ANALISTA
PROGRAMADOR

Evaluación Sumativa II

App Transportes

NOMBRE: GERSON CORDERO HERRERA
CARRERA: ANALISTA PROGRAMADOR (217)
ASIGNATURA: INTRODUCCION A LA PROGRAMACION
PROFESOR: JAVIER MILES AVELLO
FECHA: 21-05-2023

Contenido

1 Introducción

El transporte de pasajeros es una actividad que requiere de un control eficiente y preciso de los viajes realizados, los ingresos obtenidos y los gastos incurridos. En este trabajo se presenta el diseño y la implementación de un sistema que permite al conductor de un vehículo llevar el registro de todos estos aspectos, utilizando funciones, parámetros y retorno en el lenguaje de programación Python. El sistema cuenta con un menú que ofrece las siguientes opciones: iniciar el día de trabajo, registrar pasajeros, registrar el viaje del pasajero con su ubicación GPS, distancia y valor, mostrar el total de ingresos, calcular el gasto en combustible, mostrar la ganancia del día, mostrar la cantidad de kilómetros recorridos y mostrar los kilómetros sin pago por traslado a buscar un pasajero. El objetivo del sistema es facilitar al conductor la gestión de su actividad y optimizar sus recursos.

2 Manual de Usuario

2.1 Objetivos del sistema

El objetivo del sistema es entregarle al conductor una herramienta con la que gestionar el control de los viajes que realiza en su día. Para ello tendrá opciones tales como ingresar el vehículo con el que esta trabajando, y configurar opciones como número de pasajeros, rendimiento e identificación del vehículo.

Tambien podrán registrar las cargas de combustible que realicen, y el precio al que compran el mismo, para calcular la rentabilidad final conseguida.

2.2 Consideraciones iniciales

Para poder llevar un correcto registro, es necesario que, al comenzar el día, se le entreguen al sistema la ubicación inicial, y el nivel de combustible que hay en el estanque en porcentaje, tomando en cuenta el medidor de combustible del vehículo. A continuación, haremos una breve descripción de las pantallas de menú que hay en el programa, mencionando la funcionalidad que presenta cada opción.

```
Agregar combustible (% de estanque):60
Combustible disponible  42.0  lts
Ingrese latitud origen:-31.45642
Ingrese longitud origen:-70.45788
auto elegido = {'patente': 'BKN1234', 'marca': 'Hyundai', 'modelo':
    'Accent', 'año': 2020, 'capacidad pasajeros': 4, 'rendimiento': 14
    .3}
```

2.3 Menú Principal

```
# Menu Principal. Seleccione una opción:
 1) Vehiculos
 2) Viajes
 3) Resumen
 4) Finalizar Dia
Opción:
```

En este menú, podremos acceder a los 3 submenús del programa, desde donde podremos configurar las opciones necesarias para el funcionamiento optimo.

2.4 Menú vehículo

```
# Configuración Vehiculo. Seleccione una
    opción:
 1) Agregar Vehículo
 2) Mostrar vehiculos
 3) Modificar vehiculos
 4) Eliminar Vehículo
 5) Agregar Combustible
 6) Volver al menú principal
Opción:
```

En este submenú, podremos realizar todas las configuraciones que tienen que ver con los transportes utilizados.

Podremos agregar, mostrar, modificar y eliminar vehículos. Además de realizar cargas de combustible.

2.4.1 Agregar vehículo

Esta opción permite agregar vehículos, en caso de que se trabaje con más de uno. Los campos configurables son:

```
patente vehiculo :GGVP59
marca vehiculo :Hyundai
modelo vehiculo :Accent
Año vehiculo :2015
capacidad pasajeros :4
rendimiento vehiculo (km/l):15.3
vehiculo Agregado
```

- Patente
- Marca
- Modelo
- Año
- Capacidad de pasajeros
- Rendimiento de combustible

2.4.2 Mostrar vehículo

```
{'patente': 'BKN1234', 'marca': 'Hyundai',
    'modelo': 'Accent', 'año': 2020,
    'capacidad pasajeros': 4, 'rendimiento':
    14.3}
{'patente': 'GGVP59', 'marca': 'Hyundai',
    'modelo': 'Accent', 'año': '2015',
    'capacidad pasajeros': '4', 'rendimiento'
    : '15.3'}
```

Esta opción muestra todos los vehículos registrados.

