README PARCIAL "*SUPER PARQUEADERO DE MIGUEL Y THOMAS*"

Integrantes:

- Thomas Leandro Bacca Beltrán
- Miguel Lopez

¿En qué consiste nuestro proyecto?

El proyecto consiste en la creación de un sistema de parqueadero inteligente simulado mediante programación en Python. Nuestro sistema permite registrar la entrada y salida de vehículos ya sean carros o motos, asignar automáticamente espacios disponibles, calcular el tiempo de permanencia en horas y minutos y cobrar una tarifa basada en dicho tiempo. Además, muestra un mapa visual del parqueadero en tiempo real, indicando cuáles espacios están ocupados, vacíos, o son vías de circulación.

¿Por qué se consideró necesario?

Nosotros consideramos necesario desarrollar este proyecto como simulación de cómo funcionaría un sistema de parqueadero automatizado que mejore la organización, control y eficiencia del ingreso y salida de vehículos debido a que, en la vida real, muchos parqueaderos aún operan de forma manual, lo que puede generar desorden, pérdida de tiempo y errores en el cobro.

¿Cómo se llevó a cabo la implementación?

Para explicar este apartado empezaremos viendo como nosotros desarrollamos la idea para darle una solución al problema, ideas, luego la programación, errores y soluciones a los errores, veremos cada apartado por aparte:

En el parcial se nos pide hacer un sistema para administrar un parking desde 0, para eso tenemos que seguir ciertos criterios:

- Generación de Mapa
- Vías para entrada y salida
- Registro de vehículos
- Sistema de cobro
- Disponibilidad

Para esto empezamos a reducir el problema, es decir lo empezamos a resolver por partes, primero realizamos un diagrama en Exel con varias rutas vistas desde el cielo, escogiendo 1 la cual es:

E	-	-	-	-	-	-	-
V	V	V	V	V	V	V	-
-	-	-	-	-	-	-	-
-	V	V	V	V	V	V	V
-	-	-	-	-	-	-	-
V	V	V	V	V	V	V	-
-	-	-	-	-	-	-	-
S	V	V	V	V	V	V	V

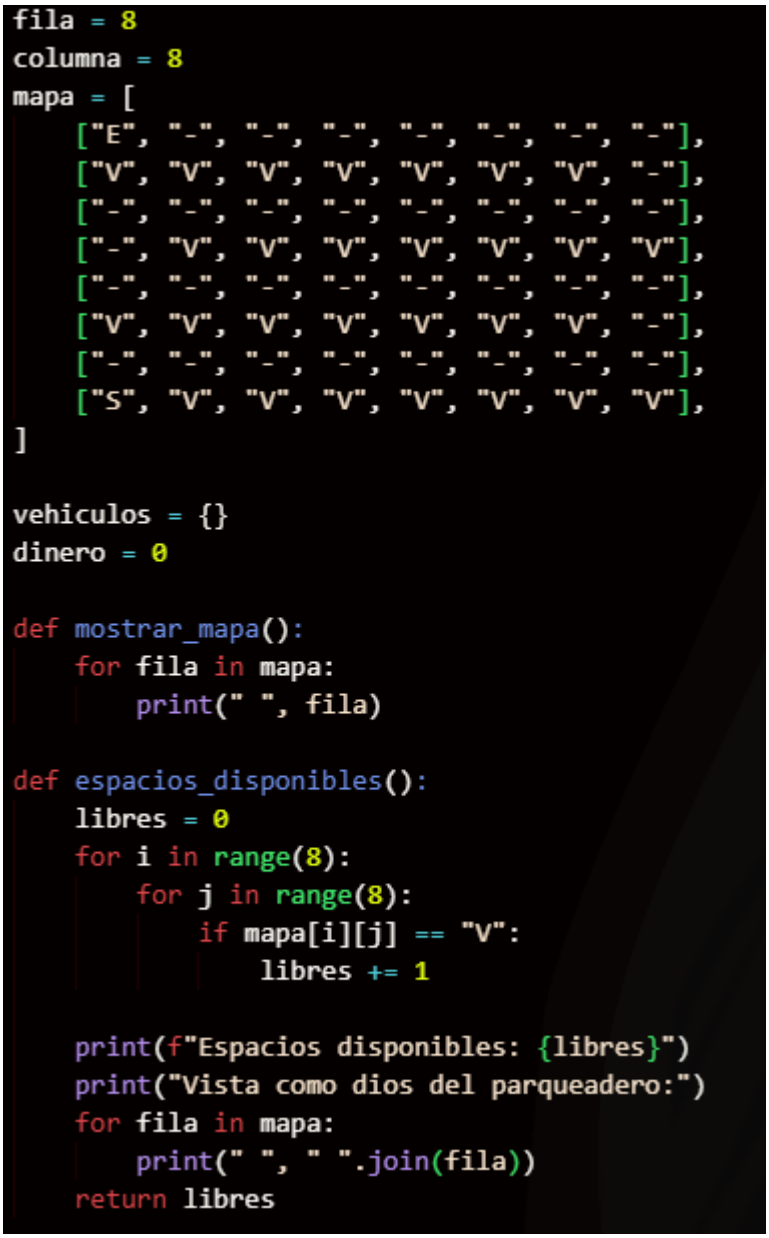
E	Entrada
S	Salida
-	Vía para vehículo
V	Espacio Vacío
O	Espacio Ocupado por Carro
M	Espacio ocupado por Moto

