

Universidad Tecnológica de Panamá  
Facultad de Sistemas Computacionales  
Asignatura: Desarrollo Lógico y Algoritmo  
Taller Práctico1

Profesor: Napoleón Ibarra

Estudiantes:

Cédulas:

Samuel Saldaña

4-811-232

Fernando Castillo

4-834-1140

Valor: 100 puntos

Fecha Inicio: 27/10/2025 --> 4:10 PM

Fecha Entrega: 28/10/2025 -->3:20 PM

**Procedimiento:**

1. De manera individual o en grupo de trabajo de 2 personas, realizar la asignación. Utilizando la herramienta Internet, investigue, desarrolle los conceptos solicitados.
2. Entregar el trabajo en formato digital (Parte I, II, III, IV) en PDF en la plataforma.

**I PARTE. Investigación. Valor 15 puntos**

Temas:

- 1) Procedimiento de búsqueda y ordenamiento de un arreglo.
  - 1.1. Búsqueda secuencial.
  - 1.2. Push Down.
- 2) Base de Datos MYSQL: Definición, ¿Qué se requiere para ser instalado?, ¿Qué se necesita para hacer una conexión PYTHON-MYSQL? Explique, ¿Qué es una replicación en una Base de Datos?

**Sustente su respuesta.**

**Procedimiento:**

1. Utilice la técnica (a su criterio) para desarrollar el tema propuesto.
2. En su desarrollo (Ponencia) debe estar los siguientes puntos:
  - 2.1. Un ejemplo (código sencillo funcional) de Arreglos. Su seudocódigo y simulación.

## 2.2. Concepto.

## 2.3. Su sintaxis.

### DESARROLLO

#### 1. Búsqueda y ordenamiento de un arreglo

##### 1.1. Búsqueda secuencial

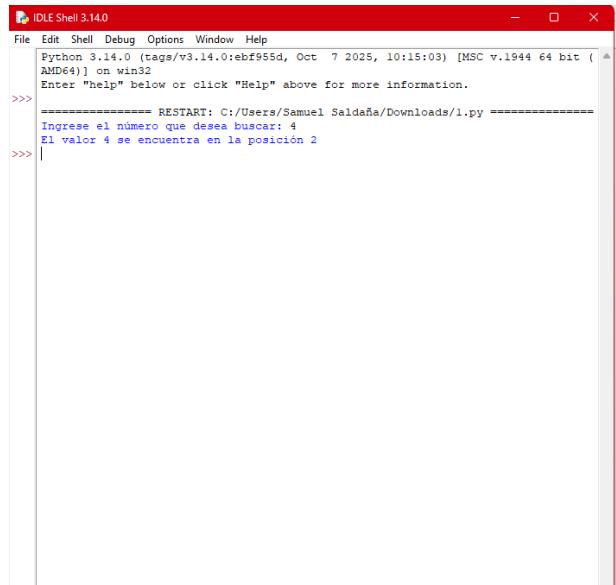
◆ **Concepto:** Busca un elemento en un arreglo recorriéndolo uno por uno hasta encontrarlo.

◆ **Sintaxis general en Python:**

```
def busqueda_secuencial(lista, valor):
    for i in range(len(lista)):
        if lista[i] == valor:
            return i
    return -1

lista = [5, 1, 4, 2]
valor = int(input("Ingrese el número que desea buscar: "))
resultado = busqueda_secuencial(lista, valor)

if resultado != -1:
    print(f"El valor {valor} se encuentra en la posición {resultado}")
else:
    print(f"El valor {valor} no se encuentra en la lista")
```



```
IDLE Shell 3.14.0
File Edit Shell Debug Options Window Help
Python 3.14.0 (tags/v3.14.0:ebf955d, Oct 7 2025, 10:15:03) [MSC v.1944 64 bit (AMD64)] on win32
Enter "help" below or click "Help" above for more information.
>>> ===== RESTART: C:/Users/Samuel Saldaña/Downloads/1.py =====
Ingrese el número que desea buscar: 4
El valor 4 se encuentra en la posición 2
>>>
```

◆ **Pseudocódigo:**

```
Inicio
    Definir lista Como Arreglo
    lista = [5, 1, 4, 2]           # (ejemplo)
    Escribir "Ingrese el valor a buscar:"
    Leer valor

    encontrado = Falso
    indice = -1

    Para i = 0 Hasta Longitud(lista) - 1 Hacer
        Si lista[i] = valor Entonces
            encontrado = Verdadero
            indice = i
            Salir Para          # opcional: detener búsqueda cuando se encuentre
        FinSi
    FinPara

    Si encontrado Entonces
        Escribir "Valor encontrado en la posición ", indice
    Sino
        Escribir "Valor no encontrado"
    FinSi
Fin
```

## 1.2. Push Down

◆ **Concepto:** Técnica de ordenamiento tipo burbuja descendente, donde los valores grandes “bajan” hasta el final.

