operativos (Plan 2009)* es el vigente para el segundo cuatrimestre del ciclo lectivo 2020, guarda consistencia con los contenidos mínimos del plan de estudios y se encuentra convenientemente actualizado”

Fabio E. Rivalta
Jefe de Cátedra
Universidad Nacional de La Matanza
06/04/2020

Descripción de los Trabajos Prácticos

Trabajo Práctico Nro. 1:

Scripts con bash, awk, sed y señales en sistema operativo GNU/Linux

Trabajo Práctico Nro. 2:

Scripts con PowerShell en Windows

Trabajo Práctico Nro. 3:

Desarrollo de procesos y comunicación en C / C++ bajo Linux

Defensa de TP's (Coloquio) y recuperatorio de parciales:

El recuperatorio de parciales será coordinado por los docentes con los alumnos durante el transcurso de la cursada.

Los recuperatorios de los trabajos prácticos y coloquios serán a partir del 01 de julio de 2020, en el horario de cursada

Notas importantes:

- Si un trabajo no es entregado en la fecha correspondiente, no se aceptará la entrega hasta la fecha de recuperación.
- Para los trabajos prácticos que tengan fechas de entregas graduales, se considerará cada entrega parcial como un trabajo práctico por separado. De forma que si un trabajo práctico está dividido en dos entregas, serán como dos trabajos independientes pero que conformarán una sola nota; y para la nota final del trabajo práctico se tomará el promedio de las notas de las dos entregas.
- No se podrá entregar un trabajo práctico sin que se encuentre entregado el anterior, y aprobados todos los trabajos prácticos previos al anterior. Ej.: Si se quiere entregar el trabajo práctico número 3, se deberá tener presentado para su evaluación el trabajo práctico 2, y aprobado el trabajo práctico 1. Ver más detalle en el Reglamento de entrega

Reglamento de entrega y reentrega de Trabajos prácticos

Todos los trabajos prácticos tienen el mismo formato de entrega o reentregas que es descrito a continuación, y que en caso de no ser cumplido no será considerado como entregado el trabajo práctico.

Entrega escrita:

Por cada entrega o reentrega de trabajo práctico se deberá entregar una carpeta o folio conteniendo:

- **Una (1) carátula** utilizando para tal fin la plantilla disponible en el sitio web de la cátedra debidamente completada y firmada por cada uno de los integrantes del curso.
Las reentregas se realizarán con una nueva carátula conteniendo toda la información pertinente para que los docentes puedan realizar la corrección y deberá estar también firmada por todos los integrantes del grupo que sigan cursando la materia.

Importante: No se aceptarán trabajos prácticos cuyas carátulas no estén debidamente firmadas por todos los integrantes del grupo

- Para poder realizar la entrega de cualquier trabajo práctico se deberá cumplir con las siguientes condiciones para cada uno de ellos:

TP	TPs entregados	TPs Aprobados
1	--	--
2	TP 1	--
3	TP 1 y TP 2	TP 1

Entrega digital:

Los trabajos prácticos deberán entregarse mediante la opción [Entrega de TPs](#) del sitio web de la cátedra (<http://www.sisop.com.ar>). Es importante tener en cuenta que la entrega la deberá realizar sólo uno de los integrantes de cada grupo. Queda a consideración de cada equipo de trabajo quién será el que realice las entregas.

El sistema aceptará un único archivo comprimido por cada entrega, con extensiones **zip, gzip, gz o tgz**.

El avance de las correcciones, feedback de los profesores y notas podrán visualizarlos todos los integrantes del grupo (sin importar quién hizo la entrega) desde la opción de menú [Mis Notas](#).

Importante: Tomando en cuenta las tareas de rediseño del sitio web que se están realizando durante el cuatrimestre podría no estar disponible este servicio en todo momento por lo que se recomienda a los estudiantes coordinar con los docentes para obtener las notas y correcciones a realizar en el Laboratorio 266 cuando estos servicios no estén disponibles.

La no disponibilidad de las notas en el sitio Web no elimina la obligatoriedad por parte de los alumnos de obtener sus calificaciones ni los exime de las reentregas de los trabajos prácticos debidamente corregidos.

Entrega presencial:

El cuerpo docente estará disponible para realizar la evaluación temprana de los trabajos prácticos durante el día de entrega correspondiente a cada trabajo práctico.

El objetivo de esta evaluación temprana es que el grupo o al menos uno de los integrantes de cada grupo, realice la evaluación de cada uno de los ejercicios incluidos en el TP, y de esta manera el grupo tenga ya la información sobre si cada uno de los ejercicios está o no en condiciones de ser evaluado por el cuerpo docente.


La idea de esta evaluación temprana es que tanto el grupo docente como el equipo de trabajo se enteren de los problemas detectados en la solución brindada y el grupo pueda realizar las correcciones lo antes posible.

Modalidad de trabajo el día de entrega:

- Se recibirán carátulas para la evaluación de trabajos prácticos durante las dos primeras horas de la jornada
- Al menos un integrante de cada grupo debe estar presente para poder efectuar la corrección temprana
- El orden de recepción de las caratulas será utilizado como orden de atención de los grupos

- Un integrante del cuerpo docente llamará al grupo luego de haber bajado el trabajo práctico desde el sitio web de entrega y en conjunto procederán a realizar las evaluaciones suministradas por el grupo de trabajo que se incluyan en la solución.
- El resultado de la evaluación realizada en cada uno de los ejercicios del trabajo práctico será el rechazo por no cumplir con los objetivos o la aceptación por parte del cuerpo docente lo que dará lugar a una prueba más exhaustiva a realizar solo por el cuerpo docente.
- En caso que un ejercicio sea desaprobado el equipo de trabajo deberá realizar la reentrega lo antes posible para que el cuerpo docente pueda realizar la evaluación.
- Las reentregas y TPs entregados en forma no presencial serán evaluados posteriormente a los TPs entregados en forma presencial.

Nota: La aceptación del ejercicio no significa la aprobación ya que solo se realizarán las pruebas mínimas correspondientes a la entrega realizada por parte del equipo de trabajo. La aprobación final se brindará por el canal normal en los días posteriores a la entrega.



Guía orientativa para el uso de GNU/Linux

- **Descripción:** A continuación se detallan una serie de preguntas y ejercicios orientados al uso de una terminal de caracteres de un ambiente multiusuario y multitarea basado en GNU/Linux. La intención buscada con esta guía es que aquellos alumnos que no estén familiarizados con una terminal de caracteres ni con la familia de sistemas operativos GNU/Linux, puedan aprender las principales características y poder realizar los trabajos posteriores con un mejor conocimiento del entorno en el que se deben realizar.
- **Nota:** Si bien este no es un trabajo práctico de entrega obligatoria, aquel alumno/a que crea conveniente realizarlo y desee consultar a los docentes o ayudantes de la materia sobre su contenido podrá hacerlo en cualquiera de las clases prácticas. Desde el cuerpo docente recomendamos a todos aquellos alumnos que nunca hayan trabajado en un ambiente de este tipo realicen esta guía y consulten a los docentes sobre los temas aprendidos y los no comprendidos
- **Formato de entrega:** sin entrega o con entrega de consulta optativa
- **Preguntas:** A continuación se detallan todas las preguntas y ejercicios que deberán ser resueltos. Tenga en cuenta que salvo en los momentos que indica que debe estar sesionado como **root**, en el resto de los ejercicios debe estar conectado como usuario común. (TIP: se le recomienda que primero realice todo el trabajo, anotando los resultados en papel, y a mano, y luego lo pase con el editor vi).

1. INTRODUCCIÓN

- 1.1. ¿Qué es la cuenta de superusuario (root) y para qué se utiliza?
- 1.2. Ingresar al sistema como superusuario (root), y realizar los siguientes pasos (éste punto no puede ser realizado en el laboratorio 266):
 - 1.2.1. adduser <apellido> (reemplazar <apellido> por el suyo).
 - 1.2.2. passwd <apellido> (Ingrese una contraseña (password) a su elección).
 - 1.2.3. logout
- 1.3. Indique claramente qué efectuaron estos comandos, e indique qué archivo/s fueron modificados (Dentro del directorio /etc) **TIP:** Utilice lo siguiente: "ls -lt /etc | more".
- 1.4. Luego ejecute "cat /etc/passwd | more" y haga lo mismo con los otros archivos que se modificaron. Analice y comente lo visto
- 1.5. ¿En qué directorio se encuentran los comandos utilizados en los puntos 1.2.1, 1.2.2, 1.2.3, 1.3, y 1.4?

