PROYECTO GRUPO #4
PARKING CENTER S.A.S

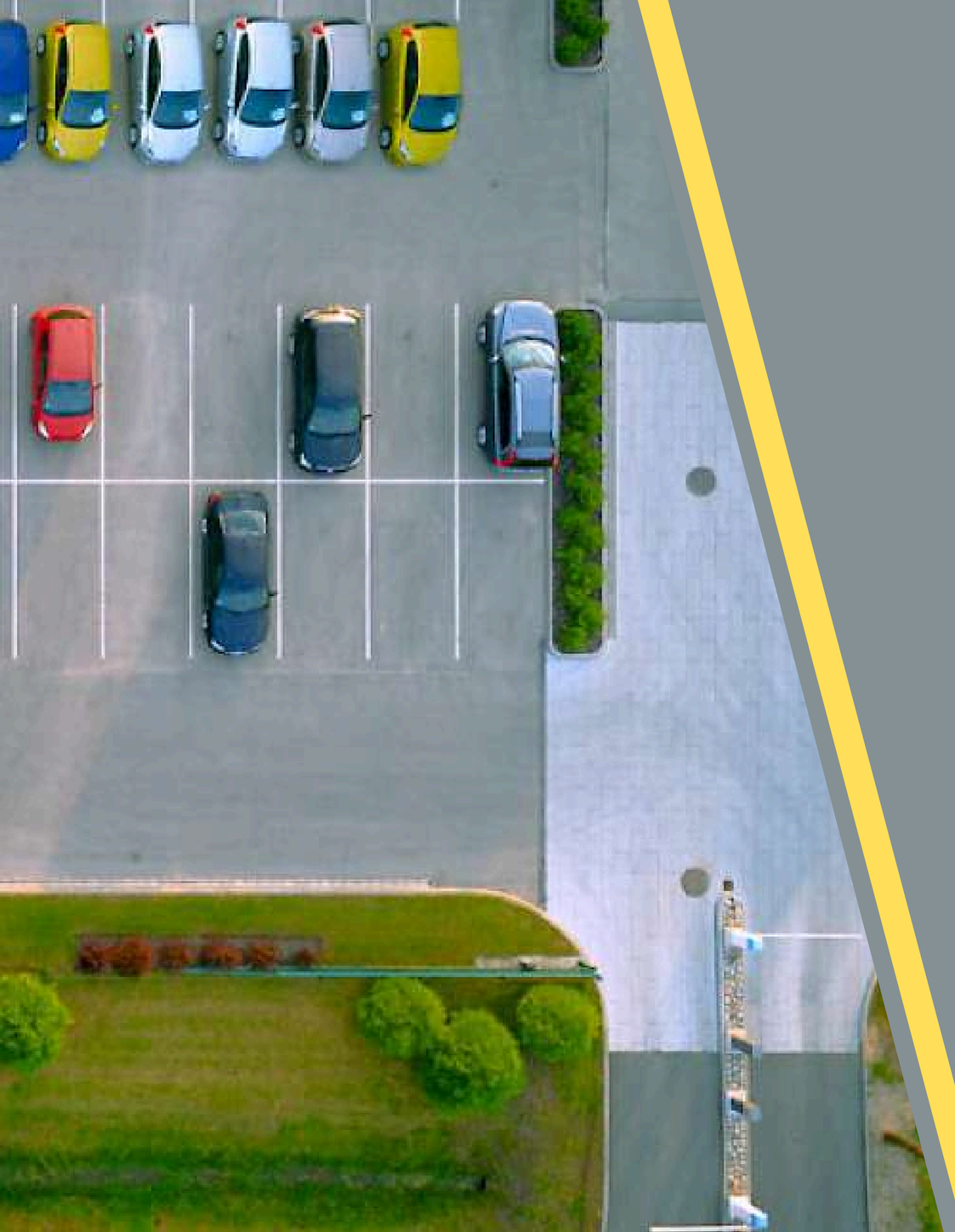
CONTROL DE PARQUEADERO

PARKING CENTER S.A.S

PARQUEADEROS

ÍNDICE

1. Introducción
2. Objetivos
3. Análisis y requerimientos
4. Problemas
5. Soluciones
6. Procesos
7. Metodos,librerias e investigaciones usadas
8. Conclusiones



INTRODUCCION

Con el objetivo de fortalecer sus procesos operativos, Parking Center S.A.S., empresa dedicada a la administración de parqueaderos en centros comerciales y zonas empresariales, plantea la creación de un prototipo de sistema de registro y cobro. Este sistema permitirá gestionar de manera sencilla el ingreso y salida de vehículos, calcular el valor a pagar según el tiempo de permanencia y, a su vez, servirá como herramienta de formación y práctica para la capacitación de nuevos empleados.

