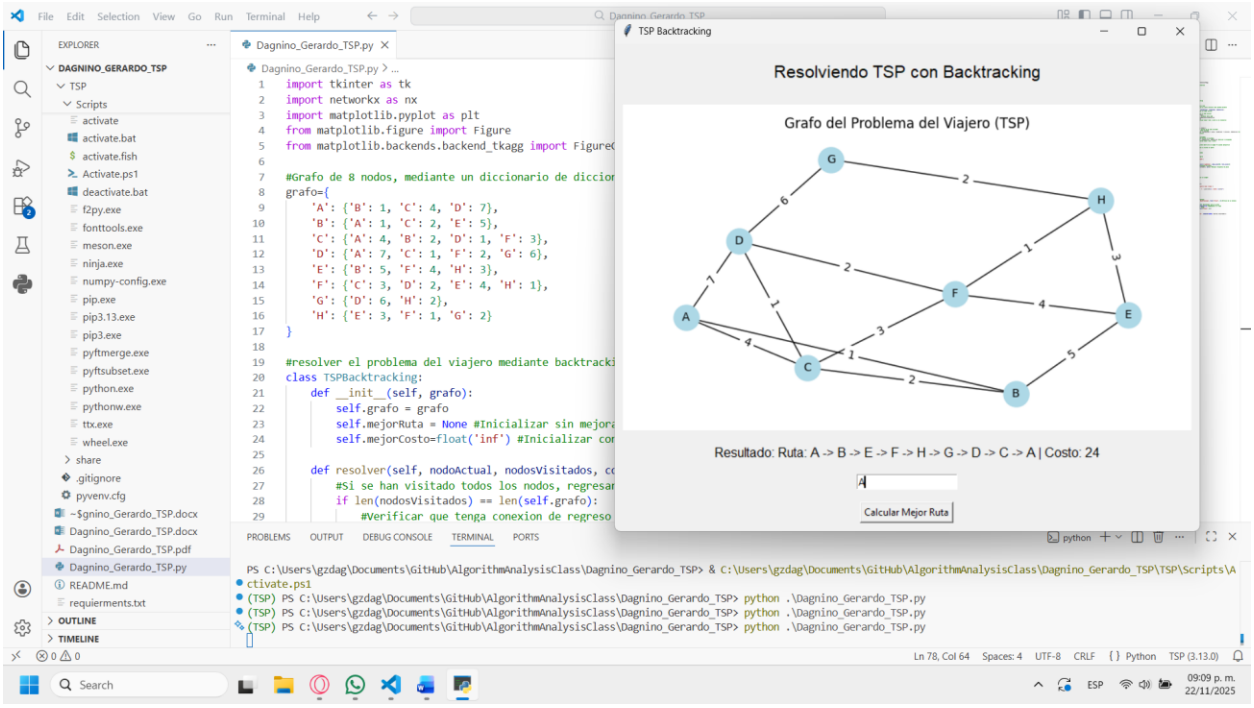
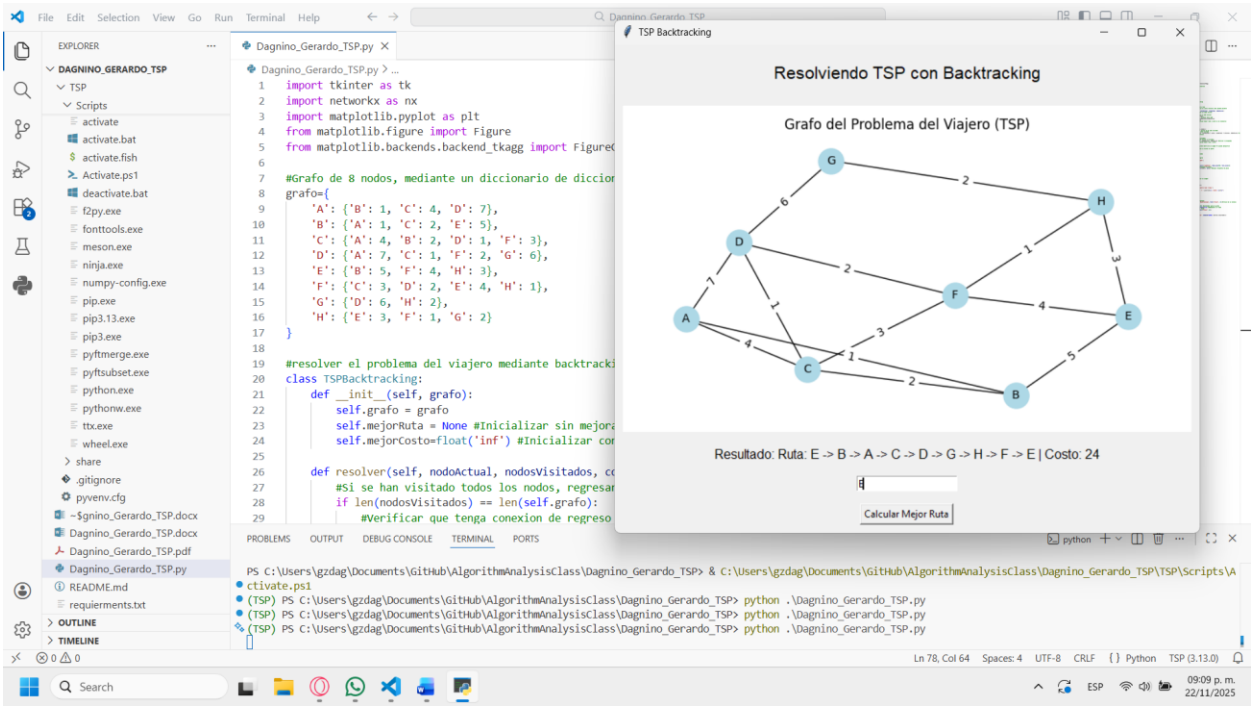
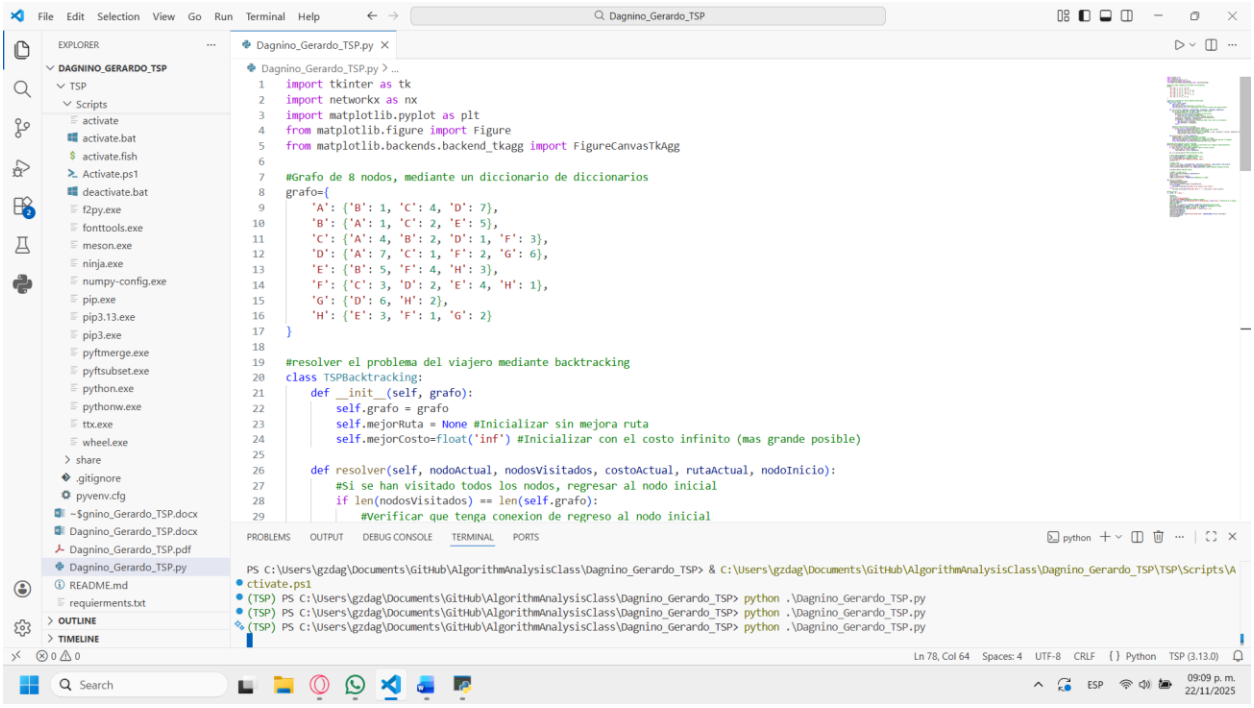
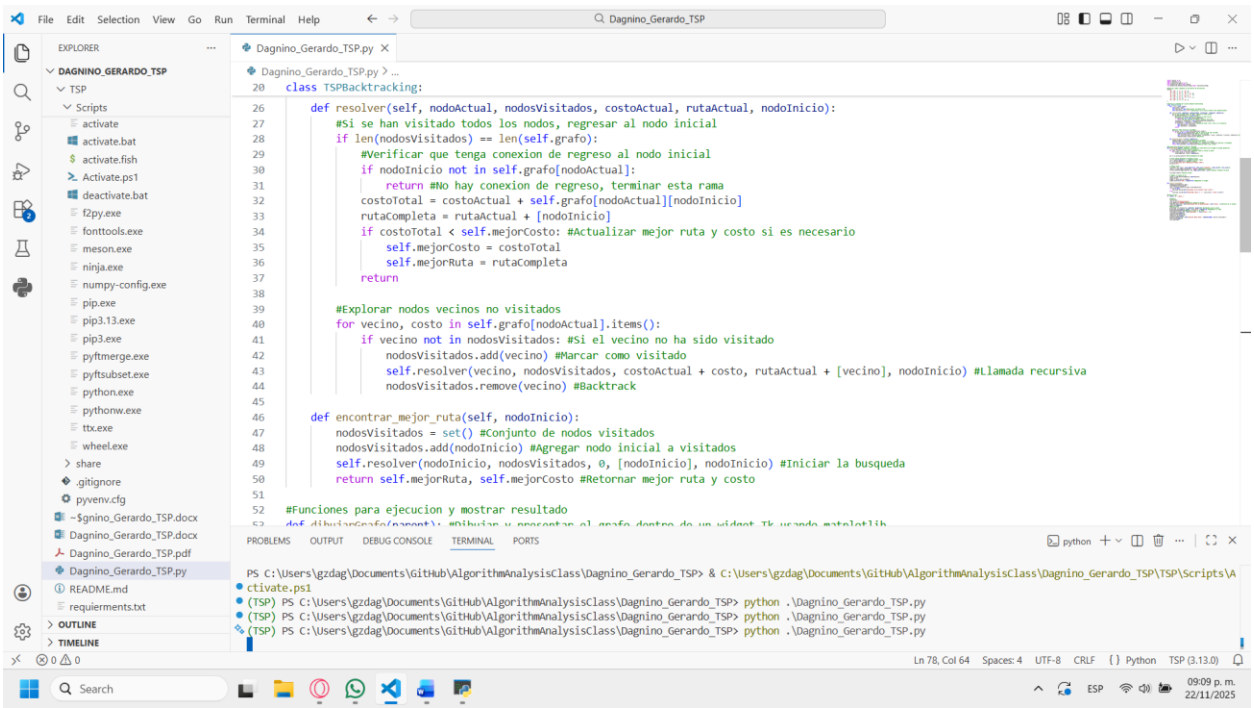
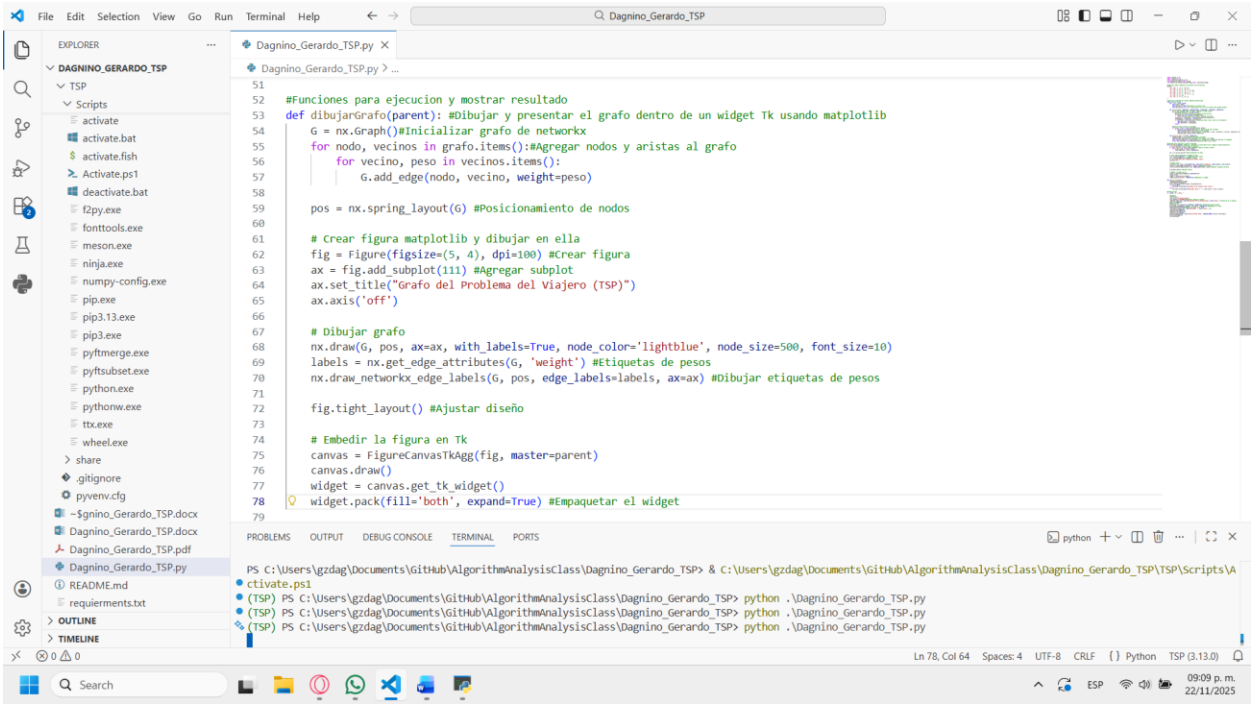
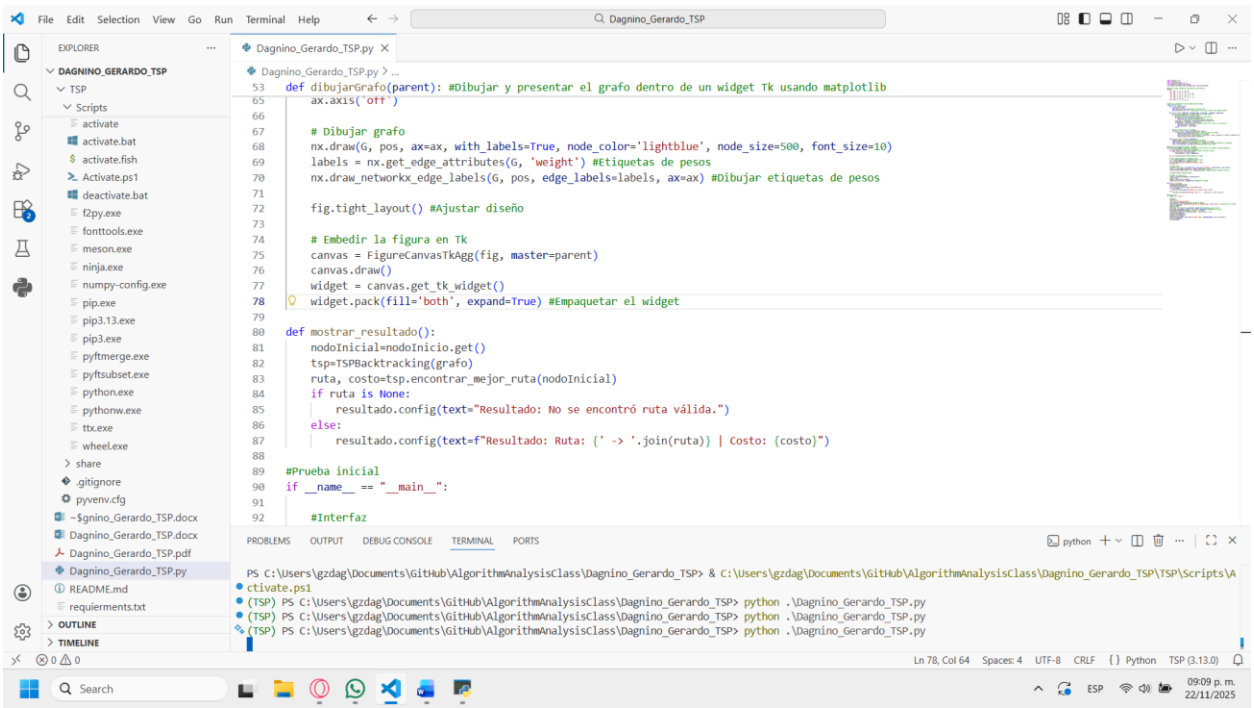

TSP Backtracking
Resolviendo TSP con Backtracking
Grafo del Problema del Viajero (TSP)
Resultado: Ruta: A -> B -> E -> F -> H -> G -> D -> C -> A | Costo: 24
Calcular Mejor Ruta

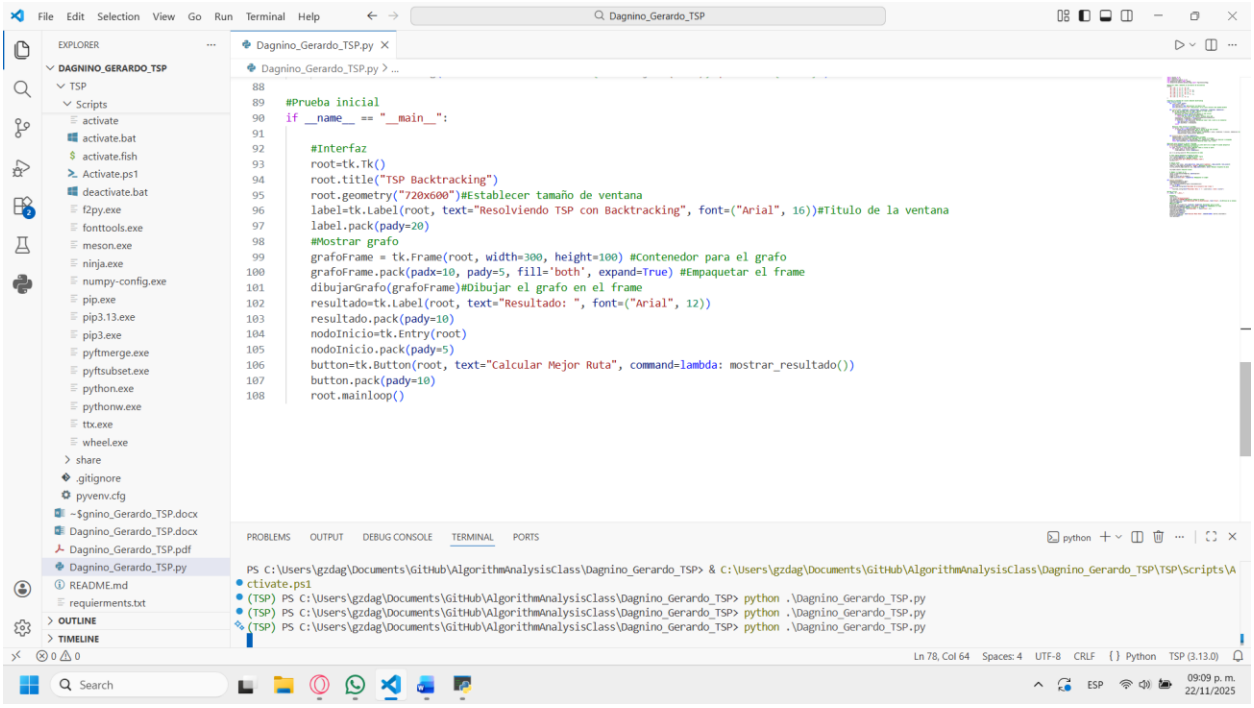

TSP Backtracking
Resolviendo TSP con Backtracking
Grafo del Problema del Viajero (TSP)