E	-	-	-	-	-	-	-
O	O	O	O	O	O	O	-
-	-	-	-	-	-	-	-
-	O	O	O	O	O	O	O
-	-	-	-	-	-	-	-
O	O	O	O	O	O	O	-
-	-	-	-	-	-	-	-
S	O	O	O	O	O	O	O

- Dentro de nuestro código se vería de esta manera:

1. Primero definimos la cantidad de columnas y filas, nosotros la definimos como 8x8, y ya que tenemos la “ruta” definida, luego creamos una variable global llamada vehículos para todos ellos

2. Luego nosotros definimos 2 funciones, **mostrar_mapa ()** para que se muestre el mapa en la terminal y el **espacios_disponibles ()** para que me cuente cuantos espacios disponibles hay, es decir cuantas “V” existen, además de mostrar la vista como si fuéramos Dios como el profe nos explicó en clase



```
fila = 8
columna = 8
mapa = [
    ["E", "-", "-", "-", "-", "-", "-", "-"],
    ["V", "V", "V", "V", "V", "V", "V", "-"],
    ["-", "-", "-", "-", "-", "-", "-", "-"],
    ["-", "V", "V", "V", "V", "V", "V", "V"],
    ["-", "-", "-", "-", "-", "-", "-", "-"],
    ["V", "V", "V", "V", "V", "V", "V", "-"],
    ["-", "-", "-", "-", "-", "-", "-", "-"],
    ["S", "V", "V", "V", "V", "V", "V", "V"],
]

vehiculos = {}
dinero = 0

def mostrar_mapa():
    for fila in mapa:
        print(" ", fila)

def espacios_disponibles():
    libres = 0
    for i in range(8):
        for j in range(8):
            if mapa[i][j] == "V":
                libres += 1

    print(f"Espacios disponibles: {libres}")
    print("Vista como dios del parqueadero:")
    for fila in mapa:
        print(" ", " ".join(fila))
    return libres
```

REGISTRO DE VEHICULOS INGRESO Y SALIDA

Ya que tenemos nuestro mapa realizado ahora podemos empezar la programación de el ingreso y salida de vehículos aquí veremos 2 agregados que nosotros quisimos implementar, el primero es un sistema de estadísticas donde se muestre:

- Cuantos vehículos hay
- Cuantos carros
- Cuantas motos
- Que cantidad de espacios quedan disponibles
- Cuánto dinero se recaudo

ESTADISTICAS

```python
def estadisticas():
    carros = 0
    motos = 0
    for datos in vehiculos.values():
        if datos[3] == "carro":
            carros += 1
        elif datos[3] == "moto":
            motos += 1

    print("Estadísticas del sistema")
    print(f"Vehículos en parqueadero: {len(vehiculos)}")
    print(f"Carros: {carros}")
    print(f"Motos: {motos}")
    espacios_disponibles()
    print(f"Dinero recaudado: ${dinero}")

def buscar_espacio():
    for i in range(8):
        for j in range(8):
            if mapa[i][j] == "V":
                return i, j
    return None
```