P|C
PARKING CENTER
S.A.S
PARQUEADEROS

OBJETIVOS

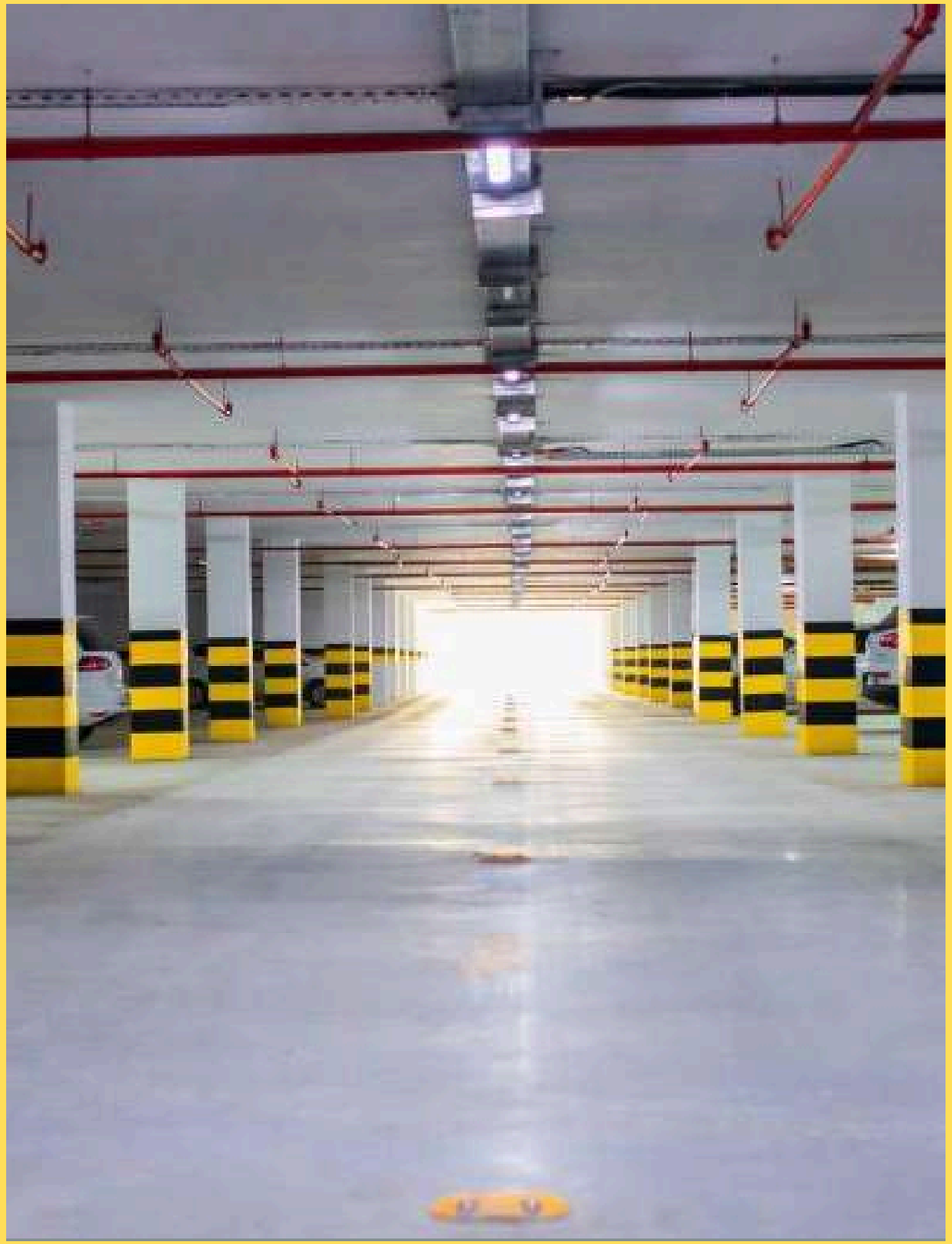
Desarrollo de un prototipo funcional: Se creó un sistema de parqueadero que registra ingresos y salidas de vehículos, además de calcular los cobros de forma básica y automatizada.

Implementación de requerimientos técnicos: Se aplicaron estructuras de datos como tuplas y diccionarios para gestionar información, calcular tarifas y simular tiempos de permanencia.

Validación de la solución planteada: Se comprobó que el sistema detecta errores como placas repetidas o inexistentes, garantizando un uso práctico y confiable.

Refuerzo de habilidades en programación: El proyecto permitió aplicar conocimientos en programación, datos y lógica, acercándolos a un problema real del entorno empresarial.