Resultado: Ruta: E -> B -> A -> C -> D -> G -> H -> F -> E | Costo: 24
Calcular Mejor Ruta


File Edit Selection View Go Run Terminal Help
Dagnino_Gerardo_TSP
EXPLORER
DAGNINO_GERARDO_TSP
TSP
Scripts
activate
activate.bat
activate.fish
Activate.ps1
deactivate.bat
f2py.exe
fonttools.exe
meson.exe
ninja.exe
numpy-config.exe
pip.exe
pip3.13.exe
pip3.exe
pyftmerge.exe
pyftsubset.exe
python.exe
pythonw.exe
ttx.exe
wheel.exe
share
.gitignore
pyvenv.cfg
~$gnino_Gerardo_TSP.docx
Dagnino_Gerardo_TSP.docx
Dagnino_Gerardo_TSP.pdf
Dagnino_Gerardo_TSP.py
README.md
requierments.txt
OUTLINE
TIMELINE
Dagnino_Gerardo_TSP.py
import tkinter as tk
import networkx as nx
import matplotlib.pyplot as plt
from matplotlib.figure import Figure
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
#Grafo de 8 nodos, mediante un diccionario de diccionarios
grafo={
'A': {'B': 1, 'C': 4, 'D': 7},
'B': {'A': 1, 'C': 2, 'E': 5},
'C': {'A': 4, 'B': 2, 'D': 1, 'F': 3},
'D': {'A': 7, 'C': 1, 'F': 2, 'G': 6},
'E': {'B': 5, 'F': 4, 'H': 3},
'F': {'C': 3, 'D': 2, 'E': 4, 'H': 1},
'G': {'D': 6, 'H': 2},
'H': {'E': 3, 'F': 1, 'G': 2}
}
#resolver el problema del viajero mediante backtracking