2.4.3 Modificar vehículo

```
Ingrese patente: BKN1234
por favor ingrese los nuevos datos
marca vehiculo :Chevrolet
modelo vehiculo :Onix
año :2016
capacidad pasajeros :3
rendimiento vehiculo (km/l):14.6
Registro actualizado.
```

Esta opción solicitará el ingreso de una patente, y permitirá modificar las características ingresadas de ese vehículo.

2.4.4 Eliminar vehículo

```
Ingrese patente que quiere eliminar, esto no
    se puede deshacer: BKN1234
Registro eliminado.
```

Esta opción solicitara el ingreso de una patente, y permitirá eliminar completamente el vehículo. No es reversible

2.4.5 Agregar combustible

```
Agregar combustible (lts comprados):10
ingrese valor total pagado: $12000
Combustible total disponible  31.0  lts,
    suficiente para recorrer  3  Kms, a un
    costo de $ 1199  pesos el lt
```

Esta opción permite registrar las cargas de combustible que se realicen. Para ello solicitara el ingreso de la cantidad de litros que se agregaran al estanque, y el precio total pagado por el combustible. Esto repercutirá luego en la rentabilidad diaria lograda.

2.5 Menú viajes

```
# Configuración Viajes. Seleccione una opción
    :
 1) Agregar nuevo viaje
 2) Mostrar viajes
3) Buscar un viaje
 4) Modificar un Viaje
 5) Eliminar un viaje
 6) Ultimo Viaje
 7) Volver al menú principal
Opción:
```

En este submenú, podremos agregar, editar y eliminar viajes. Además de poder consultar el historial, o el ultimo viaje realizado

2.5.1 Agregar nuevo viaje

```
Ingrese el nombre de la persona: Estefani Fuentes Diaz
Ingrese latitud origen:-32.8245341
Ingrese longitud origen:-70.630001
Ingrese latitud destino:-33.4800001
Ingrese longitud destino:-70.6400001
```

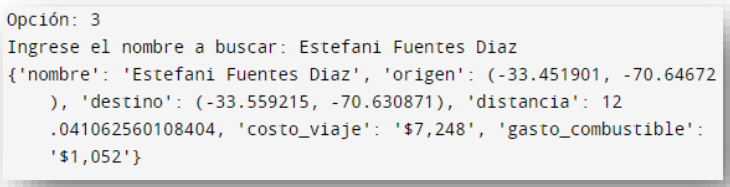
Para agregar un nuevo viaje, debemos conocer el nombre del pasajero, las coordenadas de origen del viaje, y las coordenadas de destino.

2.5.2 Mostrar viajes

```
{'nombre': 'Viaje Inicio', 'origen': (-32.78124, -70.610054),
    'destino': (-32.78124, -70.610054), 'distancia': 0.0,
    'costo_viaje': '$0', 'gasto_combustible': '$0'}
{'nombre': 'Estefani Fuentes Diaz', 'origen': (-32.8245341, -70
    .630001), 'destino': (-33.4800001, -70.6400001), 'distancia': 72
    .76519124190499, 'costo_viaje': '$43,804', 'gasto_combustible':
    '$6,360'}
{'nombre': 'Traslado entre viajes', 'origen': (-32.78124, -70.610054
    ), 'destino': (-32.8245341, -70.630001), 'distancia': 5
    .291175570404131, 'costo_viaje': '$0', 'gasto_combustible':
    '$462'}
{'nombre': 'Gerson Cordero Herrera', 'origen': (-33.451901, -70
    .64672), 'destino': (-33.559215, -70.630871), 'distancia': 12
    .041062560108404, 'costo_viaje': '$7,248', 'gasto_combustible':
    '$1,052'}
{'nombre': 'Traslado entre viajes', 'origen': (-32.8245341, -70
    .630001), 'destino': (-33.451901, -70.64672), 'distancia': 69
    .66244968107273, 'costo_viaje': '$0', 'gasto_combustible': '$6
    ,089'}
```

