UNIVERSIDAD DE GUADALAJARA

CENTRO UNIVERSITARIO DE CIENCIAS EXACTAS E INGENIERÍAS

Seminario Para la Solución de Problemas de Algoritmia

Reporte de práctica

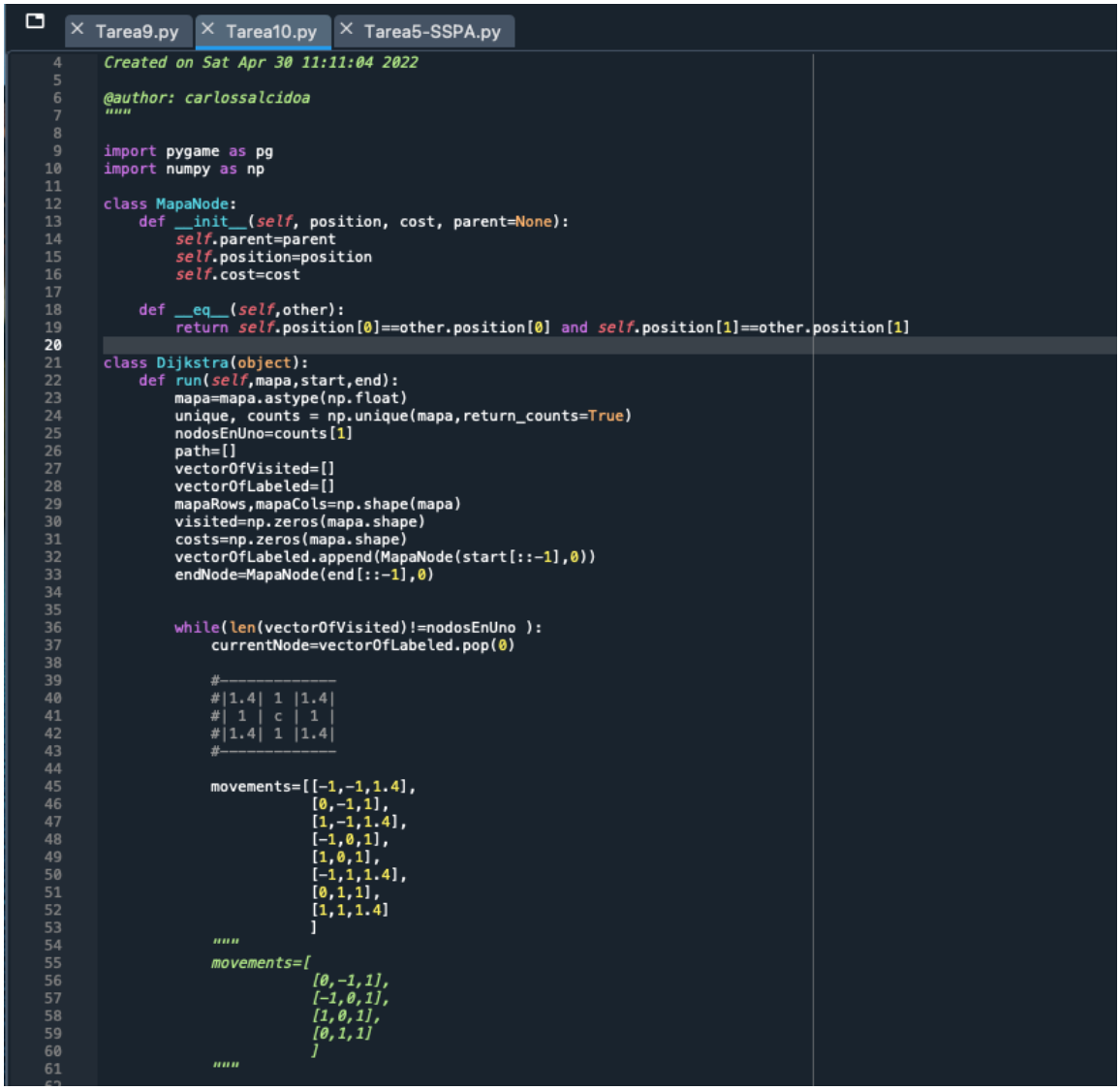
Nombre del alumno:	Carlos Uriel Salcido Aviña
Profesor:	Erasmo Gabriel Martínez Soltero
Título de la práctica:	"Tarea 10. Algoritmo de Dijkstra"
Fecha:	2 de mayo 2022

Introducción

El objetivo es trazar lineas sobre el archivo de la tarea 5, haciendo uso de nuevo el pygame, en esta ocasión el camino se trazará implementando el algoritmo de Dijkstra