class TSPBacktracking:
def __init__(self, grafo):
self.grafo = grafo
self.mejorRuta = None #Inicializar sin mejora ruta
self.mejorCosto=float('inf') #Inicializar con el costo infinito (mas grande posible)
def resolver(self, nodoActual, nodosVisitados, costoActual, rutaActual, nodoInicio):
#Si se han visitado todos los nodos, regresar al nodo inicial
if len(nodosVisitados) == len(self.grafo):
#Verificar que tenga conexion de regreso al nodo inicial
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
python
PS C:\Users\gzdag\Documents\GitHub\AlgorithmAnalysisClass\Dagnino_Gerardo_TSP> & C:\Users\gzdag\Documents\GitHub\AlgorithmAnalysisClass\Dagnino_Gerardo_TSP\TSP\Scripts\Activate.ps1
(TSP) PS C:\Users\gzdag\Documents\GitHub\AlgorithmAnalysisClass\Dagnino_Gerardo_TSP> python .\Dagnino_Gerardo_TSP.py
(TSP) PS C:\Users\gzdag\Documents\GitHub\AlgorithmAnalysisClass\Dagnino_Gerardo_TSP> python .\Dagnino_Gerardo_TSP.py
(TSP) PS C:\Users\gzdag\Documents\GitHub\AlgorithmAnalysisClass\Dagnino_Gerardo_TSP> python .\Dagnino_Gerardo_TSP.py
Ln 78, Col 64 Spaces: 4 UTF-8 CRLF Python TSP (3.13.0)
Search
ESP
09:09 p. m.