P|C
PARKING CENTER
S.A.S
PARQUEADEROS



ANALISIS Y REQUERIMIENTOS

ANALISIS

Antes de codificar,o empezar a hacer el diagrama de flujo,obviamente empezamos por algo importante; El analisis,donde analizamos e investigamos librerias,metodos,etc. En la programacion no es redactar codigo y ya,la programacion es saber DESARROLLAR; Saber racionalizar,solucionar y mejorar problemas reales,por eso,antes de todo,lo que analizamos fue:

1 - Tener una base solida; saber de condicionales y bucles

2 - Un prototipo funcional

3 - Un codigo optimizado,que no ocupe mucho espacio para ejecutarse

4 - investigaciones; librerias,metodos

5 - Análisis de los resultados

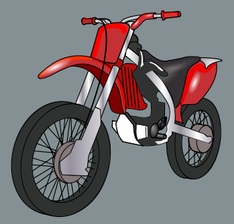
6 - Conclusiones

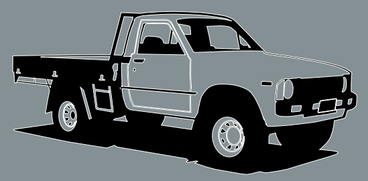
P|C
PARKING CENTER S.A.S
PARQUEADEROS

ANALISIS Y REQUERIMIENTOS

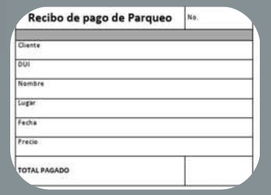
REQUERIMIENTOS

REGISTRAR EL INGRESO DE UN VEHICULO (PLACA,TIPO,HORA DE ENTRADA)





REGISTRAR LA SALIDA DE UN VEHICULO Y SU RECIBO DE PAGO



CONSULTAR EL PARQUEADERO ACTUAL

VER TOTAL RECAUDADO

SALIR DEL SISTEMA

PARKING CENTER S.A.S
PARQUEADEROS

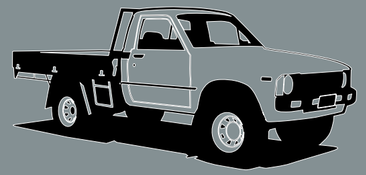
ANALISIS Y REQUERIMIENTOS

REQUERIMIENTOS: DETALLES TECNICOS

1. Datos de los vehiculos para el parqueadero

Cada vehiculo del parqueadero esta en una tupla,donde se guardan en una lista de vehiculos activos,para consultarlos en el parqueadero

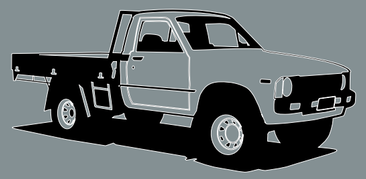
Sus datos son: Placa,tipo de vehiculo,hora de entrada



2. Tipos de vehiculos y tarifas

$1000 por HORA

$2000 por HORA



$2500 por HORA

3. Menu interactivo con bucles y condicionales

Registrar ingreso

Registrar salida

Consultar parqueadero actual

Ver total recaudado

Salir del sistema

4. Formulas matematicas

Se calculan horas de permanencia en el parqueaadero y multiplicar por tarifa mediante redondeos como principal metodo o Simular el ingreso/salida con horas ficticias (número entero entre 0 y 24)

5. F-Strings

Mostrar recibos claros: placa, tipo, horas, tarifa por hora, total a pagar.

----------RECIBO DE PAGO----------

TIPO DE VEHICULO	PLACA:
HORAS ESTACIONADO:	HH
TARIFA POR HORA:	$ XXXXXX
TOTAL A PAGAR:	$ XXXXXX
HORA DE ENTRADA:	HH/MM/SS
HORA DE ENTRADA:	HH/MM/SS

P|C
PARKING CENTER S.A.S
PARQUEADEROS

PROBLEMAS

Tuvimos tres tipos de problemas:

1 - Antes de hacer el codigo

2 - Mientras se hacia el codigo

3 - Despues del codigo

En el cual,primero analizamos los errores,y luego solucionamos,donde en cada tipo de problema eran los siguientes:

- Antes de hacer el codigo

1 - Saber que librerias importar para el tiempo,y redondear las horas y funciones para leer la hora exacta

2 - Como recorrer cada vehiculo,pero solo su placa

3 - Desempaquetar tuplas con sus datos completos

P|C
PARKING CENTER S.A.S
PARQUEADEROS

PROBLEMAS

- Mientras se hacia el codigo:

1 - Como implementar las librerias y funciones

2 - Como empezar desde cero; Analisis de cada parte

3 - Como juntar las ideas de todos

4 - Solucionar los errores de identacion

- Despues del codigo:

1 - Ver que no nos gusto

2 - Ver posibles mejoras

PARKING CENTER S.A.S
PARQUEADEROS

SOLUCIONES

- Antes de hacer el codigo:

1 - Saber que librerias importar para el tiempo,y redondear las horas y funciones para leer la hora exacta: Indagamos muchos sitios web que nos proporcionaban varias librerias,buscamos como ‘Librerias de tiempo Python’ etc,en el cual nos aporto mucho para saber cuales son las que debemos usar

2 - Como recorrer cada vehiculo,pero solo su placa: No sabiamos desde el principio como,sabiamos que era mediante un indice acumulativo,pero no sabiamos como implementarlo a los condicionales-bucle,hasta que hallamos la solucion desarmando el problema en pedazos

3 - Desempaquetar tuplas con sus datos completos: Ya que para el recibo nesecitamos sacar los datos de un carro; Osea su tupla,por lo tanto buscamos metodos de desempaquetado de tupla para poder hacerlo