2. AYUDA

- 2.1. INFO: Info es un programa para leer documentación. Este se compone de una estructura del tipo árbol, dividido en nodos de información. Cada nodo describe un específico tópico con un determinado nivel de detalle.
 - 2.1.1. Ingrese a info y responda:
 - 2.1.1.1. ¿Cómo se llama el nodo raíz de Info?
 - 2.1.1.2. Ubique el cursor en la línea (* cp:) y presione ENTER.
 - 2.1.1.3. ¿Qué sucedió?
 - 2.1.1.4. ¿Cómo se llama este nodo?
 - 2.1.1.5. ¿Cuál es el próximo nodo?
 - 2.1.1.6. ¿Cómo puedo moverme al próximo nodo?
 - 2.1.1.7. ¿Cómo puedo moverme al nodo anterior?
 - 2.1.2. Presione la tecla 'u'.
 - 2.1.2.1. ¿Qué sucedió?
 - 2.1.2.2. ¿En qué nodo se encuentra?
 - 2.1.3. Repita el punto 2.1.2. hasta que llegue a la raíz de Info.
 - 2.1.3.1. ¿Con qué tecla puedo volver directamente a este nodo?
 - 2.1.3.2. ¿Cuál es el método directo para acceder al nodo cp? (tip: sin desplazar el cursor).
 - 2.1.4. ¿Cómo puedo buscar una palabra clave dentro de un nodo?
 - 2.1.5. ¿Cómo puedo buscar la siguiente palabra clave, buscada anteriormente?
 - 2.1.6. ¿Cómo puedo salir de Info? - salga.
- 2.2. MAN: man es un programa que formatea y muestra la páginas del manual.
 - 2.2.1. ¿Cuál es la diferencia entre man e info?
 - 2.2.2. ¿Cómo puedo ver la información de un determinado comando?
 - 2.2.3. ¿Cómo puedo buscar una palabra clave dentro de la página del manual?
 - 2.2.4. ¿Cómo puedo salir?
 - 2.2.5. ¿Cómo hago para buscar una palabra clave determinada en todas las páginas del manual?
 - 2.2.6. ¿Qué es lo sucede al realizar lo siguiente?
 - 2.2.6.1. man
 - 2.2.6.2. man man
 - 2.2.6.3. man cp
 - 2.2.6.4. man printf
 - 2.2.6.5. man fprintf
 - 2.2.6.6. man sprintf

- 2.2.6.7. `man cd`
- 2.2.6.8. `man 3 printf`
- 2.2.7. Del punto anterior, responder:
 - 2.2.7.1. Al invocar `man` junto con `fprintf` y `sprintf` muestra la misma página. ¿Por qué no muestra la misma página al invocarlo con `printf`? (TIP: vea el punto 3.2.6.2).
 - 2.2.7.2. ¿Cómo puedo invocar al `man` para ver directamente la función `printf` del lenguaje C?
- 2.3. **HELP:** `help` es la ayuda que ofrece el shell de GNU/LINUX para utilizar sus comandos.
 - 2.3.1. ¿Cuál es la diferencia entre `help` e `info`?
 - 2.3.2. ¿Cuál es la diferencia entre `help` y `man`?
 - 2.3.3. ¿Qué sucede al invocar al `help`?
 - 2.3.4. ¿Cómo puedo ver la información de un determinado comando?
- 2.4. **whereis**
 - 2.4.1. ¿Qué sucede al utilizar el comando `whereis cd`?
 - 2.4.2. ¿Qué es la información que se muestra por pantalla al ejecutar el punto anterior?
 - 2.4.3. ¿Qué ocurre si se ejecuta `whereis *` sobre un directorio? (TIP: si no pasa nada, inténtelo nuevamente pero primero ejecute `cd /bin`)
 - 2.4.4. ¿Cuál es la diferencia entre `whereis` y `find`?
- 2.5. **whatis**
 - 2.5.1. ¿Qué sucede al utilizar el comando `whatis cd`?
 - 2.5.2. Si el resultado del punto anterior fue la leyenda "`cd: nothing appropriate`", utilice el comando `/usr/sbin/makewhatis`, y responda los siguientes puntos:
 - 2.5.2.1. ¿Qué realizó la sentencia anterior?
 - 2.5.2.2. Reintente el punto anterior.
 - 2.5.3. Cambie al directorio `/bin`, y ejecute el comando `whatis *` ¿Qué ocurrió?
 - 2.5.4. Utilice el comando `apropos passwd` y `whatis passwd`. Enumere las diferencias encontradas en el resultado de cada uno de los comandos.

3. TECLADO / TERMINALES

- 3.1. ¿Qué sucede si tecleo `cat /e <tab> p <tab>`? (donde `tab` es la tecla tabulación). Presione `<tab>` nuevamente ¿Qué pasó ahora?
- 3.2. ¿Qué sucede si tecleo `cat /e <tab> pas <tab>`?
- 3.3. En este punto analizaremos las distintas terminales que hay en un sistema GNU/Linux. Ejecute los siguientes comandos e indique cuál fue el resultado:
 - 3.3.1. `who`
 - 3.3.2. Presione la tecla `<alt>`, y sin soltarla presione cualquiera de las teclas de función. En la pantalla debería aparecer el login del sistema, de lo contrario, ejecute el paso nuevamente presionando otra tecla de función. Si ya tiene el login del sistema vuelva a conectarse.
 - 3.3.3. Ejecute nuevamente el comando `who`. ¿Qué diferencias encuentra con la primera vez que lo ejecutó?
 - 3.3.4. Ejecute el comando `who am i` ¿qué muestra?, ¿Qué diferencias tiene con el comando ejecutado en el punto anterior?
 - 3.3.5. Repita el paso 3.3.2 y el 3.3.3 hasta que no encuentre ninguna sesión para abrir.
 - 3.3.6. Una vez terminado el punto anterior, Ud. se encontrará sesionado en el sistema como mínimo seis veces. Lo que acaba de hacer es abrir seis terminales virtuales (que podrían ser usadas por distintos usuarios, con diferentes perfiles), en la misma máquina. Así como existen terminales virtuales dentro del mismo equipo, si Ud. cuenta con una red, o con terminales tipo serie, podría abrir tantas sesiones de trabajo como Ud. quiera o necesite. Investigue e indique cómo se denominan los distintos tipos de terminales, y cuáles son los archivos que las representan (tip: busque en el directorio `/dev`).

4. DIRECTORIOS

- 4.1. ¿Para qué se usa el comando `cd`? Ejecute las siguientes variantes de `cd` e indique cuál fue el resultado obtenido:
 - 4.1.1. `cd /`
 - 4.1.2. `cd`
 - 4.1.3. `cd /etc`
 - 4.1.4. `cd..`
 - 4.1.5. `cd ..`
- 4.2. **Bash sobre directorios:**
 - 4.2.1. ¿Cuál/es son las diferencias entre el path absoluto y el path relativo?
 - 4.2.2. ¿Qué es lo que realizan las siguientes operaciones? (tip: si no encuentra la diferencia primero haga `cd /`, y luego vuelva a intentar)
 - 4.2.2.1. `cd ~`
 - 4.2.2.2. `cd -`
 - 4.2.3. ¿Cuál es la diferencia entre `cd ..` y `cd --`?
- 4.3. **Operaciones con directorios:**
 - 4.3.1. ¿Con qué comando se puede crear un directorio?
 - 4.3.2. ¿Con qué comando se puede borrar un directorio?
 - 4.3.3. ¿Qué sucede si el directorio no está vacío?
 - 4.3.4. ¿Cómo puedo salvar la situación anterior? (Sin borrar uno a uno los archivos existentes).