22/11/2025


File Edit Selection View Go Run Terminal Help
Dagnino_Gerardo_TSP
EXPLORER
DAGNINO_GERARDO_TSP
TSP
Scripts
activate
activate.bat
activate.fish
Activate.ps1
deactivate.bat
f2py.exe
fonttools.exe
meson.exe
ninja.exe
numpy-config.exe
pip.exe
pip3.13.exe
pip3.exe
pyftmerge.exe
pyftsubset.exe
python.exe
pythonw.exe
ttx.exe
wheel.exe
share
.gitignore
pyvenv.cfg
~$gnino_Gerardo_TSP.docx
Dagnino_Gerardo_TSP.docx
Dagnino_Gerardo_TSP.pdf
Dagnino_Gerardo_TSP.py
README.md
requierments.txt
OUTLINE
TIMELINE
Dagnino_Gerardo_TSP.py
class TSPBacktracking:
def resolver(self, nodoActual, nodosVisitados, costoActual, rutaActual, nodoInicio):
#Si se han visitado todos los nodos, regresar al nodo inicial
if len(nodosVisitados) == len(self.grafo):
#Verificar que tenga conexion de regreso al nodo inicial
if nodoInicio not in self.grafo[nodoActual]:
return #No hay conexion de regreso, terminar esta rama
costoTotal = costoActual + self.grafo[nodoActual][nodoInicio]
rutaCompleta = rutaActual + [nodoInicio]
if costoTotal < self.mejorCosto: #Actualizar mejor ruta y costo si es necesario
self.mejorCosto = costoTotal
self.mejorRuta = rutaCompleta
return
#Explorar nodos vecinos no visitados
for vecino, costo in self.grafo[nodoActual].items():
if vecino not in nodosVisitados: #Si el vecino no ha sido visitado
nodosVisitados.add(vecino) #Marcar como visitado
self.resolver(vecino, nodosVisitados, costoActual + costo, rutaActual + [vecino], nodoInicio) #Llamada recursiva
nodosVisitados.remove(vecino) #Backtrack
def encontrar_mejor_ruta(self, nodoInicio):
nodosVisitados = set() #Conjunto de nodos visitados
nodosVisitados.add(nodoInicio) #Agregar nodo inicial a visitados
self.resolver(nodoInicio, nodosVisitados, 0, [nodoInicio], nodoInicio) #Iniciar la busqueda
return self.mejorRuta, self.mejorCosto #Retornar mejor ruta y costo
#Funciones para ejecucion y mostrar resultado
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
python
PS C:\Users\gzdag\Documents\GitHub\AlgorithmAnalysisClass\Dagnino_Gerardo_TSP> & C:\Users\gzdag\Documents\GitHub\AlgorithmAnalysisClass\Dagnino_Gerardo_TSP\TSP\Scripts\Activate.ps1
(TSP) PS C:\Users\gzdag\Documents\GitHub\AlgorithmAnalysisClass\Dagnino_Gerardo_TSP> python .\Dagnino_Gerardo_TSP.py
(TSP) PS C:\Users\gzdag\Documents\GitHub\AlgorithmAnalysisClass\Dagnino_Gerardo_TSP> python .\Dagnino_Gerardo_TSP.py
(TSP) PS C:\Users\gzdag\Documents\GitHub\AlgorithmAnalysisClass\Dagnino_Gerardo_TSP> python .\Dagnino_Gerardo_TSP.py
Ln 78, Col 64 Spaces: 4 UTF-8 CRLF Python TSP (3.13.0)
Search
ESP
09:09 p. m.