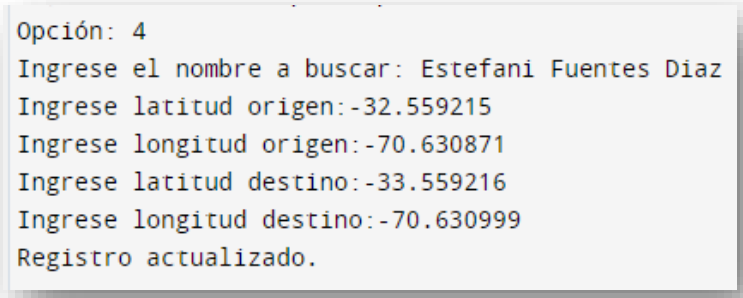
Con la opción 2, podemos ver un resumen de todos los viajes realizados, y sus datos de origen, destino, gasto de combustible, distancia recorrida, y costo total del viaje.

2.5.3 Buscar viajes



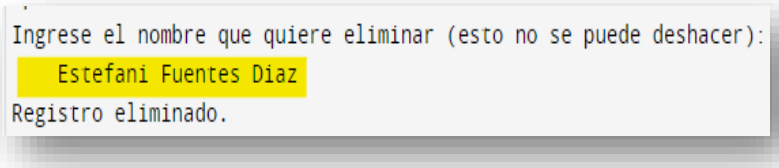
Con la opción 3, podemos buscar un viaje en específico, para ello necesitamos conocer el nombre del pasajero.

2.5.4 Modificar un viaje



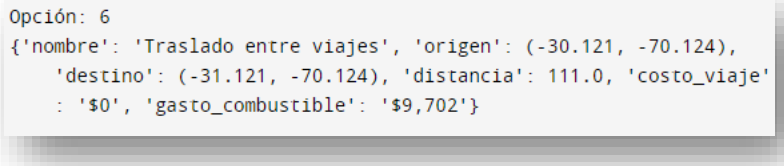
Con la opción 4, podemos editar un viaje. Para ello necesitamos conocer el nombre del pasajero. Luego podremos ingresar nuevamente los datos, para que sean actualizados.

2.5.5 Eliminar un viaje



Con la opción 5, podemos eliminar un viaje. Para ello necesitamos conocer el nombre del pasajero. Esto no se puede reversar.

2.5.6 Último viaje



Con la opción 6, veremos el ultimo viaje que se encuentre registrado.

2.6 Menú resumen

```
# Resumen Diario. Seleccione una opción:
 1) Detalle viajes
 2) Resumen diario
 3) Volver al menú principal
Opción:
```

En este submenú, podremos de manera detallada los viajes realizados, y un resumen con el total de km, viajes realizados, combustible gastado y ganancias obtenidas.

2.6.1 Detalle viajes


```
{'nombre': 'Viaje Inicio', 'origen': (-32.78124, -70.610054),
    'destino': (-32.78124, -70.610054), 'distancia': 0.0,
    'costo_viaje': '$0', 'gasto_combustible': '$0'}
{'nombre': 'Estefani Fuentes Diaz', 'origen': (-32.8245341, -70
    .630001), 'destino': (-33.4800001, -70.6400001), 'distancia': 72
    .76519124190499, 'costo_viaje': '$43,804', 'gasto_combustible':
    '$6,360'}
{'nombre': 'Traslado entre viajes', 'origen': (-32.78124, -70.610054
    ), 'destino': (-32.8245341, -70.630001), 'distancia': 5
    .291175570404131, 'costo_viaje': '$0', 'gasto_combustible':
    '$462'}
{'nombre': 'Gerson Cordero Herrera', 'origen': (-33.451901, -70
    .64672), 'destino': (-33.559215, -70.630871), 'distancia': 12
    .041062560108404, 'costo_viaje': '$7,248', 'gasto_combustible':
    '$1,052'}
{'nombre': 'Traslado entre viajes', 'origen': (-32.8245341, -70
    .630001), 'destino': (-33.451901, -70.64672), 'distancia': 69
    .66244968107273, 'costo_viaje': '$0', 'gasto_combustible': '$6
    ,089'}
```