- 4.4. ¿Qué significa la expresión ./ cuando se utiliza delante de un archivo? ¿Para qué sirve?
- 4.5. ¿Cómo puede moverse entre directorios sin utilizar el PATH completo?
- 4.6. ¿Cuál es el contenido de los siguientes directorios que confirman la estructura de cualquier sistema operativo GNU/Linux:?
- 4.6.1. /boot
- 4.6.2. /dev
- 4.6.3. /bin
- 4.6.4. /etc
- 4.6.5. /usr
- 4.6.6. /sbin
- 4.6.7. /root
- 4.6.8. /etc/rc.d (y todos los que están adentro)
- 4.6.9. /proc
- 4.6.10. /mnt
- 4.6.11. /usr/bin
- 4.6.12. /usr/sbin
- 4.6.13. /var
- 4.6.14. /usr/man (y todos los que están adentro)
- 4.6.15. /opt
- 4.6.16. /tmp

5. ARCHIVOS

- 5.1. ¿Qué hacen los siguientes comandos?
- 5.1.1. cp
- 5.1.2. mv
- 5.1.3. rm
- 5.1.4. rcp
- 5.1.5. rsh
- 5.1.6. scp
- 5.1.7. ssh
- 5.2. Para cada comando del punto anterior realice un ejemplo, e indique qué realizó.
- 5.3. ¿Con qué comando puedo concatenar el contenido de dos archivos?.
- 5.3.1. ¿Se puede usar ese comando para otra cosa?.
- 5.4. Haga un ls -l /dev
- 5.4.1. ¿Qué significa el primer carácter?
- 5.4.2. ¿Cuáles son todos los posibles valores que puede contener ese campo y que significa cada uno?
- 5.5. ¿Para qué sirve el comando touch? ¿qué utilidad le encuentra?

6. PERMISOS

- 6.1. Teniendo en cuenta el ls -l anterior, ¿indique que son los siguientes 9 caracteres? (sin considerar el primero sobre el que ya respondió anteriormente)
- 6.2. ¿qué significa cada caracter? ¿cómo están agrupados?
- 6.3. ¿Cómo se asignan los permisos? (detalle los comandos).
- 6.4. ¿Qué son el owner, y el group de un archivo?. ¿Se pueden cambiar?.
- 6.5. Intente cambiar los permisos de un archivo perteneciente al root (sesionado como usuario). Explique qué sucedió.
- 6.6. Explique la forma de cambiar los permisos con valores en octal.
- 6.7. ¿Cuál es el significado de los permisos en los directorios (se debe indicar que indica una r, una w, y una x)?

7. FILTROS

- 7.1. ¿Cuál es la diferencia de los comandos more, less y cat?. De un ejemplo de cada uno.
- 7.2. ¿Cuál es la diferencia entre tail y head?.
- 7.3. ¿Para qué sirve el comando wc y que indican los parámetros -c -l -w? ¿Proponga ejemplos de uso?
- 7.4. ¿Qué es lo que realiza el comando uniq?.
- 7.5. ¿Qué es lo que realiza el comando grep?.
- 7.5.1. ¿Para qué sirve?
- 7.5.2. ¿Qué hace la siguiente línea?: grep root /etc/passwd
- 7.5.3. ¿Qué diferencias encuentra entre la ejecución de los siguientes comandos?:
- 7.5.3.1. grep r /etc/passwd
- 7.5.3.2. grep ^r /etc/passwd
- 7.5.3.3. grep r\$ /etc/passwd

8. VI

- 8.1. Ejecute la siguiente instrucción: vi \$HOME/prueba.txt ¿Qué sucedió?. Ahora ejecute todos los pasos detallados a continuación.
- 8.1.1. Escriba la siguiente frase: "Este es el archivo prueba.txt de <nombre y apellido>"
- 8.1.2. ¿Qué tuvo que hacer para poder escribir la frase?
- 8.1.3. Guarde el archivo, y salga del editor. ¿Qué comando utilizó?
- 8.1.4. Ingrese nuevamente al archivo.
- 8.1.5. Incorpore al inicio del archivo el siguiente párrafo (los acentos puede ser evitados):
"Sistemas Operativos"

Comisión de los días <día de cursada>

Trabajo Práctico 1

Alumno: <su nombre aquí>

Matrícula: <su matrícula aquí>

Documento: <su documento aquí>

- 8.1.6. Describa todos los pasos que tuvo que realizar.
- 8.1.7. Guarde el archivo y continúe la edición. ¿Qué comandos utilizó?
- 8.1.8. Borre la línea de "Matrícula". Indique por lo menos dos formas de realizarlo.
- 8.1.9. Invierta el orden de las líneas "Comisión y TP". No está permitido rescribirlas. ¿Qué comandos utilizó?
- 8.1.10. Ubíquese en la línea 2 (dos) del archivo. No está permitido usar las teclas del cursor, ni el mouse. ¿Qué comando utilizó?
- 8.1.11. Marque para copiar las líneas 2, 3, y 4 (todas juntas, no de a una a la vez). ¿Cómo lo realizó?
- 8.1.12. Ubíquese al final del archivo (sin usar las teclas del cursor), y pegue dos veces el contenido del buffer. ¿Qué comando usó?
- 8.1.13. Deshaga uno de los copiados. No está permitido borrar línea por línea, ni carácter a carácter. ¿Qué comando usó?
- 8.1.14. ¿Cómo busco la palabra "Documento"? ¿Cómo busco la segunda ocurrencia de una palabra?
- 8.1.15. ¿Cómo puedo reemplazar la palabra "Documento" por "Documento." (sin borrar, o realizar el reemplazo a mano)?
- 8.1.16. Guarde el archivo y salga.
- 8.1.17. Ejecutar "vi buscar_reemplazar" e introducir el texto:
1/5/2009 ----- listo
1/5/2010 ----- listo
1/5/2011 ----- listo
1/5/2012 ----- listo
1/5/2013 ----- listo
1/5/2014 ----- listo
1/5/2015 ----- listo
1/5/2016 ----- listo
1/5/2017 ----- listo
1/5/2018 ----- listo
1/5/2019 ----- No listo
- 8.1.18. Ejecutar ":%s/3V/Marzo/g ¿Que paso al ejecutar esto?
- 8.1.19. Si observa el resultado de lo anterior, el cambio fue erróneo, modifique la sentencia para que funcione correctamente.
- 8.1.20. Modifique la fecha para que en lugar del 1 sea el 15. Indique que comandos uso para realizarlo.
- 8.1.21. ¿Indique si existe alguna forma de hacer un buscar y reemplazar pero que antes de realizar la sustitución pregunte?

9. DISCO

- 9.1. ¿Para qué se utiliza el comando mount?. ¿Todos los usuarios lo pueden ejecutar el comando con algunos o todos los parámetros?. En caso de que su respuesta sea negativa, ¿indique cuál /es si?.
- 9.2. Transfiera el archivo a un disquete (el mismo que utilizará para entregar el trabajo práctico, ya que éste archivo es parte de la entrega).
 - 9.2.1. Indique al menos dos formas de realizarlo.
- 9.3. ¿Recordó desmontar el disquete en todas las oportunidades que lo uso, y antes de retirarlo verdad?.
 - 9.3.1. ¿Qué problemas se pueden generar por no realizarlo?.
 - 9.3.2. Repita el punto 1 de éste trabajo práctico, creando un usuario cualquiera (si Ud. se encuentra en el Lab 266, no puede continuar con éste punto).
 - 9.3.3. Cambie de terminal virtual a otra, si se encuentra sesionada salga, e ingrese con el usuario creado en el punto anterior.
 - 9.3.4. Intente desmontar el disquete. ¿Pudo?. Si su respuesta es negativa lo mismo pasará en el laboratorio si Ud. se retira de trabajar sin desmontar la disquetera, y el próximo usuario la quiere utilizar. Por ese motivo en el laboratorio al hacer el logout del sistema se ejecuta un script que verifica si la disquetera está montada. En caso de estarlo, la desmonta, y además envía un alerta administrativo a los administradores de la red. Al tercer alerta administrativo que se genere se le bloqueará la cuenta por un período de 15 días.
 - 9.3.5. ¿De qué manera nombra el sistema a cada unidad de disco?
 - 9.3.6. ¿Cómo identifica Ud. a qué unidad se hace referencia?
 - 9.3.7. ¿Podría Ud. indicar en que unidad y partición se encuentra instalado el GNU/Linux en su computadora? ¿Qué comandos o archivos de información utilizó?

10. VARIABLES DE ENTORNO

- 10.1. ¿Qué son las variables de entorno y para qué sirven?.
 - 10.1.1. Escriba el contenido y explique el significado de las siguientes variables: HOME / LOGNAME / PATH / HOSTNAME / IFS
 - 10.1.2. ¿Qué comando usó para ver el contenido de las variables del punto anterior?
 - 10.1.3. Cree una variable de entorno HOLA que contenga el mensaje "Hola mundo".