Función	¿Qué hace?	¿Dónde se usa?
estadísticas ()	Muestra información importante del estado actual del parqueadero. Se usa para tener una idea general de cómo va el sistema.	Se llama después de ingresar o retirar un vehículo. También cuando el usuario elige la opción de ver estadísticas en el menú.
buscar_espacio ()	Sirve para encontrar el primer espacio libre disponible en el parqueadero (donde haya un “V”).	Se llama en la función ingresar () para saber dónde poner el vehículo nuevo.

INGRESAR VEHICULOS

```python
def ingresar():
    placa = input("Placa del vehículo: ")
    tipo = input("¿Es carro o moto?: ")

    if tipo not in ["carro", "moto"]:
        print("Tipo no válido.")
        return

    lugar = buscar_espacio()
    if not lugar:
        print("No hay espacio disponible.")
        return

    hora = int(input("Hora de entrada (0-23): "))
    minutos = int(input("Minutos (0-59): "))
    entrada = hora * 60 + minutos

    i, j = lugar
    mapa[i][j] = "O" if tipo == "carro" else "M"
    vehiculos[placa] = [i, j, entrada, tipo]
    print(f"{tipo.capitalize()} con placa {placa} ingresado en ({i},{j})")

    estadisticas()
```

Primero definimos 2 variables que es la placa que va a tener el vehículo, y el tipo es decir si es carro o moto, aquí entra nuestro segundo agregado y es que el sistema que programamos es capaz de diferenciar entre carro o moto y que dependiendo lo que el usuario ingrese el dinero se calculará de forma diferente y en el mapa que vera el usuario ya no aparecerá una O si no aparecerá una M.

IMPORTANTE: *"Si o si debe ponerse carro o moto en minúscula de otra forma el sistema no lo detectara y mostrara tipo no valido"*

Luego el sistema llamara a la función buscar_espacio () para encontrar un lugar disponible, si llega a no haber un espacio disponible la función termina.

Ya después se convierte la hora y los minutos en minutos totales desde las 00:00. Esto facilita calcular el tiempo después, luego se marca el espacio como ocupado con una "O" para carro o "M" para moto y guarda la información del vehículo en el global vehículos.

Para finalizar el sistema muestra un mensaje de que fue ingresado con éxito y se actualiza la función estadística.

RETIRAR VEHICULOS

```python
def retirar():
    global dinero
    placa = input("Placa del vehículo a retirar: ")

    if placa not in vehiculos:
        print("Vehículo no encontrado.")
        return

    hora = int(input("Hora de salida (0-23): "))
    minutos = int(input("Minutos (0-59): "))
    salida = hora * 60 + minutos

    i, j, entrada, tipo = vehiculos[placa]
    tiempo = salida - entrada
    if tiempo <= 0:
        print("Error: la salida debe ser después de la entrada.")
        return

    tarifa = 100 if tipo == "carro" else 50
    total = tiempo * tarifa
    dinero += total

    mapa[i][j] = "V"
    del vehiculos[placa]

    print(f"Vehículo {placa} retirado.")
    print(f"Tiempo: {tiempo} minutos")
    print(f"Total a pagar: ${total}")

    estadisticas()
```

Para definir el cómo retiramos los vehículos, primero tendremos que hacer un input de la placa para que el sistema evalué si esa placa existe o no, si no esta el sistema solo muestra el error de “Vehículo no encontrado” y si el vehículo si esta registrado en el sistema se abrirán 2 input. Uno para pedir la hora y el otro para los minutos. Luego hará el cálculo para la variable de la salida.

¿Qué hace el código después de definir las variables?

- Extrae los datos guardados del vehículo que entró al parqueadero.
- i, j: posición en el mapa.
- entrada: hora en que ingresó (en minutos).
- tipo: si es un "carro" o una "moto".
- Calcula cuántos minutos estuvo el vehículo en el parqueadero.
- Verifica que la hora de salida sea después de la hora de entrada.