P|C
PARKING CENTER S.A.S
PARQUEADEROS

SOLUCIONES

- Mientras se hacia el codigo:

1 - Como implementar las librerias y funciones: Buscamos muchos tutoriales de Yt e investigaciones en muchos sitios web,para indagar como se empieza un programa con estas librerias y cuales funciones aportan estas

2 - Como empezar desde cero; Analisis de cada parte: Empezamos pensando por las estructuras logicas; Bucles y condicionales,ya que en el codigo se puede detallar que esa es la estructura del codigo

3 - Como juntar las ideas de todos: Propusimos ideas con los mockups y funciones que todos quisimos usar al mismo tiempo para mejorar la calidad y optimizacion del codigo

4 - Solucionar los errores de identacion: Ya que al corto espacio que habia entre los condicionales y bucles,a veces teniamos que separarlos,y eso o empeoraba algo,o hacia algo indetectable

P C
PARKING CENTER
S.A.S
PARQUEADEROS

SOLUCIONES

- Despues del codigo:

1 - Qué no nos gustó

Después de la elaboración del código se notó que algunos mensajes mostrados con print no eran lo suficientemente claros, además la estructura del programa resultaba un poco extensa en ciertos apartados y la interacción con el usuario no siempre era precisa ni directa.

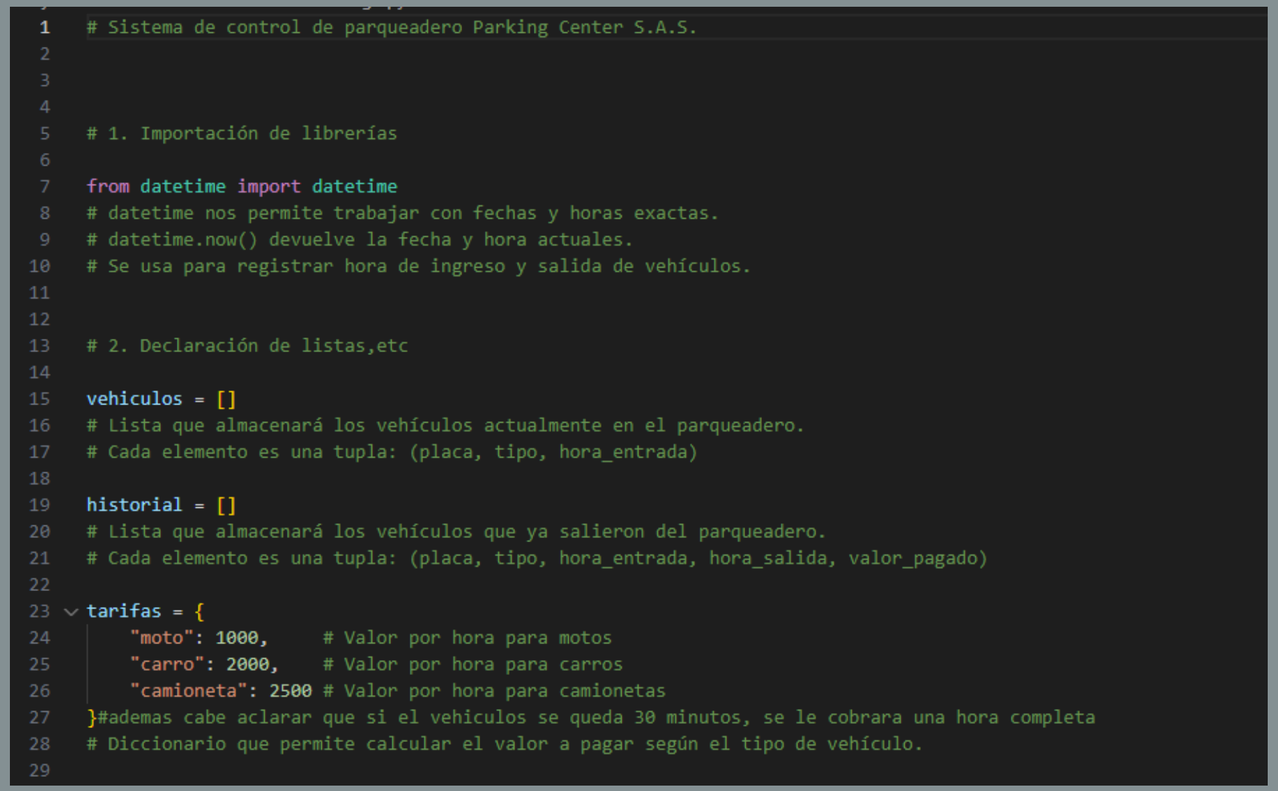
2 - Posibles mejoras

Se consideró simplificar los mensajes para hacerlos más comprensibles, reorganizar la lógica con estructuras adecuadas que reduzcan complejidad y eliminar repeticiones innecesarias, logrando así un equilibrio entre claridad, eficiencia y facilidad de mantenimiento.