En la opción 1, veremos el detalle general de cada viaje. Tambien podremos ver el detalle de los traslados entre viaje, que nos ayudaran a conocer cuanto fue el gasto realizado en estos traslados, para deducirlo de las ganancias.

2.6.2 Resumen diario

```
# Metricas
Número de viajes realizados: 3
Traslados sin ganancias: 106 KM
Distancia total recorrida: 195 KM
Combustible restante: 28 lts

# Ganancias
Monto total recaudado: $55,602
Gasto en combustible: $17,377
Impuestos: $10,564
Ganancia final del Conductor: $27,660 Neto
```

En la opción 2, veremos un resumen, con el total de viajes realizados, los kilómetros recorridos, los traslados sin ganancias, las ganancias totales, y las deducciones de impuestos y combustibles.

Finalmente podremos ver la ganancia neta que obtendrá el conductor.

3 metodología cálculos

A continuación, expongo una tabla con las formulas utilizadas para hacer los cálculos principales utilizadas en este programa.

Concepto	¿Como se calcula?
Distancia	Teorema de pitagoras para calcular con las latitudes y longitudes, de inicio y fin del viaje.
Distancia entre Viajes	Teorema de pitagoras para calcular con las latitudes y longitudes, del fin del viaje anterior, y en inicio del viaje actual.
Costo de cada viaje	Distancia * costo_mt
Combustible Gastado	(Distancia/rendimiento_vehiculo)*valor_combustible
Iva	Costo de cada viaje *0,19
Ganancia	Costo de cada viaje-combustible gastado-Iva

Variables	Explicacion
costo_mt	Costo de cada mt, utilizado para calcular costo de viajes.
rendimiento_vehiculo	Rendimiento de cada vehiculo, utilizado para calcular rendimiento de viajes.
valor_combustible	Valor del combustible al momento de la carga, utilizado para calcular valor de viajes.
estanque_combustible	capacidad de estanque de combustible, utilizado solo en la configuracion inicial.

4 Conclusión

En este trabajo se ha presentado el diseño y la implementación de un sistema que permite al conductor de un vehículo llevar el control de los viajes realizados, los ingresos obtenidos y los gastos incurridos, utilizando funciones, parámetros y retorno en el lenguaje de programación Python. El sistema cuenta con un menú que ofrece diversas opciones para configurar el vehículo, registrar los pasajeros y sus viajes, mostrar el resumen diario y finalizar el día de trabajo. Tambien facilita al conductor la gestión de su actividad y optimiza sus recursos, al calcular la distancia, el costo y el gasto en combustible de cada viaje, así como la ganancia final del día. Como posibles mejoras o líneas de investigación futuras, se podría incorporar una interfaz gráfica más amigable para el usuario, una conexión con una base de datos para almacenar la información de forma permanente y una integración con servicios web o aplicaciones móviles para ofrecer más funcionalidades al conductor. Toda la programación se ha realizado con funciones, listas y diccionarios, como hemos visto en las ultimas clases.

