TRABAJO PRÁCTICO INTEGRADOR

VIRTUALIZACIÓN CON VIRTUALBOX Y DESARROLLO EN PYTHON

Materia: Arquitectura y Sistemas Operativos

Integrantes: Luciano José Cartagena y Santiago Arroquigaray

Profesor: Lic. Martín Aristiaran

Fecha de Entrega: 05/06/2025

OBJETIVOS Y ALCANCE DEL PROYECTO

- Instalar y configurar VirtualBox como hipervisor tipo 2
- Crear máquina virtual con distribución Linux Mint 21.3
- Desarrollar programa Python para cálculo de promedios de notas
- Documentar proceso completo con capturas y evidencias
- Demostrar competencias en virtualización y programación
- Crear repositorio organizado con todo el material del proyecto
- Generar video explicativo del funcionamiento completo

Objetivos del Proyecto

7 Metas Principales

FUNDAMENTOS DE VIRTUALIZACIÓN

¿QUÉ ES LA VIRTUALIZACIÓN?

- Tecnología que simula hardware y sistemas completos
- Permite ejecutar múltiples sistemas operativos simultáneamente
- Crea entornos aislados (sandbox) para experimentación segura
- Fundamental en la industria para desarrollo y testing

VENTAJAS PRINCIPALES:

- Experimentación sin riesgos para el sistema principal
- Optimización y mejor aprovechamiento de recursos de hardware
- Facilita administración de recursos y creación de entornos de prueba
- Permite desarrollo, testing y despliegue seguro de aplicaciones

VIRTUALBOX - HIPERVISOR TIPO 2

CARACTERÍSTICAS TÉCNICAS:

- Software gratuito y multiplataforma desarrollado por Oracle
- Hipervisor tipo 2 que se ejecuta sobre el sistema operativo anfitrión
- Ideal para entornos educativos por su facilidad de uso
- Compatible con diversos sistemas operativos huésped

FUNCIONALIDADES CLAVE:

- Gestión completa de recursos virtuales (CPU, RAM, disco, red)
- Asignación dinámica de espacio en disco
- Soporte para snapshots y clonación de máquinas virtuales
- Interfaces de red virtuales configurables (NAT, Bridge, Host-only)

CONFIGURACIÓN DE LA MÁQUINA VIRTUAL

ESPECIFICACIONES IMPLEMENTADAS:

Componente	Especificación	Detalle
Nombre	Linux-Python-VM	Identificador del proyecto
OS	Linux Mint 21.3 (64-bit)	Distribución Ubuntu-based
RAM	2 GB asignados	Memoria suficiente para desarrollo
Disco	20 GB dinámico	Expansión automática

PROCESO DE INSTALACIÓN:

- Descarga de imagen ISO oficial de Linux Mint
- Configuración de usuario administrador y contraseña
- Instalación de Guest Additions para mejor integración
- Configuración de interfaces de red virtuales

Configuración Técnica

Linux Mint 21.3

DESARROLLO DEL PROGRAMA PYTHON

FUNCIONALIDADES IMPLEMENTADAS:

- Solicitud interactiva de 3 notas al usuario
- Validación estricta de entrada (rango 0-10)
- Cálculo automático del promedio con 2 decimales
- Manejo robusto de excepciones para datos inválidos
- Feedback personalizado según el resultado obtenido

☆ BUENAS PRÁCTICAS APLICADAS:

- Validación de datos con bucles while y try-except
- Interfaz de usuario clara y amigable
- Código comentado y bien estructurado
- Manejo de errores para entrada de letras o valores fuera de rango
- Mensajes informativos para guiar al usuario

IMPLEMENTACIÓN TÉCNICA DEL CÓDIGO

ESTRUCTURA PRINCIPAL DEL PROGRAMA:

```python
# Validación robusta de entrada
for i in range(3):
    while True:
        try:
            nota = float(input(f"Ingrese la nota {i+1} (0-10): "))
            if 0 <= nota <= 10:
                notas.append(nota)
                break
            else:
                print("La nota debe estar entre 0 y 10")
        except ValueError:
            print("Por favor ingrese un número válido")

# Cálculo y feedback personalizado
promedio = sum(notas) / len(notas)
if promedio >= 7:
    print("¡Excelente trabajo!")
elif promedio >= 5:
    print("Buen trabajo, pero se puede mejorar")
else:
    print("Es necesario estudiar más")
```