- 10.1.4. ¿Cuál es el uso que le da el sistema a la variable PATH? ¿Qué ocurre si intenta ejecutar un comando que no se encuentra ubicado en alguno de los directorios que contiene la variable? ¿Cómo lo soluciona?
- 10.1.5. ¿Por qué existen las variables PS1 y PS2? ¿Qué es un comando multilínea?

11. PLACA DE RED (En caso de no tener en su máquina, realizarlo en el Lab266)

- 11.1. ¿Para qué sirve el comando ifconfig y en qué directorio se encuentra?
- 11.2. ¿Qué IP o IP's tiene asignada la computadora?
- 11.3. ¿Qué es el adaptador de red y para qué se utiliza?
- 11.4. ¿Cuál es la salida del comando ping -c4 (ip del eth0)?

Trabajo Práctico Nro. 1 (GRUPAL):

- **Tema:** Programación de scripts en tecnología bash
 - **Descripción:** Se programarán todos los scripts mencionados en el presente trabajo, teniendo especialmente en cuenta las recomendaciones sobre programación mencionadas en la introducción de este trabajo.
 - **Formato de entrega:** Siguiendo el protocolo especificado anteriormente. Recomendamos realizar la entrega presencial
 - **Documentación:** Todos los scripts que se entreguen deben tener un encabezado y un fin de archivo. Dentro del encabezado deben figurar el nombre del script, el trabajo práctico al que pertenece y el número de ejercicio dentro del trabajo práctico al que corresponde, el nombre de cada uno de los integrantes detallando nombre y apellido y el número de DNI de cada uno (tenga en cuenta que para pasar la nota final del trabajo práctico será usada dicha información, y no se le asignará la nota a ningún alumno que no figure en todos los archivos con todos sus datos), también deberá indicar el número de entrega a la que corresponde (entrega, primera reentrega, segunda reentrega, etc.).
 - **Evaluación:** Luego de entregado el trabajo práctico los ayudantes procederán a evaluar los ejercicios resueltos, en caso de encontrar errores se documentará en la carátula del TP que será devuelta al grupo con la evaluación final del TP y una fecha de reentrega en caso de ser necesaria (en caso de no cumplir con dicha fecha de reentrega el trabajo práctico será desaprobado). Cada ayudante podrá determinar si un determinado grupo debe o no rendir coloquio sobre el trabajo práctico presentado.
- Las notas sobre los trabajos también estarán disponibles en el sitio de la cátedra (www.sisop.com.ar) donde además del estado de la corrección se podrá ver un detalle de las pruebas realizadas y los defectos encontrados
- Importante:** Cada ejercicio cuenta con una lista de validaciones mínimas que se realizará, esto no implica que se puedan hacer otras validaciones al momento de evaluar el trabajo presentado.
- **Fecha de entrega: 29/04/2020 o 30/04/2020**, dependiendo del día que se curse la materia.

Introducción:

La finalidad del presente práctico es que los alumnos adquieran un cierto entrenamiento sobre la programación de shell scripts, practicando el uso de utilitarios comunes provistos por los sistemas operativos (GNU/Linux). Todos los scripts, están orientados a la administración de una máquina, o red y pueden ser interrelacionados de tal manera de darles una funcionalidad real.

Cabe destacar que todos los scripts deben poder ser ejecutados en forma batch o interactiva, y que en ningún caso serán probados con el usuario root, por lo cual deben tener en cuenta al realizarlos, no intentar utilizar comandos, directorios, u otros recursos que solo están disponibles para dicho usuario. Todos los scripts deberán funcionar en las instalaciones del laboratorio 266, ya que es ahí donde serán controlados por el grupo docente.

Para un correcto y uniforme funcionamiento, todos ellos deberán respetar algunos lineamientos generales indicados en el trabajo práctico anterior.

Ejercicios:

Tip: En caso de tener problemas con un script bajado desde un dispositivo formateado con DOS, y que contiene ^M al final de cada línea puede usar el siguiente comando para eliminarlos:

```
tr -d '\r' <archivo_con_M >archivo_sin_M
```

En caso de ejecutar scripts que usen el archivo de passwords, en el laboratorio, debe cambiar "cat /etc/passwd" por "getent passwd"

Ejercicio 1:

Tomando en cuenta el siguiente script responda las preguntas que se encuentran más abajo. **Importante:** como parte del resultado se deberá entregar el script en un archivo tipo sh y las respuestas en el mismo código.

```
#!/bin/bash
ErrorS()
{
    echo "Error. La sintaxis del script es la siguiente:"
    echo ".....: $0 nombre_archivo L" # COMPLETAR
    echo ".....: $0 nombre_archivo C" # COMPLETAR
    echo ".....: $0 nombre_archivo M" # COMPLETAR
}

ErrorP()
{
    echo "Error. nombre_archivo ....." # COMPLETAR
}

if test $# -lt 2; then
    ErrorS
fi
if !test $1 -r; then
    ErrorP
elif test -f $1 && (test $2 = "L" || test $2 = "C" || test $2 = "M"); then
    if test $2 = "L" then
        res=`wc -l $1`
        echo ".....: $res" # COMPLETAR
    elif test $2 = "C"; then
        res=`wc -m $1`
        echo ".....: $res" # COMPLETAR
    elif test $2 = "M"; then
        res=`wc -L $1`
        echo ".....: $res" # COMPLETAR
    fi
else
    ErrorS
fi
```

Responda:

- ¿Cuál es el objetivo de este script?
- ¿Qué parámetros recibe?
- Comentar el código según la funcionalidad (no describa los comandos, indique la lógica)
- Completar los "echo" con el mensaje correspondiente.
- ¿Qué información brinda la variable "\$#"? ¿Qué otras variables similares conocen? Explíquelas.
- Explique las diferencias entre los distintos tipos de comillas que se pueden utilizar en Shell scripts.

Ejercicio 2:

Se cuenta con un archivo de log, donde se registraron todas las llamadas realizadas en una semana por un call center, donde se detalla el inicio y fin de las mismas. Estos registros tienen el siguiente formato, donde se indica quién realizó la llamada (usuario). Se debe considerar el primer registro como el inicio de la llamada y la siguiente como la finalización. Tener en cuenta que el proceso que registra las llamadas tiene problemas de sincronización, por lo que los registros no se encuentran ordenados.

Se pide obtener y mostrar por pantalla los siguientes datos uno debajo del otro:

- Promedio de tiempo de las llamadas realizadas por día.
- Promedio de tiempo y cantidad por usuario por día.
- Los 3 usuarios con más llamadas en la semana.
- Cuántas llamadas no superan la media de tiempo por día y el usuario que tiene más llamadas por debajo de la media en la semana.

Parámetros:

- -f "path": Directorio en el que se encuentran los archivos de log. Puede ser una ruta relativa o absoluta. No se debe realizar búsqueda recursiva dentro del directorio.

Formato del registro de llamada:

- Fecha y Hora: Formato YYYY-MM-DD HH:mm:ss
- Usuario que realiza la llamada.
- Separador: “ _ ” (espacio, guión bajo, espacio)

Ejemplo de registros: ((*inicio*) y (*fin*)) son a modo de ejemplo, no son parte del archivo)

(*inicio*) 2020-03-09 14:22:00 _ aerodriguez
 (*inicio*) 2020-03-09 14:22:10 _ fmarino
 (*inicio*) 2020-03-09 14:22:11 _ vbo
 (*fin*) 2020-03-09 14:25:00 _ fmarino
 (*fin*) 2020-03-09 14:25:10 _ aerodriguez
 (*inicio*) 2020-03-09 14:26:40 _ aerodriguez
 (*fin*) 2020-03-09 14:26:41 _ vbo
 (*fin*) 2020-03-09 14:32:00 _ aerodriguez

Criterios de corrección:

Control	Criticidad
Debe cumplir con el enunciado	Obligatorio
El script cuenta con una ayuda (-?, -h, --help)	Obligatorio
Validación correcta de los parámetros	Obligatorio
Proveer archivos de ejemplo para realizar pruebas	Obligatorio

Ejercicio 3:

Además de problemas de sincronización, el sistema que maneja los logs del call center tiene problemas de almacenamiento. Esto se debe a que se crea un archivo de log por cada semana y empresa que contrata el servicio. Tratando de encontrarle una solución a este problema, Fernando, un administrador de sistemas, tuvo una idea: crear un script que se encargue de eliminar automáticamente los logs viejos cada vez que se crea un archivo nuevo.