5 Código

5.1 Versión Online

En el siguiente link, hay una versión Online del código utilizado en este trabajo.

https://raw.githubusercontent.com/Gers0n23/Inacap/main/UberFruna 3 (casi final)

5.2 Código completo

Este es el código completo escrito para este trabajo:

```
personas=[]
auto5=[]
costo_mt = 602
rendimiento_vehiculo=14.3
valor_combustible=1250
vacio= "pendiente"
Estanque_Combustible=70

import math
from math import sqrt

def mostrar_menu(nombre, opciones):
    print(f'# {nombre}. Seleccione una opción:')
    for clave in sorted(opciones):
        print(f' {clave}) {opciones[clave][0]}')
def leer_opcion(opciones):
    while (a := input('Opción: ')) not in opciones:
        print('Opción incorrecta, vuelva a intentarlo.')
    return a

def ejecutar_opcion(opcion, opciones):
    opciones[opcion][1]()

def generar_menu(nombre, opciones, opcion_salida):
    opcion = None
    while opcion != opcion_salida:
        mostrar_menu(nombre, opciones)
        opcion = leer_opcion(opciones)
        ejecutar_opcion(opcion, opciones)
        print()

def menu_inicio():
    opciones = {
        '1': ('Iniciar Dia >', iniciar)
    }

    generar_menu('Menú inicio', opciones, '1')
def menu_principal():
    opciones = {
        '1': ('Vehiculos', menu_vehiculo),
        '2': ('Viajes', menu_viajes),
        '3': ('Resumen', menu_resumen),
        '4': ('Finalizar Dia', salir)
    }

    generar_menu('Menu Principal', opciones, '4')

def menu_vehiculo():
    opciones = {
        '1': ('Agregar Vehículo', configurar_vehiculo),
```

```
        '2': ('Mostrar vehiculos', mostrar_vehiculos),
        '3': ('Modificar vehiculos', modifica_vehiculo1),
        '4': ('Eliminar Vehículo', elimina_auto),
        '5': ('Agregar Combustible', comprar_combustible),
        '6': ('Volver al menú principal', menu_principal)
    }

    generar_menu('Configuración Vehiculo', opciones, '6')

def menu_viajes():
    opciones = {
        '1': ('Agregar nuevo viaje', nuevo_viaje),
        '2': ('Mostrar viajes', mostrar_viajes),
        '3': ('Buscar un viaje',busca_viaje1),
        '4': ('Modificar un Viaje', modifica_viaje1),
        '5': ('Eliminar un viaje', elimina_viaje1),
        '6': ('Ultimo Viaje', obtener_ultimo_destino),
        '7': ('Volver al menú principal', menu_principal)
    }

    generar_menu('Configuración Viajes', opciones, '7')

def menu_resumen():
    opciones = {
        '1': ('Detalle viajes', mostrar_viajes),
        '2': ('Resumen diario', resumen_viajes),
        '3': ('Volver al menú principal', menu_principal)
    }

    generar_menu('Resumen Diario', opciones, '3')

# A partir de aquí estan todas las funciones de las opciones de los menu

def funcion1():
    print('Falta Construir')

def agregar_punto_inicio(tipo):
    georeferencia_i = {}
    georeferencia_i['latitud'] = float(input(f"Ingrese latitud {tipo}:"))
    georeferencia_i['longitud'] = float(input(f"Ingrese longitud {tipo}:"))
    return georeferencia_i

def agregar_georeferencia(tipo):
    latitud = float(input(f"Ingrese latitud {tipo}:"))
    longitud = float(input(f"Ingrese longitud {tipo}:"))
    return (latitud, longitud)

def agregar_combustible(tipo):
    global Estanque_Combustible
    Estanque_Combustible = float(input(f"Agregar combustible {tipo}:"))
    return (Estanque_Combustible)

def iniciar():
    persona = {}
    Estanque_Combustible=agregar_combustible('(% de estanque)')/100*70
    print("Combustible disponible ",Estanque_Combustible," lts")
    persona['nombre'] = "Viaje Inicio"
    persona['origen'] = agregar_georeferencia('origen')
    persona['destino'] = persona['origen']
    x1, y1 = persona['origen']
    x2, y2 = persona['destino']
```

```
    distancia = sqrt((x2 - x1)**2 + (y2 - y1)**2)*111
    costo_viaje = distancia * costo_mt