22/11/2025


File Edit Selection View Go Run Terminal Help
Dagnino_Gerardo_TSP
EXPLORER
DAGNINO_GERARDO_TSP
TSP
Scripts
activate
activate.bat
activate.fish
Activate.ps1
deactivate.bat
f2py.exe
fonttools.exe
meson.exe
ninja.exe
numpy-config.exe
pip.exe
pip3.13.exe
pip3.exe
pyftmerge.exe
pyftsubset.exe
python.exe
pythonw.exe
ttx.exe
wheel.exe
share
.gitignore
pyvenv.cfg
~$gnino_Gerardo_TSP.docx
Dagnino_Gerardo_TSP.docx
Dagnino_Gerardo_TSP.pdf
Dagnino_Gerardo_TSP.py
README.md
requierments.txt
OUTLINE
TIMELINE
Dagnino_Gerardo_TSP.py
Dagnino_Gerardo_TSP.py > ...
#Funciones para ejecucion y mostrar resultado
def dibujarGrafo(parent): #Dibujar y presentar el grafo dentro de un widget Tk usando matplotlib
G = nx.Graph()#Inicializar grafo de networkx
for nodo, vecinos in grafo.items():#Agregar nodos y aristas al grafo
for vecino, peso in vecinos.items():
G.add_edge(nodo, vecino, weight=peso)
pos = nx.spring_layout(G) #Posicionamiento de nodos
# Crear figura matplotlib y dibujar en ella
fig = Figure(figsize=(5, 4), dpi=100) #Crear figura
ax = fig.add_subplot(111) #Agregar subplot
ax.set_title("Grafo del Problema del Viajero (TSP)")
ax.axis('off')
# Dibujar grafo
nx.draw(G, pos, ax=ax, with_labels=True, node_color='lightblue', node_size=500, font_size=10)
labels = nx.get_edge_attributes(G, 'weight') #Etiquetas de pesos
nx.draw_networkx_edge_labels(G, pos, edge_labels=labels, ax=ax) #Dibujar etiquetas de pesos
fig.tight_layout() #Ajustar diseño
# Embedir la figura en Tk
canvas = FigureCanvasTkAgg(fig, master=parent)
canvas.draw()
widget = canvas.get_tk_widget()
widget.pack(fill='both', expand=True) #Empaquetar el widget
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
python
PS C:\Users\gzdag\Documents\GitHub\AlgorithmAnalysisClass\Dagnino_Gerardo_TSP> & C:\Users\gzdag\Documents\GitHub\AlgorithmAnalysisClass\Dagnino_Gerardo_TSP\TSP\Scripts\Activate.ps1
(TSP) PS C:\Users\gzdag\Documents\GitHub\AlgorithmAnalysisClass\Dagnino_Gerardo_TSP> python .\Dagnino_Gerardo_TSP.py
(TSP) PS C:\Users\gzdag\Documents\GitHub\AlgorithmAnalysisClass\Dagnino_Gerardo_TSP> python .\Dagnino_Gerardo_TSP.py
(TSP) PS C:\Users\gzdag\Documents\GitHub\AlgorithmAnalysisClass\Dagnino_Gerardo_TSP> python .\Dagnino_Gerardo_TSP.py
Ln 78, Col 64 Spaces: 4 UTF-8 CRLF Python TSP (3.13.0)
Search
ESP
09:09 p. m.
22/11/2025


File Edit Selection View Go Run Terminal Help
Dagnino_Gerardo_TSP
EXPLORER
DAGNINO_GERARDO_TSP
TSP
Scripts
activate
activate.bat
activate.fish
Activate.ps1
deactivate.bat
f2py.exe
fonttools.exe
meson.exe
ninja.exe
numpy-config.exe
pip.exe
pip3.13.exe
pip3.exe
pyftmerge.exe
pyftsubset.exe
python.exe
pythonw.exe
ttx.exe
wheel.exe
share
.gitignore
pyvenv.cfg
~$gnino_Gerardo_TSP.docx
Dagnino_Gerardo_TSP.docx
Dagnino_Gerardo_TSP.pdf
Dagnino_Gerardo_TSP.py
README.md
requierments.txt
OUTLINE
TIMELINE
Dagnino_Gerardo_TSP.py
Dagnino_Gerardo_TSP.py > ...
def dibujarGrafo(parent): #Dibujar y presentar el grafo dentro de un widget Tk usando matplotlib
ax.axis('off')
# Dibujar grafo
nx.draw(G, pos, ax=ax, with_labels=True, node_color='lightblue', node_size=500, font_size=10)
labels = nx.get_edge_attributes(G, 'weight') #Etiquetas de pesos
nx.draw_networkx_edge_labels(G, pos, edge_labels=labels, ax=ax) #Dibujar etiquetas de pesos
fig.tight_layout() #Ajustar diseño
# Embedir la figura en Tk
canvas = FigureCanvasTkAgg(fig, master=parent)
canvas.draw()
widget = canvas.get_tk_widget()
widget.pack(fill='both', expand=True) #Empaquetar el widget
def mostrar_resultado():
nodoInicial=nodoInicio.get()
tsp=TSPBacktracking(grafo)
ruta, costo=tsp.encontrar_mejor_ruta(nodoInicial)
if ruta is None:
resultado.config(text="Resultado: No se encontró ruta válida.")