Una vez creado un nuevo archivo, se deben evaluar todos los archivos del directorio y dejar solamente los de la última semana de cada empresa. Los archivos siguen la siguiente convención para los nombres: nombreEmpresa-númeroSemana.log. Por ejemplo, si en el directorio a evaluar, luego de creado el nuevo archivo existen los siguientes archivos, se deberían borrar los indicados:

```
unlam-11.log (archivo a eliminar)
unlam-12.log (archivo creado)
personal-11.log
movistar-11.log (archivo a eliminar)
movistar-12.log
```

Tener en cuenta que los archivos a eliminar deben cumplir con la convención de nombres mencionada anteriormente.

El script debe recibir los siguientes parámetros:

- -f "path": Directorio en el que se encuentran los archivos de log. Puede ser una ruta relativa o absoluta. No se debe realizar búsqueda recursiva dentro del directorio.
- -t "segundos": Intervalo de tiempo a evaluar el directorio.

Nota: Para controlar cuando se crea un archivo nuevo, es necesario disponer de un proceso que evalúe el directorio cada determinado tiempo y realice las acciones necesarias.

Criterios de corrección:

Control	Criticidad
Debe cumplir con el enunciado	Obligatorio
El script cuenta con una ayuda (-?, -h, --help)	Obligatorio
Validación correcta de los parámetros	Obligatorio
Crear un demonio para controlar la creación del archivo	Obligatorio
Proveer archivos de ejemplo para realizar pruebas	Obligatorio
Usar expresiones regulares para validar que el nombre del archivo cumple con nombreEmpresa-númeroSemana.log	Obligatorio

Ejercicio 4:

Gonzalo, otro administrador de sistemas del call center decidió encarar el problema de almacenamiento de otra forma: comprimir los archivos de log antes de borrarlos, de esta manera no se pierden los archivos en caso de querer consultarlos más adelante y se ahorra espacio hasta que la gerencia tome la decisión de destinar presupuesto a la compra de nuevos discos rígidos. Para ahorrar trabajo, decidió tomar el script de Fernando y modificarlo.

A diferencia del script de Fernando, este script se tiene que ejecutar manualmente ya que no utiliza eventos para saber cuándo se creó un archivo nuevo. Tener en cuenta que los nuevos archivos se deben agregar al ZIP existente para no perder los ya procesados.

El script debe recibir los siguientes parámetros:

- -f "path": Directorio en el que se encuentran los archivos de log. Puede ser una ruta relativa o absoluta. No se debe realizar búsqueda recursiva dentro del directorio.
- -z "path": Directorio en el que se generarán los archivos comprimidos de los clientes.
- -e "empresa": Nombre de la empresa a procesar. (opcional)

Criterios de corrección:

Control	Criticidad
Debe cumplir con el enunciado	Obligatorio
El script cuenta con una ayuda (-?, -h, --help)	Obligatorio
Validación correcta de los parámetros	Obligatorio
Proveer archivos de ejemplo para realizar pruebas	Obligatorio

Ejercicio 5:

Por un requerimiento gubernamental, una universidad pública tiene la obligación de obtener estadísticas de aprobación y deserción de su alumnado. El sistema que mantiene el registro de las notas de los alumnos tiene la opción de exportarlas con el siguiente formato:

```
DNI|IdMateria|PrimerParcial|SegundoParcial|RecuParcial|RecuNota|Final
42736222|1|4|4|||
42736222|2|7|1|4|8
40586527|1||2|||
39873564|5|9|10|||
```

Para obtener dichas estadísticas, la universidad convoca a los alumnos de Sistemas Operativos a desarrollar un script que analice el archivo exportado y genere un archivo con los siguientes datos requeridos por el gobierno agrupados por materia:

- Cantidad de alumnos aptos para rendir final (sin final dado y notas en parciales/recuperatorio entre 4 y 6 inclusive).
- Cantidad de alumnos que recursarán (notas menor a 4 en final o en parciales y/o recuperatorio).
- Cantidad de alumnos con posibilidad de rendir recuperatorio (sin recuperatorio rendido y al menos una nota de parcial menor a 7).
- Cantidad de alumnos que abandonaron la materia (sin nota en al menos un parcial y sin recuperatorio rendido para dicho parcial).

El script debe recibir los siguientes parámetros:

- -f "archivo": Ruta del archivo a procesar. Puede ser una ruta relativa o absoluta.

El formato del archivo de salida debe ser el siguiente:

```
"Materia","Final","Recursan","Recuperan","Abandonaron"
"1","1","2","0","0"
"2","6","3","7","0"
"3","5","6","0","10"
```

Criterios de corrección:

Control	Criticidad
Debe cumplir con el enunciado	Obligatorio
El script cuenta con una ayuda (-?, -h, --help)	Obligatorio
Validación correcta de los parámetros	Obligatorio
Proveer archivos de ejemplo para realizar pruebas	Obligatorio

Ejercicio 6:

Realizar un script que permita realizar la suma de todos los números fraccionarios contenidos en un archivo. Dichos números se encuentran separados por coma como se ve en el siguiente ejemplo:

16/8,8/9,1:5/8,9/2,-7/5

Tener en cuenta que:

- Los números fraccionarios pueden ser negativos.

- Los números pueden estar expresados en notación mixta, es decir, un número entero seguido de un número fraccionario. En el ejemplo anterior: $1:5/8$, equivale a $13/8$.

El resultado debe mostrarse por pantalla expresado como una fracción simplificada (el menor denominador posible que mantenga al numerador como número entero). Además, este resultado debe guardarse en un archivo ubicado en el mismo directorio del script que se llame `salida.out`. Si el archivo ya existe, sobrescribirlo.

Parámetros:

- -f "path": ruta del archivo de entrada ya sea de manera relativa o absoluta.

Nota: se asume que el formato del archivo es siempre correcto. Archivos vacíos o con un solo número fraccionario se consideran como archivos válidos.

Criterios de corrección:

Control	Criticidad
Debe cumplir con el enunciado	Obligatorio
El script cuenta con una ayuda (-?, -h, --help)	Obligatorio
Validación correcta de los parámetros	Obligatorio
Validación del archivo de entrada. Debe ser un archivo válido	Obligatorio
Proveer archivos de ejemplo para realizar pruebas	Obligatorio

Trabajo Práctico Nro. 2 (GRUPAL):

- **Tema:** Programación de scripts básicos en PowerShell
- **Descripción:** Se programarán todos los scripts mencionados en el presente trabajo, teniendo especialmente en cuenta las recomendaciones sobre programación mencionadas en la introducción de este trabajo. Los contenidos de este trabajo práctico pueden ser incluidos como tema de parciales.
- **Formato de entrega:** Según el protocolo especificado anteriormente. Recomendamos realizar la entrega presencial
- **Documentación:** Todos los scripts que se entreguen deben tener un encabezado y un fin de archivo. Dentro del encabezado deben figurar el nombre del script, el trabajo práctico al que pertenece y el número de ejercicio dentro del trabajo práctico al que corresponde, el nombre de cada uno de los integrantes detallando nombre y apellido y el número de DNI de cada uno (tenga en cuenta que para pasar la nota final del trabajo práctico será usada dicha información, y no se le asignará la nota a ningún alumno que no figure en todos los archivos con todos sus datos), también deberá indicar el número de entrega a la que corresponde (entrega, primera reentrega, segunda reentrega, etc.).
- **Evaluación:** Luego de entregado el trabajo práctico los ayudantes procederán a evaluar los ejercicios resueltos, en caso de encontrar errores se documentará en la carátula del TP que será devuelta al grupo con la evaluación final del TP y una fecha de reentrega en caso de ser necesaria (en caso de no cumplir con dicha fecha de reentrega el trabajo práctico será desaprobado). Cada ayudante podrá determinar si un determinado grupo debe o no rendir coloquio sobre el trabajo práctico presentado.
Las notas sobre los trabajos también estarán disponibles en el sitio de la cátedra (www.sisop.com.ar) donde además del estado de la corrección se podrá ver un detalle de las pruebas realizadas y los defectos encontrados.
Importante: Cada ejercicio cuenta con una lista de validaciones mínimas que se realizará, esto no implica que se puedan hacer otras validaciones al momento de evaluar el trabajo presentado
- **Fecha de entrega:** 20/05/2020 o 21/05/2020, dependiendo del día que se curse la materia.