    gasto_combustible = distancia / rendimiento_vehiculo*valor_combustible
    persona['distancia'] = distancia
    persona['costo_viaje'] = f"${int(0):,}"
    persona['gasto_combustible'] = f"${int(gasto_combustible):,}"
    personas.append(persona)
    auto =
{"patente":"BKN1234","marca":"Hyundai","modelo":"Accent","año":2020,"capacidad
pasajeros":4,"rendimiento":rendimiento_vehiculo}
    auto5.append(auto)
    for auto in auto5:
        print("auto elegido =",auto)
    menu_principal()

def obtener_ultimo_destino():
    if personas:
        ultimo_registro = personas[-1]
        print(ultimo_registro)
        return ultimo_registro['destino']
    else:
        return None

def nuevo_viaje():
    persona = {}
    persona['nombre'] = input("Ingrese el nombre de la persona: ")
    persona['origen'] = agregar_georeferencia('origen')
    persona['destino'] = agregar_georeferencia('destino')
    x1, y1 = persona['origen']
    x2, y2 = persona['destino']
    distancia = sqrt((x2 - x1)**2 + (y2 - y1)**2)*111
    costo_viaje = distancia * costo_mt
    gasto_combustible = distancia / rendimiento_vehiculo*valor_combustible
    persona['distancia'] = distancia
    persona['costo_viaje'] = f"${int(costo_viaje):,}"

    persona['gasto_combustible'] = f"${int(gasto_combustible):,}"
    personas.append(persona)

    traslado_entre_viajes()

def traslado_entre_viajes():
    if len(personas) >= 2:
        penultimo = personas[-2]
        ultimo = personas[-1]
        x1, y1 = penultimo['destino']
        x2, y2 = ultimo['origen']
        distancia_last = math.sqrt((x2 - x1)**2 + (y2 - y1)**2)*111

        persona = {}
        persona['nombre'] = "Traslado entre viajes"
        persona['origen'] = penultimo['destino']
        persona['destino'] = ultimo['origen']
        costo_viaje = distancia_last * costo_mt
        gasto_combustible_last = distancia_last /
rendimiento_vehiculo*valor_combustible
        persona['distancia'] = distancia_last
        persona['costo_viaje'] = f"${int(0):,}"

        persona['gasto_combustible'] = f"${int(gasto_combustible_last):,}"
        personas.append(persona)
```

```
        return distancia_last

    else:
        return None

def mostrar_viajes():
    for persona in personas:
        print(persona)

def busca_viaje1():
    nombre=nombre = input("Ingrese el nombre a buscar: ")
    buscar_viaje(nombre)
def buscar_viaje(nombre):
    for persona in personas:
        if persona['nombre'] == nombre:
            print(persona)

def modifica_viaje1():
    nombre=nombre = input("Ingrese el nombre a buscar: ")
    modificar_viaje(nombre)
def modificar_viaje(nombre):
    for persona in personas:
        if persona['nombre'] == nombre:
            nuevo_origen = agregar_georeferencia('origen')
            nuevo_destino = agregar_georeferencia('destino')
            persona['origen'] = nuevo_origen
            persona['destino'] = nuevo_destino
            print("Registro actualizado.")
            return
    print("No se encontró el registro.")

def elimina_viaje1():
    nombre=nombre = input("Ingrese el nombre que quiere eliminar (esto no se
puede deshacer): ")
    eliminar_registro(nombre)
def eliminar_registro(nombre):
    for i, persona in enumerate(personas):
        if persona['nombre'] == nombre:
            del personas[i]
            print("Registro eliminado.")
            return
    print("No se encontró el registro.")

def resumen_viajes():
    global Estanque_Combustible
    global rendimiento_vehiculo

    suma_distancias = sum(p['distancia'] for p in personas if p['nombre'] ==
'Traslado entre viajes')

    viajes_sin_pago = sum (1 for p in personas if p['nombre'] == 'Traslado
entre viajes')

    total_km = sum([int(persona['distancia']) for persona in personas])