else:
resultado.config(text=f"Resultado: Ruta: {' -> '.join(ruta)} | Costo: {costo}")
#Prueba inicial
if __name__ == "__main__":
#Interfaz
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
python
PS C:\Users\gzdag\Documents\GitHub\AlgorithmAnalysisClass\Dagnino_Gerardo_TSP> & C:\Users\gzdag\Documents\GitHub\AlgorithmAnalysisClass\Dagnino_Gerardo_TSP\TSP\Scripts\Activate.ps1
(TSP) PS C:\Users\gzdag\Documents\GitHub\AlgorithmAnalysisClass\Dagnino_Gerardo_TSP> python .\Dagnino_Gerardo_TSP.py
(TSP) PS C:\Users\gzdag\Documents\GitHub\AlgorithmAnalysisClass\Dagnino_Gerardo_TSP> python .\Dagnino_Gerardo_TSP.py
(TSP) PS C:\Users\gzdag\Documents\GitHub\AlgorithmAnalysisClass\Dagnino_Gerardo_TSP> python .\Dagnino_Gerardo_TSP.py
Ln 78, Col 64 Spaces: 4 UTF-8 CRLF Python TSP (3.13.0)
Search
ESP
09:09 p. m.
22/11/2025


File Edit Selection View Go Run Terminal Help
Dagnino_Gerardo_TSP
EXPLORER
DAGNINO_GERARDO_TSP
TSP
Scripts
activate
activate.bat
activate.fish
Activate.ps1
deactivate.bat
f2py.exe
fonttools.exe
meson.exe
ninja.exe
numpy-config.exe
pip.exe
pip3.13.exe
pip3.exe
pyftmerge.exe
pyftsubset.exe
python.exe
pythonw.exe
ttx.exe
wheel.exe
share
.gitignore
pyvenv.cfg
~$gnino_Gerardo_TSP.docx
Dagnino_Gerardo_TSP.docx
Dagnino_Gerardo_TSP.pdf
Dagnino_Gerardo_TSP.py
README.md
requierments.txt
OUTLINE
TIMELINE
Dagnino_Gerardo_TSP.py
Dagnino_Gerardo_TSP.py > ...
88
89 #Prueba inicial
90 if __name__ == "__main__":
91
92 #Interfaz
93 root=tk.Tk()
94 root.title("TSP Backtracking")
95 root.geometry("720x600")#Establecer tamaño de ventana
96 label=tk.Label(root, text="Resolviendo TSP con Backtracking", font=("Arial", 16))#Titulo de la ventana
97 label.pack(pady=20)
98 #Mostrar grafo
99 grafoFrame = tk.Frame(root, width=300, height=100) #Contenedor para el grafo
100 grafoFrame.pack(padx=10, pady=5, fill='both', expand=True) #Empaquetar el frame
101 dibujarGrafo(grafoFrame)#Dibujar el grafo en el frame
102 resultado=tk.Label(root, text="Resultado: ", font=("Arial", 12))
103 resultado.pack(pady=10)
104 nodoInicio=tk.Entry(root)
105 nodoInicio.pack(pady=5)
106 button=tk.Button(root, text="Calcular Mejor Ruta", command=lambda: mostrar_resultado())
107 button.pack(pady=10)
108 root.mainloop()
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
python
PS C:\Users\gzdag\Documents\GitHub\AlgorithmAnalysisClass\Dagnino_Gerardo_TSP> & C:\Users\gzdag\Documents\GitHub\AlgorithmAnalysisClass\Dagnino_Gerardo_TSP\TSP\Scripts\Activate.ps1
(TSP) PS C:\Users\gzdag\Documents\GitHub\AlgorithmAnalysisClass\Dagnino_Gerardo_TSP> python .\Dagnino_Gerardo_TSP.py
(TSP) PS C:\Users\gzdag\Documents\GitHub\AlgorithmAnalysisClass\Dagnino_Gerardo_TSP> python .\Dagnino_Gerardo_TSP.py
(TSP) PS C:\Users\gzdag\Documents\GitHub\AlgorithmAnalysisClass\Dagnino_Gerardo_TSP> python .\Dagnino_Gerardo_TSP.py
0 0
Ln 78, Col 64 Spaces: 4 UTF-8 CRLF Python TSP (3.13.0)
Search
ESP
09:09 p. m.
22/11/2025