◆ **Sintaxis en Python:**

```
File Edit Format Run Options Window Help
def push_down(lista):
    n = len(lista)
    for i in range(n):
        for j in range(0, n - i - 1):
            if lista[j] > lista[j + 1]:
                # Intercambia los valores
                lista[j], lista[j + 1] = lista[j + 1], lista[j]
    return lista

# Lista original
lista = [5, 1, 4, 2]

print("Lista original:", lista)

# Llamar a la función
lista_ordenada = push_down(lista)

print("Lista ordenada (de menor a mayor):", lista_ordenada)
```

```
File Edit Shell Debug Options Window Help
Python 3.14.0 (tags/v3.14.0:ebf955d, Oct 7 2025, 10:15:03) [MSC v.1944 64 bit (AMD64)] on win32
Enter "help" below or click "Help" above for more information.
>>>
=====
RESTART: C:/Users/Samuel Saldaña/Downloads/2.py =====
Lista original: [5, 1, 4, 2]
Lista ordenada (de menor a mayor): [1, 2, 4, 5]
>>>
```

◆ **Pseudocódigo**

```
File Edit Format Run Options Window Help
Inicio
    Definir lista[4] como Entero
    lista = [5, 1, 4, 2]
    Para i = 1 Hasta Longitud(lista) Hacer
        Para j = 1 Hasta (Longitud(lista) - i) Hacer
            Si lista[j] > lista[j + 1] Entonces
                aux = lista[j]
                lista[j] = lista[j + 1]
                lista[j + 1] = aux
            FinSi
        FinPara
    Escribir "Lista ordenada:", lista
Fin
```

## Base de Datos MySQL

◆ **Definición:** La conexión Python–MySQL permite que un programa hecho en Python se comunique con una base de datos MySQL para insertar, consultar, actualizar o eliminar datos.

Para eso, se utiliza el módulo llamado mysql.connector, que permite conectar, ejecutar consultas y cerrar la conexión.

◆ **Requisitos para instalar:**

- Instalar MySQL Server
- Instalar MySQL Workbench

- Configurar usuario y contraseña

◆ **Conexión Python–MySQL:**

Es el proceso mediante el cual un programa hecho en Python se comunica con una base de datos MySQL para enviar y recibir información.

A través de esta conexión, Python puede crear bases de datos, insertar, modificar, eliminar o consultar datos, usando librerías como mysql-connector-python o PyMySQL.

**Ejemplo:**

Python envía una orden (“inserta este cliente”) y MySQL la ejecuta en su base de datos.

◆ **Replicación en una base de datos:**

Es una técnica que duplica los datos de una base de datos principal (maestra) hacia una o más copias (esclavas o réplicas).

Su objetivo es mejorar la disponibilidad, seguridad y rendimiento, ya que si una base falla, las réplicas pueden seguir funcionando sin perder la información.