P C
PARKING CENTER
S.A.S
PARQUEADEROS

PROCESOS

En esta parte; Procesos,explicaremos de manera resumida y detallada de manera corta,las estructuras del codigo,para que las entiendan antes de explicarlo en el documento WORD de documentacion acerca del codigo y la exposicion de este mismo; Osea es solo una introduccion hacia el codigo



```python
# Sistema de control de parqueadero Parking Center S.A.S.




# 1. Importación de librerías

from datetime import datetime
# datetime nos permite trabajar con fechas y horas exactas.
# datetime.now() devuelve la fecha y hora actuales.
# Se usa para registrar hora de ingreso y salida de vehículos.


# 2. Declaración de listas,etc

vehiculos = []
# Lista que almacenará los vehículos actualmente en el parqueadero.
# Cada elemento es una tupla: (placa, tipo, hora_entrada)

historial = []
# Lista que almacenará los vehículos que ya salieron del parqueadero.
# Cada elemento es una tupla: (placa, tipo, hora_entrada, hora_salida, valor_pagado)

tarifas = {
    "moto": 1000,     # Valor por hora para motos
    "carro": 2000,    # Valor por hora para carros
    "camioneta": 2500 # Valor por hora para camionetas
}#ademas cabe aclarar que si el vehiculos se queda 30 minutos, se le cobrara una hora completa
# Diccionario que permite calcular el valor a pagar según el tipo de vehículo.

```

En esta parte importamos la libreria datetime para permitirnos trabajar con horas exactas,luego la lista de vehiculos donde se guardaran todos los vehiculos activos,luego la de historial,que nos permitira saber cuales han sido todos los vehiculos registrados en el dia,las tarifas de precio por hora segun el tipo de vehiculo...

P|C
PARKING CENTER S.A.S
PARQUEADEROS

PROCESOS

```
30  opc = 0
31  # Variable para almacenar la opción elegida por el usuario en el menú principal.
32
33
34  # 3. Bucle principal del sistema
35
36  while opc != 5:
37      # Este bucle se ejecuta mientras el usuario NO seleccione la opción 5 (Salir) si presiona el 5 el bucle se terminará
38
39      print("----------Parking Center S.A.S.----------")
40      # Imprime el encabezado del sistema
41
42      print("""
43  1. Registrar ingreso
44  2. Registrar salida
45  3. Consultar parqueadero actual
46  4. Ver total recaudado
47  5. Salir
48  """)
49      # Muestra el menú de opciones numeradas para que el usuario elija
50
51      opc = int(input("Ingresa el número de la opción deseada: "))
52      # Solicitamos al usuario que ingrese un número.
53      # int() convierte la entrada (str) a entero para poder usarla en condicionales.
54
```

Creamos la variable 0 para que el bucle empieze con los procesos de cada opciones mientras no sea la opcion 5,osea salir; Donde imprimos las opciones,y le pedimos que opcion quiere escoger...

P|C
PARKING CENTER S.A.S
PARQUEADEROS

PROCESOS

```python
    # OPCIÓN 1 - Registrar ingreso de vehículo

    if opc == 1:
        placa = input("Ingresar placa: ")
        # Solicitamos la placa del vehículo que desea ingresar

        # Verificamos que la placa no esté ya registrada con este bucle:
        j = 0  # Inicializamos un contador para recorrer la lista de vehículos,donde J es el indice que cambiara de tupla
        while j < len(vehiculos):
            # Recorremos la lista de vehículos activos usando un índice j,mientras que j sea menor a la cantidad de vehiculos,ya que si j es mayor a la lista de vehiculos,estara reccoriendo tuplas de vehiculos que no existen
            if vehiculos[j][0] == placa:
                # Si la placa ya existe (vehiculos[j][0] accede a la placa) imprimira:
                print("Error: Un vehículo con esta placa ya está en el parqueadero.")
                placa = input("Ingresar otra placa: ")
                # Solicitamos nuevamente la placa
                j = -1
                # Reiniciamos j a -1 porque al final del ciclo se hace j += 1 para que no recorra la misma placa
                # Esto permite volver a recorrer toda la lista desde el inicio
            j += 1
            # Incrementamos j en 1 para pasar al siguiente vehículo en la lista
            # Si no hacemos esto, el bucle se quedaría infinito

        # Selección del tipo de vehículo
        print("""
Tipos de vehículos:
1. Moto
2. Carro
3. Camioneta
""")
        tipo_num = int(input("Ingresa el número del tipo de vehículo: "))
        # Solicitamos al usuario elegir un tipo mediante un número

        if tipo_num == 1:
            tipo = "moto"
        elif tipo_num == 2:
            tipo = "carro"
        elif tipo_num == 3:
            tipo = "camioneta"
        else:
            tipo = "carro"
            # Si el número ingresado es inválido, se asigna carro por defecto

        hora_entrada = datetime.now()
        # Guardamos la hora exacta de ingreso usando datetime.now()

        vehiculos.append((placa, tipo, hora_entrada))
        # Añadimos una tupla con la información del vehículo a la lista de vehículos activos
        # append() agrega el elemento al final de la lista

        # Mostramos un mensaje de confirmación
        # strftime('%H:%M:%S') convierte el objeto datetime a formato de hora legible HH:MM:SS
        print(f"Vehículo {placa} tipo {tipo} registrado a las {hora_entrada.strftime('%H:%M:%S')}\n")
```

