# 

# **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](#_Toc135604383)

[**Evaluación Sumativa II** 1](#_Toc135604384)

[App Transportes 1](#_Toc135604385)

[1 Introducción 2](#_Toc135604386)

[2 Manual de Usuario 3](#_Toc135604387)

[**2.1** **Objetivos del sistema** 3](#_Toc135604388)

[**2.2** **Consideraciones iniciales** 3](#_Toc135604389)

[**2.3** **Menú Principal** 4](#_Toc135604390)

[**2.4** **Menú vehículo** 4](#_Toc135604391)

[2.4.1 Agregar vehículo 4](#_Toc135604392)

[2.4.2 Mostrar vehículo 5](#_Toc135604393)

[2.4.3 Modificar vehículo 5](#_Toc135604394)

[2.4.4 Eliminar vehículo 5](#_Toc135604395)

[2.4.5 Agregar combustible 5](#_Toc135604396)

[**2.5** **Menú viajes** 6](#_Toc135604397)

[2.5.1 Agregar nuevo viaje 6](#_Toc135604398)

[2.5.2 Mostrar viajes 6](#_Toc135604399)

[2.5.3 Buscar viajes 7](#_Toc135604400)

[2.5.4 Modificar un viaje 7](#_Toc135604401)

[2.5.5 Eliminar un viaje 7](#_Toc135604402)

[2.5.6 Último viaje 7](#_Toc135604403)

[**2.6** **Menú resumen** 8](#_Toc135604404)

[2.6.1 Detalle viajes 8](#_Toc135604405)

[2.6.2 Resumen diario 8](#_Toc135604406)

[3 metodología cálculos 9](#_Toc135604407)

[4 Conclusión 10](#_Toc135604408)

[5 Código 11](#_Toc135604409)

[**5.1** **Versión Online** 11](#_Toc135604410)

[**5.2** **Código completo** 11](#_Toc135604411)

# 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.

# Manual de Usuario

## **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.

## **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.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

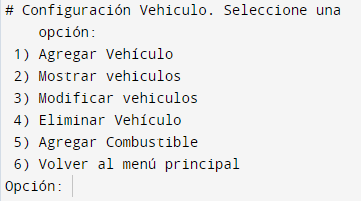
## **Menú Principal**

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

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

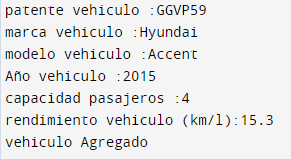
## **Menú vehículo**

 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.

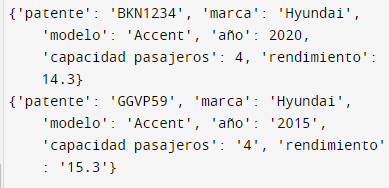
### Agregar vehículo

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



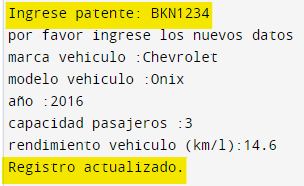
* Patente
* Marca
* Modelo
* Año
* Capacidad de pasajeros
* Rendimiento de combustible

### Mostrar vehículo



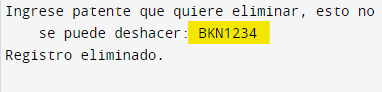
Esta opción muestra todos los vehículos registrados.

### Modificar vehículo



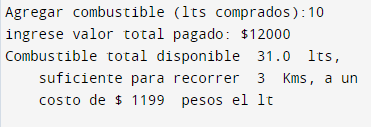
Esta opción solicitará el ingreso de una patente, y permitirá modificar las características ingresadas de ese vehículo.

### Eliminar vehículo



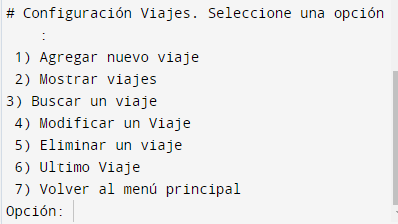
Esta opción solicitara el ingreso de una patente, y permitirá eliminar completamente el vehículo. No es reversible

### Agregar combustible



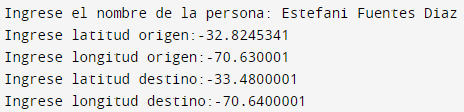
Esta opción permite registrar las cargas de combustible que se realicen. Para ello solicitara el ingreso de la cantidad de litros que se agregaran al estanque, y el precio total pagado por el combustible. Esto repercutirá luego en la rentabilidad diaria lograda.

## **Menú viajes**



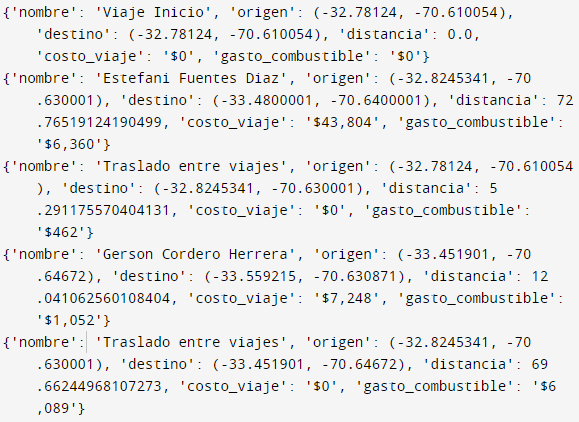
En este submenú, podremos agregar, editar y eliminar viajes. Además de poder consultar el historial, o el ultimo viaje realizado

### Agregar nuevo viaje



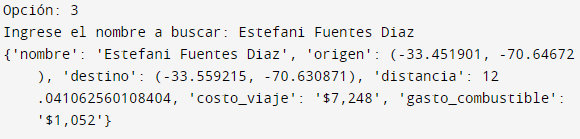
Para agregar un nuevo viaje, debemos conocer el nombre del pasajero, las coordenadas de origen del viaje, y las coordenadas de destino.

### Mostrar viajes



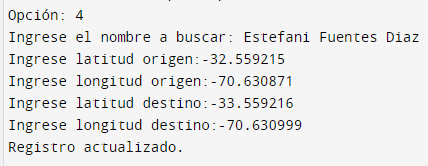
Con la opción 2, podemos ver un resumen de todos los viajes realizados, y sus datos de origen, destino, gasto de combustible, distancia recorrida, y costo total del viaje.

### Buscar viajes



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

### Modificar un viaje



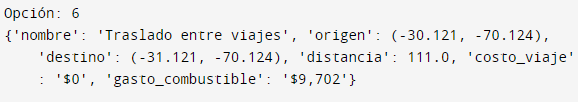
Con la opción 4, podemos editar un viaje. Para ello necesitamos conocer el nombre del pasajero. Luego podremos ingresar nuevamente los datos, para que sean actualizados.

### Eliminar un viaje

Imagen que contiene Texto

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

### Último viaje



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

## **Menú resumen**

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

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.

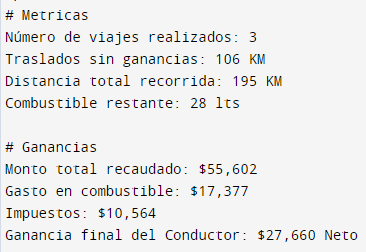
### Detalle viajes

Texto, Carta

Descripción generada automáticamente

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.

### Resumen diario

 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.

# 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. |

# 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.

# Código

## **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)](https://raw.githubusercontent.com/Gers0n23/Inacap/main/UberFruna%203%20(casi%20final))

## **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()