Introducción:

La finalidad del presente práctico es que los alumnos adquieran un cierto entrenamiento sobre la programación de shell scripts en lenguaje PowerShell. Todos los scripts están orientados a la administración de una máquina, o red y pueden ser interrelacionados de tal manera de darles una funcionalidad real.

Cabe destacar que todos los scripts deben poder ser ejecutados en forma batch o interactiva, y que en ningún caso serán probados con el usuario administrator, por lo cual deben tener en cuenta al realizarlos, no intentar utilizar comandos, directorios, u otros recursos que solo están disponibles para dicho usuario, o usuarios del grupo. Todos los scripts deberán funcionar en las instalaciones del laboratorio 266, ya que es ahí donde serán controlados por el grupo docente.

Para un correcto y uniforme funcionamiento, se deberán respetar algunos lineamientos generales:

1. Modularidad:

Si bien es algo subjetivo del programador, se trata de privilegiar la utilización de funciones (internas / externas), para lograr una integración posterior menos trabajosa.

2. Claridad:

Se recomienda fuertemente el uso de comentarios que permitan la máxima legibilidad de los scripts.

3. Verificación:

Todos los scripts deben realizar un control de las opciones que se le indiquen por línea de comando, es decir, verificar la sintaxis de la misma. En caso de error u omisión de opciones, al estilo de la mayoría de los comandos deben indicar mensajes como:

```
"Error en llamada!  
Uso: comando [...]"
```

Donde se indicará entre corchetes [], los parámetros opcionales; y sin ellos los obligatorios, dando una breve explicación de cada uno de ellos.

Todos los scripts deben incluir una opción standard '-?' que indique el número de versión y las formas de llamada.

Ejercicios:

IMPORTANTE: la versión mínima a utilizar para confeccionar y evaluar este TP debe ser Powershell 6, ya que es la versión de Powershell Core donde se realizará la corrección

Ejercicio 1:

En base al siguiente código, responder:

```
Param (
    [Parameter(Position = 1, Mandatory = $false)]
    [String] $pathsalida = ".\procesos.txt ",
    [int] $cantidad = 3
)
$existe = Test-Path $pathsalida
if ($existe -eq $true) {
    $listaproceso = Get-Process
    foreach ($proceso in $listaproceso) {
        $proceso | Format-List -Property Id,Name >> $pathsalida
    }
    for ($i = 0; $i -lt $cantidad ; $i++) {
        Write-Host $listaproceso[$i].Name - $listaproceso[$i].Id
    }
} else {
    Write-Host "El path no existe"
}
```

Responder:

1. ¿Cuál es el objetivo del script?
2. ¿Agregaría alguna otra validación a los parámetros?
3. ¿Qué sucede si se ejecuta el script sin ningún parámetro?

Ejercicio 2:

Se cuenta con un archivo de log, donde se registraron todas las llamadas realizadas en una semana por un call center, donde se detalla el inicio y fin de las mismas. Estos registros tienen el siguiente formato, donde se indica quién realizó la llamada (usuario). Se debe considerar el primer registro como el inicio de la llamada y la siguiente como la finalización. Tener en cuenta que el proceso que registra las llamadas tiene problemas de sincronización, por lo que los registros no se encuentran ordenados.

Se pide obtener y mostrar por pantalla los siguientes datos uno debajo del otro:

- Promedio de tiempo de las llamadas realizadas por día.
- Promedio de tiempo y cantidad por usuario por día.
- Los 3 usuarios con más llamadas en la semana.
- Cuántas llamadas no superan la media de tiempo por día y el usuario que tiene más llamadas por debajo de la media en la semana.

Parámetros:

- -Path: Directorio en el que se encuentran los archivos de log. Puede ser una ruta relativa o absoluta. No se debe realizar búsqueda recursiva dentro del directorio.

Formato del registro de llamada:

- Fecha y Hora: Formato YYYY-MM-DD HH:mm:ss
- Usuario que realiza la llamada.
- Separador: " _ " (espacio, guión bajo, espacio)

Ejemplo de registros: ((**inicio**) y (**fin**)) son a modo de ejemplo, no son parte del archivo)

(**inicio**) 2020-03-09 14:22:00 _ aerodriguez

(**inicio**) 2020-03-09 14:22:10 _ fmarino

(**inicio**) 2020-03-09 14:22:11 _ vbo

(fin) 2020-03-09 14:25:00 _ fmarino
 (fin) 2020-03-09 14:25:10 _ aerodriguez
 (inicio) 2020-03-09 14:26:40 _ aerodriguez
 (fin) 2020-03-09 14:26:41 _ vbo
 (fin) 2020-03-09 14:32:00 _ aerodriguez

Criterios de corrección:

Control	Criticidad
Debe cumplir con el enunciado	Obligatorio
El script cuenta con una ayuda visible con Get-Help	Obligatorio
Validación correcta de los parámetros (Param)	Obligatorio
Proveer archivos de ejemplo para realizar pruebas	Obligatorio
Utilizar Format-List para formatear la salida por pantalla	Obligatorio

Ejercicio 3:

Además de problemas de sincronización, el sistema que maneja los logs del call center tiene problemas de almacenamiento. Esto se debe a que se crea un archivo de log por cada semana y empresa que contrata el servicio. Tratando de encontrarle una solución a este problema, Fernando, un administrador de sistemas, tuvo una idea: crear un script que se encargue de eliminar automáticamente los logs viejos cada vez que se crea un archivo nuevo.

Una vez creado un nuevo archivo, se deben evaluar todos los archivos del directorio y dejar solamente los de la última semana de cada empresa. Los archivos siguen la siguiente convención para los nombres: nombreEmpresa-númeroSemana.log. Por ejemplo, si en el directorio a evaluar, luego de creado el nuevo archivo existen los siguientes archivos, se deberían borrar los indicados:

```

unlam-11.log (archivo a eliminar)
unlam-12.log (archivo creado)
personal-11.log
movistar-11.log (archivo a eliminar)
movistar-12.log
  
```

Tener en cuenta que los archivos a eliminar deben cumplir con la convención de nombres mencionada anteriormente.

El script debe recibir los siguientes parámetros:

- Directorio: Directorio en el que se encuentran los archivos de log. Puede ser una ruta relativa o absoluta. No se debe realizar búsqueda recursiva dentro del directorio.

Nota: Para controlar cuando se crea un archivo nuevo, utilizar FileSystemWatcher y el evento correspondiente a creación de archivos.

Criterios de corrección:

Control	Criticidad
Debe cumplir con el enunciado	Obligatorio
El script cuenta con una ayuda visible con Get-Help	Obligatorio
Validación correcta de los parámetros (Param)	Obligatorio
Proveer archivos de ejemplo para realizar pruebas	Obligatorio
Utilizar los eventos de FileSystemWatcher	Obligatorio
Usar expresiones regulares para validar que el nombre del archivo cumple con nombreEmpresa-númeroSemana.log	Obligatorio

Ejercicio 4:

Gonzalo, otro administrador de sistemas del call center decidió encarar el problema de almacenamiento de otra forma: comprimir los archivos de log antes de borrarlos, de esta manera no se pierden los archivos en caso de querer consultarlos más adelante y se ahorra espacio hasta que la gerencia tome la decisión de destinar presupuesto a la compra de nuevos discos rígidos. Para ahorrar trabajo, decidió tomar el script de Fernando y modificarlo.

A diferencia del script de Fernando, este script se tiene que ejecutar manualmente ya que no utiliza eventos para saber cuándo se creó un archivo nuevo. Tener en cuenta que los nuevos archivos se deben agregar al ZIP existente para no perder los ya procesados.