- Si no lo es, detiene el proceso y muestra un mensaje de error.
- Calcula la tarifa si es carro es 100 pesos si es moto es 50
- Modifica el mapa y cambia el espacio de ocupado a vacío
- Esos datos los actualiza en las estadísticas

MENU

```python
def menu():
    while True:
        print("Menú del Super Parqueadero Miguel y Leandro")
        print("1. Ingresar vehículo")
        print("2. Retirar vehículo")
        print("3. Ver mapa actual")
        print("4. Ver estadísticas")
        print("5. Salir")
        opcion = input("Opción: ")

        if opcion == "1":
            ingresar()
        elif opcion == "2":
            retirar()
        elif opcion == "3":
            espacios_disponibles()
        elif opcion == "4":
            estadisticas()
        elif opcion == "5":
            print("Gracias por usar nuestro sistema de parqueadero. Profe, pónganos 50 :)")
            break
        else:
            print("Opción no válida.")

menu()
```

Opción	Función llamada	Descripción / Qué hace
1	ingresar ()	Ingresa un vehículo al parqueadero, asigna espacio y registra hora y tipo.
2	retirar ()	Retira un vehículo, calcula tiempo, cobra y libera espacio.
3	espacios_disponibles ()	Muestra la cantidad de espacios disponibles y el mapa desde arriba.
4	estadísticas ()	Muestra el total de vehículos, cuántos son carros o motos, espacios libres y dinero total.
5	break (salir)	Termina el programa con un mensaje de despedida.
Otra	Ninguna	Muestra "Opción no válida" si el número ingresado no está entre 1 y 5.

Errores y soluciones durante el proceso

1. **Columnas y fila**
   El primer error que nos ocurrió es que, al realizar el mapa, en ocasiones confundíamos las

columnas con las filas, y el mapa se llegaba a dar toda la vuelta. O que cuando hacíamos el print para que el mapa apareciera no aparecía.

Solución:

https://youtu.be/v25-m1LOUiU?si=V13Y8_p_Q025S-KB

Vimos este video como re introducción al tema y nos guiamos en el código utilizando la parte que decía conjuntos.

2. **Olvidar el return en espacios_disponibles ()**:
   Aunque cuentas los espacios disponibles correctamente, si no usamos el return libres, otras funciones no podrán usar ese número. Por ejemplo, más adelante necesitamos guardar o comparar ese valor, no funcionará sin el return.
   Solución:
   Si una función como estadísticas () quiere mostrar cuántos espacios hay, necesita recibir ese valor. Para solucionarlo, simplemente hay que asegurarse de que al final de la función espacios_disponibles () se incluya una instrucción que devuelva el número de espacios libres. Así, otras funciones pueden usar ese valor cuando lo necesiten.
   (Nos ayudó https://www.youtube.com/watch?v=hHhkLEnowLA)

3. **Confundir cómo funciona una matriz (lista de listas)**:
   No comprenden bien cómo funcionan las listas anidadas. Pueden confundirse al recorrer el mapa o pensar que for fila in mapa: es lo mismo que acceder por índices como mapa[i][j].
   Solución:
   (Nos ayudó https://docs.python.org/es/3/tutorial/datastructures.html)

4. **Acceder por índices sin saber qué contienen (datos [3]):**
   En la función estadisticas(), se accede directamente a datos[3] para saber si el vehículo es "carro" o "moto". Si alguien cambia el orden de los elementos en la lista vehículos[placa], este código dejará de funcionar.
   Solución:
   (Nos ayudo https://www.youtube.com/watch?v=yunyV8JDC6)

5. Retirar placas FALLO MUCHO ;(
   Cuando nosotros introducíamos la placa del auto y después queríamos retirar el auto, la terminal mostraba un error que decía vehículo no encontrado.
   Solución:

   Para arreglarlo, debes usar el mismo tipo de dato para la placa en ambas funciones. Te recomiendo manejarla como cadena (str), porque algunas placas pueden incluir letras.

   Cambia esto en Ingresar Vehículo ():

python
CopiarEditar
placa = input("Ingrese la placa del Carro: ") # <-- quita el int()

(Profe para este problema nos ayudo CHATGPT)