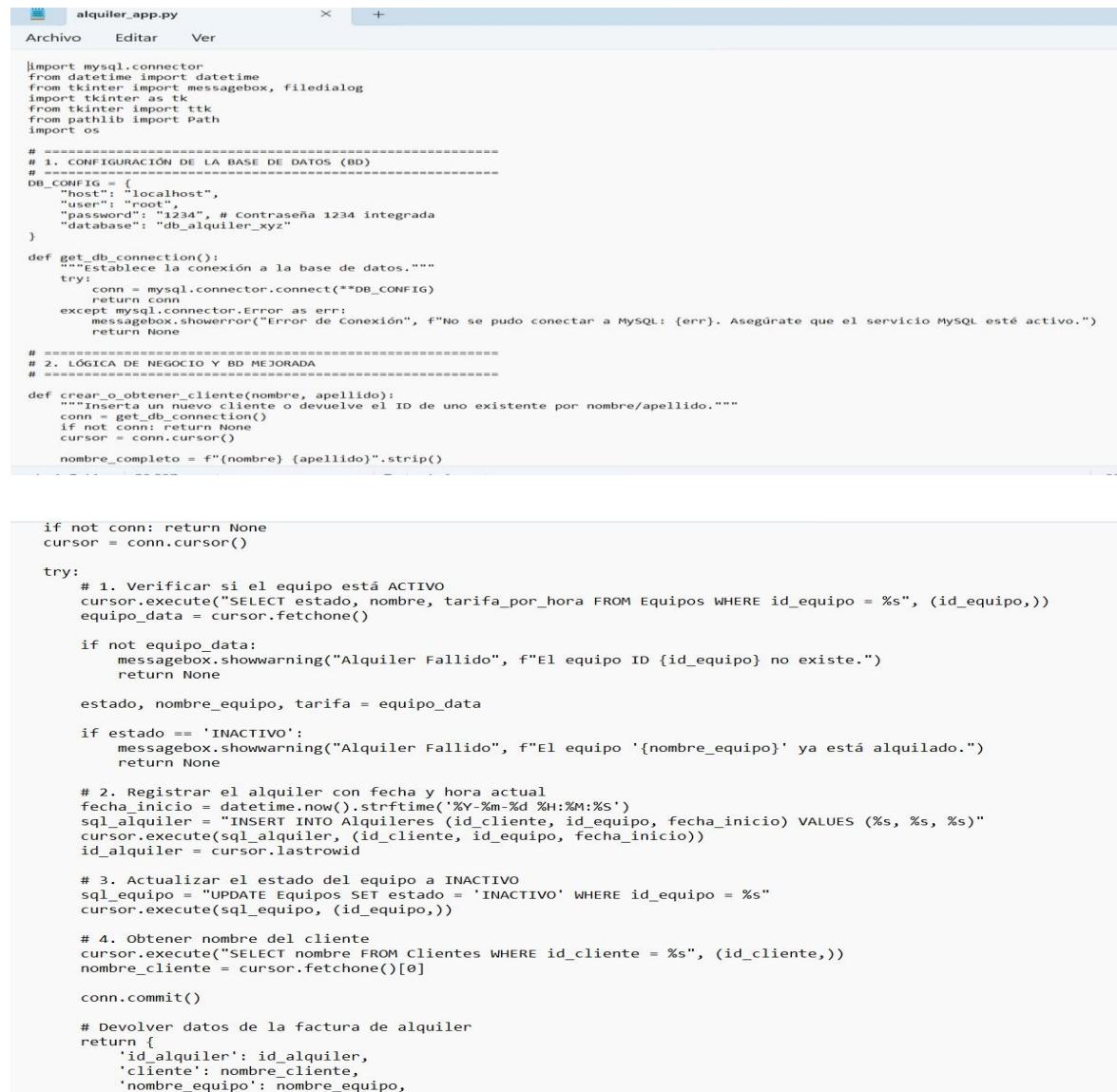
**Ejemplo:**

Si tienes una base de datos en un servidor principal y otra en respaldo, la replicación mantiene ambas sincronizadas automáticamente.