Implementación Python

Código Principal

METODOLOGÍA Y ETAPAS DEL PROYECTO

PROCESO ESTRUCTURADO IMPLEMENTADO:

1. **Investigación previa:** Documentación oficial de VirtualBox y recursos académicos
2. **Instalación y configuración:** Setup paso a paso con documentación de cada etapa
3. **Desarrollo colaborativo:** Distribución de tareas entre los integrantes del equipo
4. **Programación en Python:** Aplicación de buenas prácticas y manejo de errores
5. **Pruebas exhaustivas:** Validación con diferentes casos de uso y escenarios
6. **Documentación técnica:** Creación de informe siguiendo estándares académicos
7. **Control de versiones:** Organización del material en repositorio GitHub

Metodología de Trabajo

7 Etapas Estructuradas

CASOS DE PRUEBA Y VALIDACIÓN

CASOS DE PRUEBA IMPLEMENTADOS:

Caso	Notas de Entrada	Promedio	Mensaje de Feedback
Notas altas	8.5, 9.0, 7.5	8.33	"¡Excelente trabajo!"
Notas medias	6.0, 5.5, 7.0	6.17	"Buen trabajo, pero se puede mejorar"
⚠ Notas bajas	3.0, 4.0, 2.5	3.17	"Es necesario estudiar más"

VALIDACIÓN DE ERRORES:

- Entrada de caracteres no numéricos (letras, símbolos)
- Números fuera del rango permitido (negativos, mayores a 10)
- Manejo de espacios en blanco y entradas vacías

Resultado: Programa robusto, confiable y con excelente experiencia de usuario

RESULTADOS OBTENIDOS Y COMPETENCIAS

LOGROS TÉCNICOS ALCANZADOS:

- Entorno virtualizado completamente funcional y estable
- Programa Python con manejo profesional de errores
- Documentación técnica completa con capturas de pantalla
- Repositorio GitHub organizado con estructura profesional
- Video explicativo demostrando funcionamiento completo

COMPETENCIAS DESARROLLADAS:

- Dominio de herramientas de virtualización (VirtualBox)
- Experiencia práctica con sistemas operativos Linux
- Programación Python aplicando buenas prácticas
- Habilidades de documentación técnica y organización
- Capacidad de resolución autónoma de problemas técnicos

Resultados del Proyecto

Competencias Adquiridas

DESAFÍOS ENCONTRADOS Y RESOLUCIÓN

PRINCIPALES DIFICULTADES Y SOLUCIONES:

Adaptación inicial a Linux: Curva de aprendizaje para usuarios acostumbrados a Windows

Solución: Investigación de comandos básicos y práctica constante con terminal

Configuración de recursos VM: Optimización del rendimiento de la máquina virtual

Solución: Ajuste de RAM asignada y habilitación de Guest Additions

Manejo de errores en Python: Implementación de validación robusta de entrada

Solución: Uso de try-except y validaciones múltiples con bucles while

Organización del proyecto: Estructura clara y profesional del repositorio

Solución: Seguimiento de estándares de documentación y control de versiones

Resolución de Problemas

Soluciones Implementadas

CONCLUSIONES Y PROYECCIONES FUTURAS

CONCLUSIONES PRINCIPALES:

- La virtualización es una herramienta fundamental para desarrollo seguro
- VirtualBox demostró ser una plataforma accesible y potente para aprendizaje
- Python permite desarrollo rápido y robusto con excelente manejo de errores
- La documentación técnica es clave para proyectos profesionales
- El trabajo colaborativo y la organización son esenciales para el éxito

PROPUESTAS DE MEJORA Y TRABAJO FUTURO:

- Explorar tecnologías de contenedorización como Docker
- Implementar interfaces gráficas en Python (Tkinter, PyQt)
- Automatizar configuraciones mediante scripts Bash
- Probar otras distribuciones Linux especializadas (Ubuntu Server, CentOS)
- Incluir pruebas unitarias y testing automatizado en el código

Proyecto Completado

Futuras Mejoras