Luego de pedirle la opcion al usuario,ya empezamos con los condicionales; La opcion 1: Registrar ingreso de vehiculo: Donde se le pide la placa,y hacemos una variable J que sera la encargada de aumentarse para recorrer cada tupla (osea cada vehiculo) mediante un bucle y un condicional para detectar si algun carro tiene la placa que el usuario quiere registrar; Dando error,y vuelve a pedir una placa,y j se regresa al inicio

Para luego imprimirle los tipos de vehiculos,y que escoga una de los 3,para luego con condicionales crear la variable del tipo ingresado,si es 1 moto,si es 2 carro,y si es 3 camioneta,pero si no ingresa ninguna de las 3,se asignara por descarte que es carro.
Para luego coger la hora de entrada con la funcion .now() acompañada de la libreria
Luego guardamas en la lista de vehiculos la placa,tipo de vehiciculo,y hora de entrada,e imprimirle lo que registro

PARKING CENTER S.A.S
PARQUEADEROS

PROCESOS

```python

    # OPCIÓN 2 - Registrar salida de vehículo

    elif opc == 2:
        placa_salida = input("Ingresar placa para retirar: ")
        # Solicitamos la placa del vehículo que desea salir

        j = 0  # Inicializamos un contador para recorrer la lista de vehículos
        encontrado = False
        # Bandera que nos indica si encontramos la placa en la lista de vehículos,en este caso False,para que lo confirme y tengamos esa señalizacion

        while j < len(vehiculos):
            if vehiculos[j][0] == placa_salida:
                # Comprobamos si la placa ingresada coincide con la placa del vehiculo en la posición j (el indice de la tupla que accede al valor de la placa)
                encontrado = True # Si es verdadera,la bandera sera True
                placa, tipo, hora_entrada = vehiculos[j]
                # Desempaquetamos la tupla para obtener placa, tipo y hora de entrada
                # Donde el primer valor lo desempaquetamos como placa,tipo como tipo,hora de entrada como hora entrada de la tupla del carro indice(j)

                hora_salida = datetime.now()
                # Obtenemos la hora exacta de salida con la funcion now que agarra la hora exacta

                duracion = hora_salida - hora_entrada
                # Calculamos el tiempo que el vehículo estuvo estacionado
                # El resultado es un objeto timedelta

                horas = duracion.total_seconds() / 3600
                # Convertimos la duración de segundos a horas decimales con el metodo total.seconds que convierte a decimales,las horas minutos segundos

                horas = int(horas) + 1
                # Redondeamos hacia arriba al siguiente número entero
                # Esto asegura que cualquier fracción de hora se cobre como hora completa

                valor = horas * tarifas[tipo]
                # Calculamos el valor a pagar multiplicando las horas por la tarifa correspondiente

                # Mostramos el recibo de pago
                print(f"\n--- Recibo de Pago ---")
                print(f"Placa: {placa}")
                print(f"Tipo: {tipo}")
                print(f"Horas estacionado: {horas}")
                print(f"Tarifa por hora: ${tarifas[tipo]}")
                print(f"Total a pagar: ${valor}")
                print(f"Hora entrada: {hora_entrada.strftime('%H:%M:%S')}") # strftime convierte valores de hora,minuto y segundo a strings
                print(f"Hora salida: {hora_salida.strftime('%H:%M:%S')}\n") # strftime convierte valores de hora,minuto y segundo a strings

                historial.append((placa, tipo, hora_entrada, hora_salida, valor))
                # Guardamos la información completa en la lista historial
                # append() agrega la tupla al final de la lista

                vehiculos.pop(j)
                # Eliminamos el vehículo de la lista de vehículos activos
                # pop(j) elimina el elemento en la posición j para mantener la lista actualizada

                break  # Salimos del bucle porque ya encontramos y procesamos la placa
            j += 1
            # Incrementamos j para pasar al siguiente vehículo en la lista

        if not encontrado:
            # Si la bandera sigue en False, significa que la placa no estaba registrada
            print("Error: La placa no se encuentra registrada en el parqueadero.\n")


```

La opcion 2: Registrar salida de vehiculo:
Donde se le pide la placa para retirar,y se inicia la variable indexativa de los carros J y una bandera llamada False para confirmar si el carro fue o no encontrado.
Donde inicia el bucle,y si el indice [J][0] osea la placa es igual a la que da el usuario,se pone la bandera encontrado,y se desempaqueta la tupla para el recibo.
Y luego se calcula las horas y se cobran,para luego mostrar el recibo de pago