## II PARTE. Laboratorio. Valor 15 puntos

### Procedimiento:

1. Escenario local: instale, configure, haga pruebas de funcionamiento a MySQL en su equipo de producción.
2. Una vez instalado confeccione una base de datos (Usted elige su nombre). Tome en cuenta la III parte de la actividad.



```
alquiler_app.py
Archivo Editar Ver

import mysql.connector
from datetime import datetime
from tkinter import messagebox, filedialog
import tkinter as tk
from tkinter import ttk
from pathlib import Path
import os

# ===== 1. CONFIGURACIÓN DE LA BASE DE DATOS (BD) =====
# =====
DB_CONFIG = {
    "host": "localhost",
    "user": "root",
    "password": "1234", # Contraseña 1234 integrada
    "database": "db_alquiler_xyz"
}

def get_db_connection():
    """Establece la conexión a la base de datos."""
    try:
        conn = mysql.connector.connect(**DB_CONFIG)
        return conn
    except mysql.connector.Error as err:
        messagebox.showerror("Error de Conexión", f"No se pudo conectar a MySQL: {err}. Asegúrate que el servicio MySQL esté activo.")
        return None

# ===== 2. LÓGICA DE NEGOCIO Y BD MEJORADA =====
# =====

def crear_o_obtener_cliente(nombre, apellido):
    """Inserta un nuevo cliente o devuelve el ID de uno existente por nombre/apellido."""
    conn = get_db_connection()
    if not conn: return None
    cursor = conn.cursor()
    nombre_completo = f"{nombre} {apellido}".strip()

    if not conn: return None
    cursor = conn.cursor()
    try:
        # 1. Verificar si el equipo está ACTIVO
        cursor.execute("SELECT estado, nombre, tarifa_por_hora FROM Equipos WHERE id_equipo = %s", (id_equipo,))
        equipo_data = cursor.fetchone()

        if not equipo_data:
            messagebox.showwarning("Alquiler Fallido", f"El equipo ID {id_equipo} no existe.")
            return None

        estado, nombre_equipo, tarifa = equipo_data

        if estado == 'INACTIVO':
            messagebox.showwarning("Alquiler Fallido", f"El equipo '{nombre_equipo}' ya está alquilado.")
            return None

        # 2. Registrar el alquiler con fecha y hora actual
        fecha_inicio = datetime.now().strftime('%Y-%m-%d %H:%M:%S')
        sql_alquiler = "INSERT INTO Alquileres (id_cliente, id_equipo, fecha_inicio) VALUES (%s, %s, %s)"
        cursor.execute(sql_alquiler, (id_cliente, id_equipo, fecha_inicio))
        id_alquiler = cursor.lastrowid

        # 3. Actualizar el estado del equipo a INACTIVO
        sql_equipo = "UPDATE Equipos SET estado = 'INACTIVO' WHERE id_equipo = %s"
        cursor.execute(sql_equipo, (id_equipo,))

        # 4. Obtener nombre del cliente
        cursor.execute("SELECT nombre FROM Clientes WHERE id_cliente = %s", (id_cliente,))
        nombre_cliente = cursor.fetchone()[0]

        conn.commit()
        # Devolver datos de la factura de alquiler
        return {
            'id_alquiler': id_alquiler,
            'cliente': nombre_cliente,
            'nombre_equipo': nombre_equipo,
        }
    except:
        conn.rollback()
        raise
```

**III PARTE. Desarrollo prototipo. Valor 30 Puntos. Caso de Estudio: La empresa XYZ tiene sus servicios tecnológicos a disposición de sus clientes: alquiler de equipos (PC), impresiones, fotocopiado, otros. Requiere que Usted elabore/programe un prototipo de pseudocódigo y programa (PYTHON) que permita asignar/alquilar equipos, calcule el**

**costo final, guarde, almacene los registros, genere la factura en caso de que el cliente la solicite. Actualmente la organización cuenta con 11 equipos entre laptop, PC escritorio, impresora multifuncional. El programa debe ser capaz de insertar, actualizar, eliminar registros. Tome en cuenta los equipos (10), puede darse el caso que los mismos todos se estén utilizando a la misma vez, a modo de sugerencia controle quienes están activos/no activos.**

Sistema de Alquiler de Equipos XYZ					
Control de Alquileres		ID Cliente (Ej. 1): 1		ID Equipo a Alquilar: [ ]	
		1. Iniciar Alquiler		ID Alquiler a Devolver: [ ]	
		2. Devolver y Facturar		Actualizar Inventario	
<b>Estado del Inventario</b>					
ID Equipo	Nombre	Tipo	Tarifa/Hora	Estado	
1	LAPTOP-01	LAPTOP	3.50	ACTIVO	
2	LAPTOP-02	LAPTOP	3.50	ACTIVO	
3	LAPTOP-03	LAPTOP	3.50	ACTIVO	
4	PC-01	PC_ESCRITORIO	2.50	ACTIVO	
5	PC-02	PC_ESCRITORIO	2.50	ACTIVO	
6	PC-03	PC_ESCRITORIO	2.50	ACTIVO	
7	PC-04	PC_ESCRITORIO	2.50	ACTIVO	
8	PC-05	PC_ESCRITORIO	2.50	ACTIVO	
9	IMPRESORA-01	IMPRESORA_MF	1.00	ACTIVO	
10	IMPRESORA-02	IMPRESORA_MF	1.00	ACTIVO	
11	LAPTOP-04	LAPTOP	3.50	ACTIVO	

**IV PARTE. Diagrama de RED LAN. Valor 10 puntos. Procedimiento: Utilizando la herramienta Packet Tracer confeccione la propuesta de la Red LAN de acuerdo al caso de estudio de la II parte. Verifique su funcionamiento.**