El script debe recibir los siguientes parámetros:

- -Directorio: Directorio en el que se encuentran los archivos de log. Puede ser una ruta relativa o absoluta. No se debe realizar búsqueda recursiva dentro del directorio.
- -DirectorioZip: Directorio en el que se generarán los archivos comprimidos de los clientes.
- -Empresa: Nombre de la empresa a procesar. (opcional)

Criterios de corrección:

Control	Criticidad
Debe cumplir con el enunciado	Obligatorio
El script cuenta con una ayuda visible con Get-Help	Obligatorio
Validación correcta de los parámetros (Param)	Obligatorio
Proveer archivos de ejemplo para realizar pruebas	Obligatorio
Utilizar el cmdlet Compress-Archive	Obligatorio

Ejercicio 5:

Por un requerimiento gubernamental, una universidad pública tiene la obligación de obtener estadísticas de aprobación y deserción de su alumnado. El sistema que mantiene el registro de las notas de los alumnos tiene la opción de exportarlas con el siguiente formato:

```
DNI|IdMateria|PrimerParcial|SegundoParcial|RecuParcial|RecuNota|Final
42736222|1|4|4|||
42736222|2|7|1|4|8
40586527|1||2|||
39873564|5|9|10|||
```

Para obtener dichas estadísticas, la universidad convoca a los alumnos de Sistemas Operativos a desarrollar un script que analice el archivo exportado y genere un archivo con los siguientes datos requeridos por el gobierno agrupados por materia:

- Cantidad de alumnos aptos para rendir final (sin final dado y notas en parciales/recuperatorio entre 4 y 6 inclusive).
- Cantidad de alumnos que recursarán (notas menor a 4 en final o en parciales y/o recuperatorio).
- Cantidad de alumnos con posibilidad de rendir recuperatorio (sin recuperatorio rendido y al menos una nota de parcial menor a 7).
- Cantidad de alumnos que abandonaron la materia (sin nota en al menos un parcial y sin recuperatorio rendido para dicho parcial).

El script debe recibir los siguientes parámetros:

- -Nomina: Ruta del archivo a procesar. Puede ser una ruta relativa o absoluta.

El formato del archivo de salida debe ser el siguiente:

```
"Materia","Final","Recursan","Recuperan","Abandonaron"
"1","1","2","0","0"
"2","6","3","7","0"
"3","5","6","0","10"
```

Criterios de corrección:

Control	Criticidad
Debe cumplir con el enunciado	Obligatorio
El script cuenta con una ayuda visible con Get-Help	Obligatorio
Validación correcta de los parámetros (Param)	Obligatorio
Proveer archivos de ejemplo para realizar pruebas	Obligatorio
Utilizar Import-Csv y Export-Csv para el manejo de los archivos	Obligatorio

Ejercicio 6:

Realizar un script que permita realizar la suma de todos los números fraccionarios contenidos en un archivo. Dichos números se encuentran separados por coma como se ve en el siguiente ejemplo:

16/8,8/9,1:5/8,9/2,-7/5

Tener en cuenta que:

- Los número fraccionarios pueden ser negativos.
- Los números pueden estar expresados en notación mixta, es decir, un número entero seguido de un número fraccionario. En el ejemplo anterior: 1:5/8, equivale a 13/8.

El resultado debe mostrarse por pantalla expresado como una fracción simplificada (el menor denominador posible que mantenga al numerador como número entero). Además, este resultado debe guardarse en un archivo ubicado en el mismo directorio del script que se llame `salida.out`. Si el archivo ya existe, sobrescribirlo.

Parámetros:

- -Path: Ruta del archivo de entrada ya sea de manera relativa o absoluta.

Nota: se asume que el formato del archivo es siempre correcto. Archivos vacíos o con un solo número fraccionario se consideran como archivos válidos.

Criterios de corrección:

Control	Criticidad
Debe cumplir con el enunciado	Obligatorio
El script cuenta con una ayuda visible con Get-Help	Obligatorio
Validación correcta de los parámetros (Param)	Obligatorio
Validación del archivo de entrada. Debe ser un archivo válido	Obligatorio
Proveer archivos de ejemplo para realizar pruebas	Obligatorio

Trabajo Práctico Nro. 3 (GRUPAL):

- **Tema:** Procesos, comunicación y sincronización
- **Descripción:** Codificar todos los programas mencionados en el presente trabajo, teniendo especialmente en cuenta las recomendaciones sobre programación mencionadas en la introducción de este trabajo
- **Formato de entrega:** Siguiendo el protocolo especificado anteriormente. Recomendamos realizar la entrega presencial
- **Documentación:** Todos los programas que se entreguen deben tener un encabezado y un fin de archivo. Dentro del encabezado deben figurar el nombre del programa, el trabajo práctico al que pertenece y el número de ejercicio dentro del trabajo práctico al que corresponde, el nombre de cada uno de los integrantes detallando nombre y apellido y el número de DNI de cada uno (tenga en cuenta que para pasar la nota final del trabajo práctico será usada dicha información, y no se le asignará la nota a ningún alumno que no figure en todos los archivos con todos sus datos), también deberá indicar el número de entrega a la que corresponde (entrega, primera reentrega, segunda reentrega, etc.).
- **Evaluación:** Luego de entregado el trabajo práctico los ayudantes procederán a evaluar los ejercicios resueltos, en caso de encontrar errores se documentará en la carátula del TP que será devuelta al grupo con la evaluación final del TP y una fecha de reentrega en caso de ser necesaria (en caso de no cumplir con dicha fecha de reentrega el trabajo práctico será desaprobado). Cada ayudante podrá determinar si un determinado grupo debe o no rendir coloquio sobre el trabajo práctico presentado.

Las notas sobre los trabajos también estarán disponibles en el sitio de la cátedra (www.sisop.com.ar) donde además del estado de la corrección se podrá ver un detalle de las pruebas realizadas y los defectos encontrados

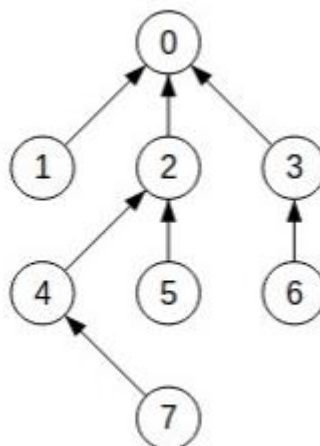
Importante: Cada ejercicio cuenta con una lista de validaciones mínimas que se realizará, esto no implica que se puedan hacer otras validaciones al momento de evaluar el trabajo presentado.

- **Fecha de entrega: 17/06/2020 o 18/06/2020**, dependiendo del día que se curse la materia.

Ejercicios:

Ejercicio 1:

Se solicita desarrollar un programa que dado un parámetro N de entrada recree el siguiente árbol de procesos. Cada proceso deberá imprimir su pid y el de toda su ascendencia. Ejemplo el proceso 7 deberá imprimir 7(PID) – 4(PID) – 2(PID) – 0(PID).



Adicionalmente, el proceso representado con PID 7 en el grafo, deberá crear N jerarquías de procesos hijos adicionales.
Por ejemplo:

Si $N=2$, deberá crear un proceso hijo que imprima por pantalla $8(PID) - 7(PID) - 4(PID) - 2(PID) - 0(PID)$ y a su vez este crear otro proceso hijo que muestre: $9(PID) - 8(PID) - 7(PID) - 4(PID) - 2(PID) - 0(PID)$.

Criterios de corrección:

Control	Criticidad
Compila sin errores con el makefile entregado	Obligatorio
Funciona correctamente según enunciado	Obligatorio
Existe algún tipo de ayuda para ejecutar el proceso	Obligatorio
Utiliza la función fork para crear procesos	Obligatorio
Valida que el parámetro de entrada sea un número entero no negativo	Obligatorio

Ejercicio 2:

Un determinado grupo de investigadores lo han contratado para desarrollar una función matemática que ejecute en varios Threads. La misma trabaja con la sucesión de Fibonacci.

El flujo del programa es el siguiente:

Primero recibirá por parámetro un número N y creará N thread donde cada uno ejecutará el algoritmo matemático mencionado bajo la siguiente lógica.

Supongamos que $N=3$

El primer Thread deberá sumar los 3 primeros números de Fibonacci (en este caso $1+1+2$)

El segundo Thread deberá sumar los 2 primeros números de Fibonacci (en este caso $1+1$)

El tercer Thread deberá devolver el primer número de la sucesión de Fibonacci (en este caso 1)

(los resultados lo irán almacenando en un array global donde cada thread tendrá una posición asignada para no pisar data)

Una vez finalizado el proceso se generarán 2 nuevos threads, el primero sumará todos los elementos del array (en este caso $4+2+1$) mientras que el segundo los multiplicará ($4*2*1$). La información será almacenada nuevamente en un vector global, en este caso de 2 elementos.