Y luego agregar ese vehiculo a lista de vehiculos que ya no estan activos,y eliminarlo de la de activos,y luego j aumenta en el bucle para pasar al siguiente si no lo encuentra.
Y si despues de recorrer toda la lista no esta,esta el if ultimo para decir Error

PARKING CENTER S.A.S
PARQUEADEROS

PROCESOS

```
172     # OPCIÓN 3 - Consultar parqueadero actual
173
174     elif opc == 3:
175         if vehiculos != []:  # Si la lista no esta vacia se hace el la condicion e inicia el bucle
176             # Verificamos si hay vehículos en el parqueadero
177             print("\n--- Vehículos en el parqueadero ---")
178             j = 0
179             while j < len(vehiculos):
180                 # Recorremos la lista de vehículos activos
181                 placa, tipo, hora_entrada = vehiculos[j]
182                 # Desempaquetamos cada tupla
183                 print(f"Placa: {placa} | Tipo: {tipo} | Hora entrada: {hora_entrada.strftime('%H:%M:%S')}")
184                 j += 1  # Incrementamos j para pasar al siguiente vehículo en la lista
185             print()
186         else: # Si no se cumple,y esta vacia,ose muestra:
187             # Si no hay vehículos, mostramos un mensaje
188             print("No hay vehículos en el parqueadero.\n")
189
190
```

La opcion 3: Consultar parqueadero actual; Donde empieza un condicional,que empieza si la lista de vehiculos no esta vacia,para luego empezar de nuevo con la variable indexativa J y el bucle,para desempaquetar cada tupla por el indice J y mostrar cada uno,y si la lista esta vacia; No hay vehiculos

PARKING CENTER S.A.S
PARQUEADEROS

PROCESOS

```
191    # OPCIÓN 4 - Ver total recaudado
192
193    elif opc == 4:
194        total = 0  # Inicializamos la variable acumuladora
195        j = 0 # Inicializamos el contador del indice para pasar a los vehiculos
196        while j < len(historial):  # Mientras el indice j sea menor a todas las tuplas de vehiculos se recorre,ya que si es mayor a la lista,estaria buscando tuplas inexistentes
197            # Recorremos toda la lista historial
198            total += historial[j][4]
199            # Sumamos el valor pagado (índice 4 de cada tupla que es valor que lo hicimos y agregamos a cada tupla en la opcion 4)
200            j += 1  # Incrementamos j para pasar al siguiente elemento osea vehiculo
201        print(f"\nTotal recaudado: ${total}\n")  # Mostramos el total acumulado
202        print(f"Dia; Hoy") # Solo sirve para dias de hoy
203
204
```

La opcion 4: Ver total recaudado o total de ingresos; Iniciamos una variable acumuladora de precio,y la variable indexadora de J junto a su bucle,y se va sumando la posicion 4 de la tupla de cada vehiculo de los vehiculos que ya salieron,osea que pagaron,para que vaya sumando de cada tupla la posicion 4,osea lo que pago,para luego mostrar el total recaudado

P|C
PARKING CENTER S.A.S
PARQUEADEROS

PROCESOS

```
205     # OPCIÓN 5 - Salir del sistema
206
207     elif opc == 5:
208         print("Saliendo del sistema...")
209         print("Has salido del sistema")
210         # Rompemos el bucle principal y terminamos el programa
211
212     # Opción inválida
213
214     else:
215         print("Opción no válida. Intente de nuevo.\n")
216
217
218
219
```

La opcion 5: Salir del sistema,donde si pone opcion 5,el bucle no correria obviamente,pero imprimos saliendo del sistema y has salido,para que el usuario sepa que ya salio,y no se quede en blanco viendo nada