    total_costo =
sum([int(persona['costo_viaje'].replace('$','').replace(',','')) for persona in
personas]) # Suma de los costos de todos los viajes
    gasto_combustible_total =
sum(round(float(persona['gasto_combustible'].replace('$','').replace(',','')),
0) for persona in personas)
```

```
    impuestos= total_costo*0.19

    print("# Metricas")
    print(f"Número de viajes realizados: {len(personas)-int(viajes_sin_pago)-
1}") # Conteo del número total de viajes
    print(f"Traslados sin ganancias: {int(suma_distancias)} KM")
    print(f"Distancia total recorrida: {total_km} KM")
    print(f"Combustible restante: {int((Estanque_Combustible/100*70)-
(total_km/rendimiento_vehiculo))} lts")
    print("")
    print("# Ganancias")
    print(f"Monto total recaudado: ${total_costo:,}")
    print(f"Gasto en combustible: ${int(gasto_combustible_total):,}")
    print(f"Impuestos: ${int(impuestos):,}")
    print(f"Ganancia final del Conductor: ${int(total_costo -
gasto_combustible_total-impuestos):,} Neto")

    return total_costo
def resumen_gasto_combustible():
    gasto_combustible_total =
sum(round(float(persona['gasto_combustible'].replace('$','').replace(',','')),
0) for persona in personas)

    print(gasto_combustible_total)
    return gasto_combustible_total

def configurar_vehiculo():
    auto = {}
    auto['patente'] = input("patente vehiculo :")
    auto['marca'] = input("marca vehiculo :")
    auto['modelo'] = input("modelo vehiculo :")
    auto['año'] = input("Año vehiculo :")
    auto['capacidad pasajeros'] = input("capacidad pasajeros :")
    rendimiento_vehiculo=input("rendimiento vehiculo (km/l):")
    auto['rendimiento'] = rendimiento_vehiculo
    auto5.append(auto)
    print("vehiculo Agregado")

def mostrar_vehiculos():
    for auto in auto5:
        print(auto)

def modifica_vehiculo1():
    patauto=patauto = input("Ingrese patente: ")
    modificar_vehiculo(patauto)
def modificar_vehiculo(patauto):
    for auto in auto5:
        if auto['patente'] == patauto:
            print("por favor ingrese los nuevos datos")
            nueva_marca = input("marca vehiculo :")
            nuevo_modelo = input("modelo vehiculo :")
            nuevo_año = input("año :")
            nuevo_cap = input("capacidad pasajeros :")
            rendimiento_vehiculo=input("rendimiento vehiculo (km/l):")

            auto['marca'] = nueva_marca
            auto['modelo'] = nuevo_modelo
            auto['año'] = nuevo_año
            auto['capacidad pasajeros'] = nuevo_cap
            auto['rendimiento'] = rendimiento_vehiculo
```

```
            print("Registro actualizado.")
            return
    print("No se encontró el registro.")

def elimina_auto():
    patauto=patauto = input("Ingrese patente que quiere eliminar, esto no se
puede deshacer: ")
    eliminar_registroauto(patauto)
def eliminar_registroauto(patauto):
    for i, auto in enumerate(auto5):
        if auto['patente'] == patauto:
            del auto5[i]
            print("Registro eliminado.")
            return
    print("No se encontró el registro.")

def comprar_combustible():
    global Estanque_Combustible
    global rendimiento_vehiculo
    global valor_combustible
    com_restante=Estanque_Combustible
    nuevo_combustible=int(agregar_combustible('(lts comprados)'))*100/70
    valor_combustible= int(input("ingrese valor total pagado:
$"))/(nuevo_combustible/100*70)
    Estanque_Combustible = com_restante + nuevo_combustible
    print("Combustible total disponible ",Estanque_Combustible/100*70," lts,
suficiente para recorrer ",int(Estanque_Combustible/rendimiento_vehiculo), "
Kms, a un costo de $",int(valor_combustible), " pesos el lt")

def salir():
    print('Finalizando Sesion, Hasta Pronto')

if __name__ == '__main__':
    menu_inicio()
```