Por último, se restará el resultado del producto con el resultado de la suma, el valor final deberá ser mostrado por pantalla.

Criterios de corrección:

Control	Criticidad
Utilizar la biblioteca thread de C++	Obligatorio
Compila sin errores con el makefile entregado	Obligatorio
Funciona correctamente según enunciado	Obligatorio
Existe algún tipo de ayuda para ejecutar el proceso	Obligatorio
Valida que el parámetro de entrada sea un número entero mayor a 0	Obligatorio
Utiliza una clase hiloFibonacci cuyo constructor reciba el número N a calcular en la sucesión	Opcional

Ejercicio 3:

El club de barrio "La Juanita" está teniendo problemas para llevar a cabo la correcta contabilidad del pago de las cuotas de sus asociados, así como el control de la correcta asistencia en los días correspondientes

debido al incremento de socios que se ha dado en el último año. Para automatizar de la forma más económica posible lo contactan a usted para brindar solución a dicha problemática.

El club mantiene los siguientes archivos (Todos los archivos separan registros por punto y coma (;)):

- Socios: contiene: Nombre;Apellido;DNI;NombreDelDeporte;DíaAsistencia. Por ahora no está permitido anotarse a más de un deporte y además solo se puede asistir un día a la semana.
 - El campo nombreDelDeporte puede tomar los valores: "Fútbol", "Voley", "Basquet" y "Natación" (respetando las minúsculas y mayúsculas)
 - El campo Día puede tomar los valores "Lunes, Martes, Miércoles, Jueves, Viernes, Sabado y Domingo"
- Pagos: Archivo separado por punto y coma que posee el siguiente formato: DNI;Fecha. Si el pago se realiza del 1 al 10 se realiza un descuento del 10%.
 - Formato de fecha: YYYY-MM-DD
- Asistencia: Archivo separado por Punto y Coma que posee el siguiente formato: DNI;DiaAsistencia
 - El campo DiaAsistencia toma los mismos valores que el del archivo Socios

En cuanto a los deportes que se practican, se tiene la siguiente información (no se encuentra en ningún archivo, puede estar como valores en el código):

Deported	Días posibles	Valor de la cuota
Fútbol	Lunes - Miércoles	\$ 1.000
Vóley	Martes - Jueves	\$ 1.200
Básquet	Viernes	\$ 1.300
Natación	Sábado	\$ 1.800

Lo que necesita principalmente el club es:

1. Calcular automáticamente el monto total cobrado en el mes (contemplando los descuentos).
2. Detectar asociados que no hayan pagado la cuota mensual.
3. Detectar asistencias en días que no corresponden.

Dado que en el club utilizan Linux como sistema operativo, después de la etapa de análisis con el personal

del club, se definen los siguientes lineamientos:

1. Habrá dos procesos productores, uno leerá la información del archivo Pagos y el otro del archivo Asistencia, ambos enviarán la información a través de memoria compartida a un proceso consumidor, que será el encargado de realizar el proceso correspondiente para responder a las necesidades del club, mostrando los resultados por pantalla.
2. Tiene que existir alternancia estricta en el envío de información entre los procesos productores.

Criterios de corrección:

Control	Criticidad
Compila sin errores con el makefile entregado	Obligatorio
Funciona correctamente según enunciado	Obligatorio
Existe algún tipo de ayuda para la ejecución de los procesos	Obligatorio
Utiliza semáforos Posix y Memoria Compartida	Obligatorio
Cierra correctamente los recursos utilizados	Obligatorio
Finalizan correctamente todos los procesos	Obligatorio
No hay pérdida de información	Obligatorio

Ejercicio 4:

En una PyME de desarrollo de software, el área de testing necesita de una herramienta que pueda mínimamente detectar “fugas” de memoria, así como excesiva utilización de CPU, para esto le solicita a usted, pasante de la empresa, que implemente dicha herramienta siguiendo con las especificaciones de diseño que fueron realizadas por el líder del equipo de testing;

1. Codificar un proceso Principal cuya tarea será crear dos procesos hijos (Control y Registro), los tres procesos deberán quedar ejecutando en segundo plano. Una vez creados ambos procesos hijos, el proceso Principal deberá quedar a la espera de la señal SIGUSR1, cuando se reciba dicha señal los tres procesos deberán finalizar.
2. El proceso Control deberá detectar (cada un segundo) si algún proceso en ejecución supera un valor límite de consumo de memoria o CPU, si dicha situación ocurre deberá enviar al proceso Registro un conjunto de datos a través de un FIFO.

Los datos a enviar son:

- PID
- Nombre
- Tipo de exceso (Memoria – CPU – Ambos)
- Hora del sistema (HH:MM: SS)

Los valores **límite** deben ser pasados como parámetros al proceso **Principal**.

1. El proceso **Registro** recibirá los datos y los registrará en un **archivo** (una línea por cada proceso), cabe destacar que solo se registrará el mismo proceso a los sumo dos veces, por ejemplo;

#{PID}: Supera CPU (Primer registro)

#{PID}: Supera memoria o Ambos (Segundo registro)

ó

#{PID}: Supera memoria (Primer registro)

#{PID}: Supera CPU o Ambos (Segundo registro)

(Nota el campo #{PID} deberá ser reemplazado por el PID del proceso que haya generado el evento)

Solo se registrará una vez aquellos procesos donde el primer registro sea por exceso de tipo Ambos o que solo excedan siempre CPU y no memoria o viceversa.

Criterios de corrección:

Control	Criticidad
Compila sin errores con el makefile entregado	Obligatorio
Funciona correctamente según enunciado	Obligatorio
Existe algún tipo de ayuda para la ejecución del proceso	Obligatorio
Valida correctamente los parámetros	Obligatorio
Cierra correctamente los recursos utilizados	Obligatorio
Finalizan correctamente todos los procesos	Obligatorio

Ejercicio 5:

La UNLaM desea centralizar la gestión de asistencias, para ello requiere un sistema que de acuerdo al rol del usuario que lo utilice permita:

El servidor contendrá:

El listado de alumnos/docentes de la Universidad. Un archivo con el siguiente formato:

- Usuario.txt

NOMBRE|CONTRASEÑA|ROL|COD_COMISION

LUCA|x5sqsa|D|1

JUAN|x2s1sa|A|1

PEPE|x3sbsa|A|1

ADRI|xaxqsa|D|2

SOFI|abxdda|A|2

El listado de todas las asistencias que se generarán.

Mostrar por pantalla los datos necesarios para loguearse.

Mostrará por pantalla y guardará en un archivo de extensión .log todas las conexiones que se establezcan como la información que reciba y envíe a sus clientes.

Como cliente:

- Al conectarse con el servidor deberá mostrar una pantalla para ingresar usuario y password. Dependiendo el rol con el que se loguee deberá:
- Docente (D)
 - Consultar asistencias de sus alumnos. Enviará como parámetro la fecha (formato de todas las fechas yyyy-mm-dd) a consultar, y el sistema imprimirá el listado de alumnos del curso asignado mostrando el presentismo correspondiente.
 - Cargar asistencias. Como primer paso escribirá la fecha de asistencia y luego listará por pantalla (de uno a la vez) los nombres de todos los alumnos asociados a esa comisión. El docente deberá indicar si están presentes o no completando con la letra P o A en cada caso.
 - Asistencia_[FECHA]_[COMISIÓN].txt

NOMBRE|PRESENTE

JUAN|A

PEPE|P

NOTA: No se requieren parámetros, con la fecha y hora actual se deberá determinar el listado de alumnos. Si el archivo ya se encuentra cargado arrojar un error

- Alumno (A)
 - Consultar si asistió en determinado día
 - Consultar el porcentaje de asistencias/inasistencias

Criterios de corrección:

Control	Criticidad
Compila sin errores con el makefile entregado .	Obligatorio
Funciona correctamente según enunciado .	Obligatorio
Existe algún tipo de ayuda.	Obligatorio
Utiliza socket. Modelo cliente servidor.	Obligatorio
Cierra correctamente los recursos utilizados.	Obligatorio
Finalizan correctamente todos los procesos.	Obligatorio
Soporta múltiples clientes en simultaneo.	Obligatorio