Y si no puso ni 1,2,3,4,5 pues se dice opcion no valida

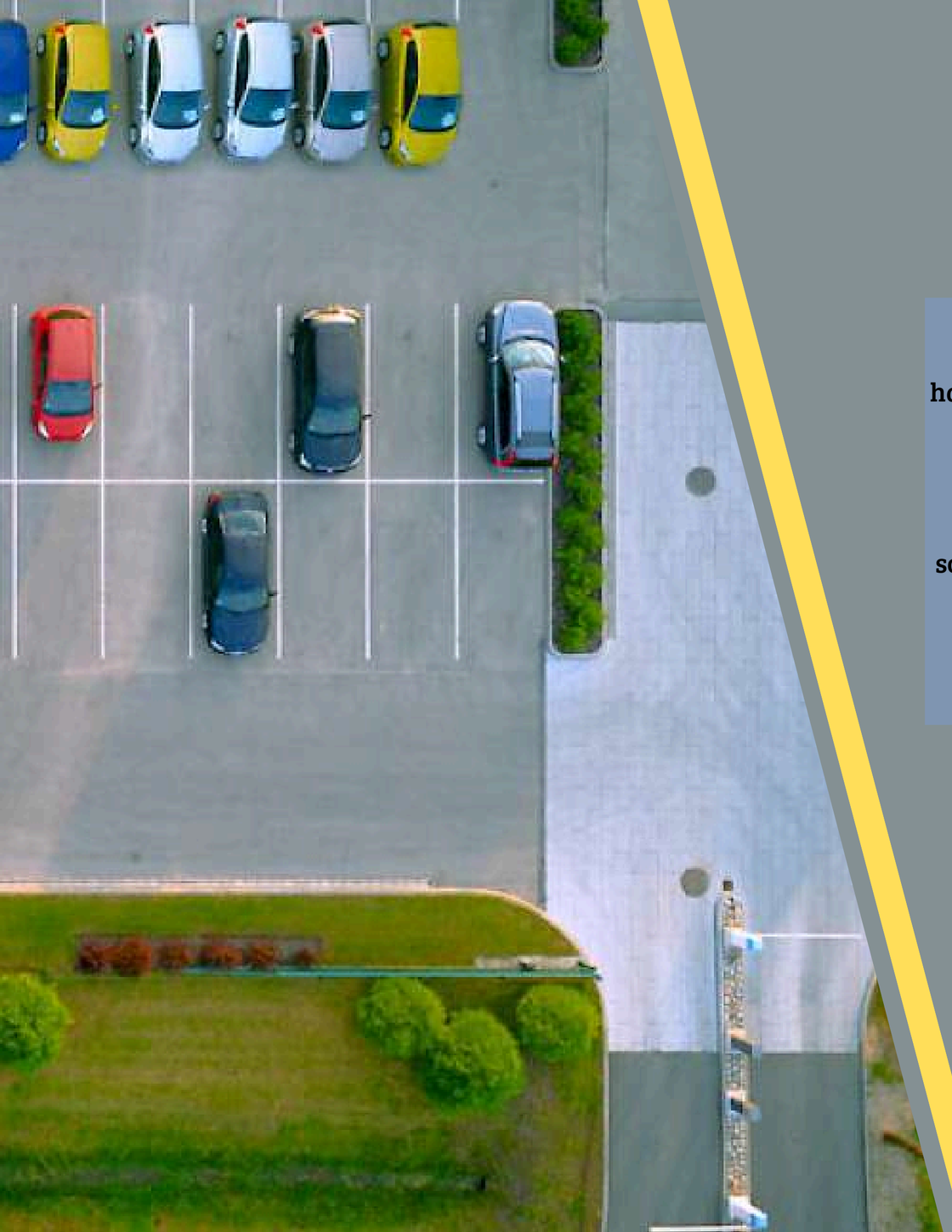
PARKING CENTER S.A.S
PARQUEADEROS

METODOS,LIBRERIAS E INVESTIGACIONES USADAS

import math	import DATETIME	.NOW()	.total_seconds()	math.ceil()
math es un módulo de funciones matemáticas en Python; en este caso se importa para usar math.ceil() y redondear hacia arriba.	datetime sirve para manejar fechas y horas completas (día, mes, año, hora, minuto, segundo); en el programa permite registrar los momentos de entrada y salida de los vehículos	.now() obtiene la fecha y hora exacta del sistema en ese instante; se usa para guardar la hora de entrada y la hora de salida.	.total_seconds() convierte un objeto timedelta en la cantidad total de segundos en número decimal; así se puede transformar el tiempo de estacionamiento en horas.	math.ceil() redondea un número decimal hacia arriba al entero más cercano; en el parqueadero asegura que cualquier fracción de hora se cobre como hora completa.

.strftime	timedelta	Desempaquetado de tupla
.strftime('%H:%M:%S') formatea una fecha/hora en texto según el formato indicado; aquí convierte las horas de entrada y salida a cadenas legibles como “14:25:08”.	timedelta: objeto que representa la diferencia entre dos fechas/horas. En nuestro programa indica cuánto tiempo estuvo un vehículo en el parqueadero.	Desempaquetado de tupla permite extraer de una tupla varios valores a variables en una sola línea; en el programa se usa para obtener placa, tipo, hora_entrada sin acceder por índices.

P C
PARKING CENTER S.A.S
PARQUEADEROS



CONCLUSION

Durante el proceso de desarrollo se reforzaron habilidades en programación, especialmente en la manipulación de listas y tuplas, además de aplicar lógica algorítmica en la solución de problemas, lo que permitió afianzar la comprensión de conceptos fundamentales de la computación.

El trabajo evidenció cómo la programación puede aplicarse en soluciones reales, logrando un sistema funcional de parqueadero y reforzando la lógica necesaria para futuros proyectos.

P|C
PARKING CENTER S.A.S
PARQUEADEROS

Este proyecto nos permitió elaborar un prototipo funcional de un sistema de parqueadero, cumpliendo con los objetivos planteados. Se logró llevar un control de los vehículos, realizar cálculos de tarifas y generar reportes, confirmando la factibilidad de la propuesta como apoyo a la gestión del personal de Parking Center S.A.S.

¡GRACIAS POR SU ATENCION!

P|C
PARKING CENTER S.A.S
PARQUEADEROS