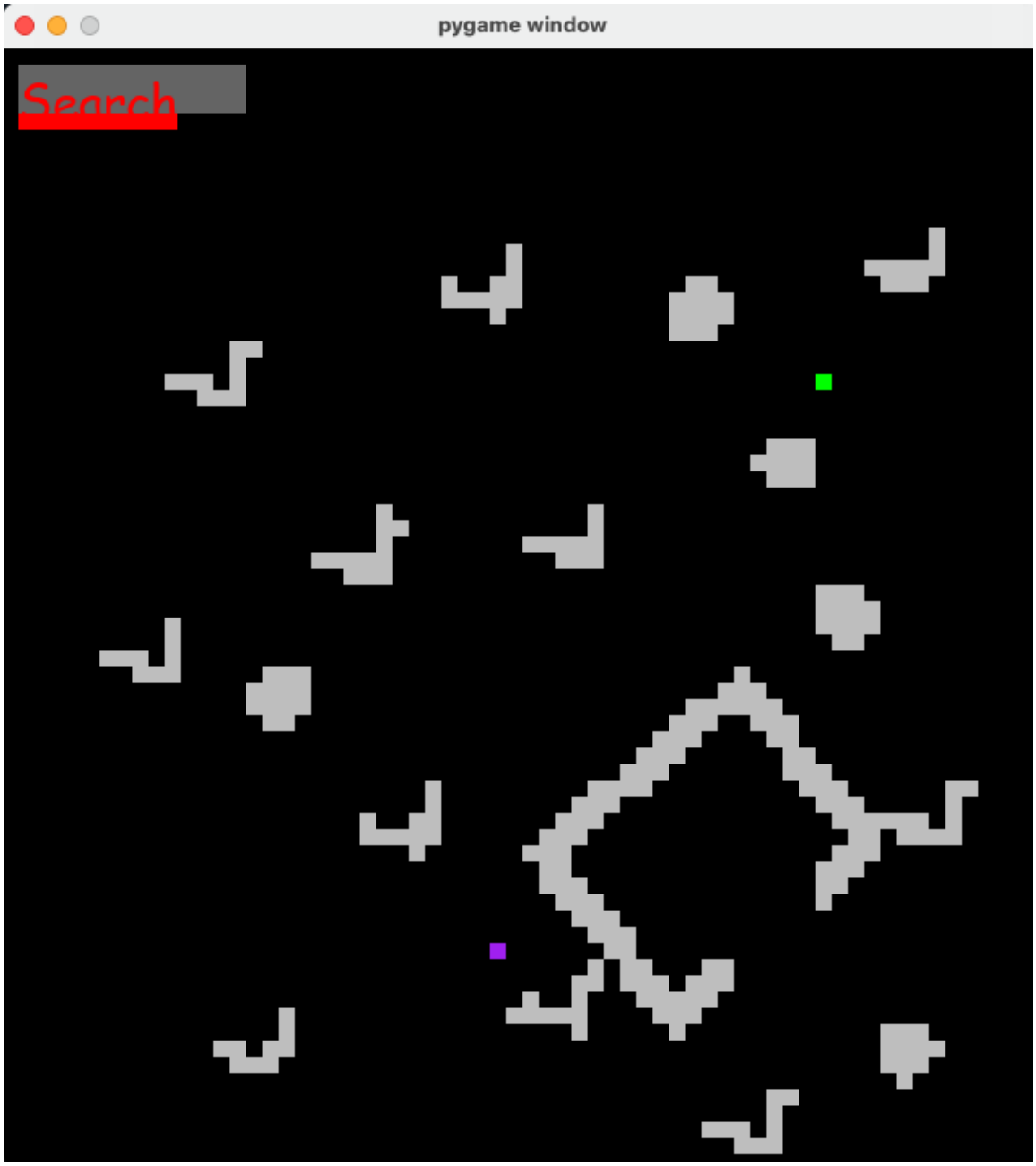
Metodología

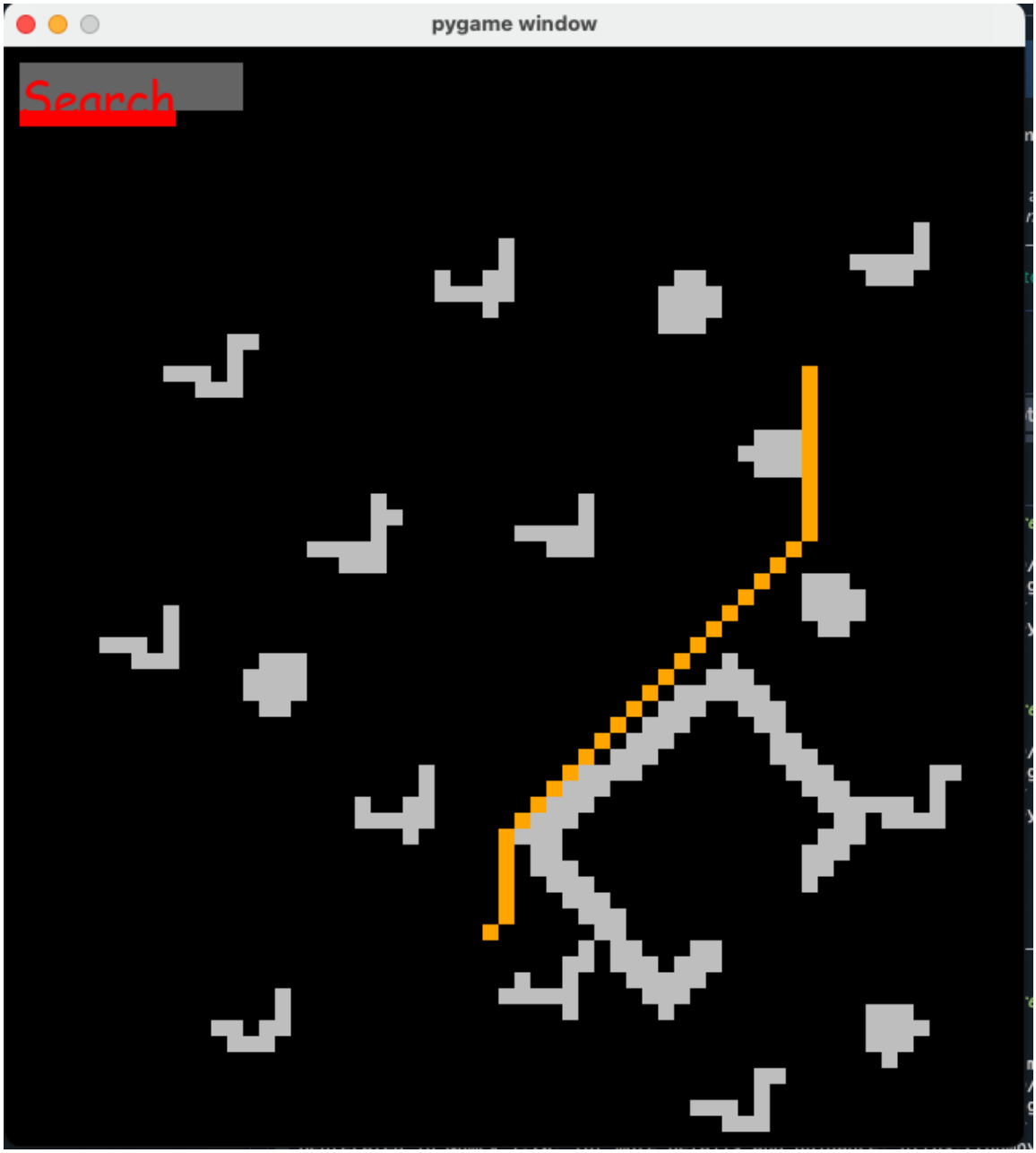
Para cumplir el objetivo de la práctica, tomé el mismo código que utilicé en la tarea 5, luego tocó modificar la función de longitud por una que utilice el algoritmo Dijktra. Desarrollé el siguiente código:


Tarea9.py
Tarea10.py
Tarea5-SSPA.py
Created on Sat Apr 30 11:11:04 2022
@author: carlossalcidoa
import pygame as pg
import numpy as np
class MapaNode:
def __init__(self, position, cost, parent=None):
self.parent=parent
self.position=position
self.cost=cost
def __eq__(self,other):
return self.position[0]==other.position[0] and self.position[1]==other.position[1]
class Dijkstra(object):
def run(self,mapa,start,end):
mapa=mapa.astype(np.float)
unique, counts = np.unique(mapa,return_counts=True)
nodosEnUno=counts[1]
path=[]
vectorOfVisited=[]
vectorOfLabeled=[]
mapaRows,mapaCols=np.shape(mapa)
visited=np.zeros(mapa.shape)
costs=np.zeros(mapa.shape)
vectorOfLabeled.append(MapaNode(start[::-1],0))
endNode=MapaNode(end[::-1],0)
while(len(vectorOfVisited)!=nodosEnUno):
currentNode=vectorOfLabeled.pop(0)
#------------
#|1.4| 1 |1.4|
#| 1 | c | 1 |
#|1.4| 1 |1.4|
#------------
movements=[[-1,-1,1.4],
[0,-1,1],
[1,-1,1.4],
[-1,0,1],
[1,0,1],
[-1,1,1.4],
[0,1,1],
[1,1,1.4]
]
"""
movements=[
[0,-1,1],
[-1,0,1],
[1,0,1],
[0,1,1]
]
"""

```
62
63              for movement in movements:
64  #               creamos la posicion del adjacente
65                  newPosition=[currentNode.position[0]+movement[0],currentNode.position[1]+movement[1]]
66                  adjacentNode=MapaNode(newPosition, currentNode.cost+movement[2],currentNode)
67
68  #                REVisamos que este dentro del mapa, que no haya sido visitado y que no sea obstaculo
69                  if newPosition[0]<0 or newPosition[1]<0 or newPosition[1]>=mapaCols or newPosition[0]>=mapaRows:
70                      continue
71                  elif mapa[newPosition[0]][newPosition[1]]==0:
72                      continue
73                  elif visited[newPosition[0]][newPosition[1]]==1:
74                      continue
75                  else:
76                      #se busca la lista de etiquetados
77                      encontrado=False
78                      for labeled in vectorOfLabeled:
79                          if (labeled==adjacentNode):
80                              encontrado=True
81                              if (labeled.cost>adjacentNode.cost):
82                                  labeled.cost=adjacentNode.cost
83                                  costs[newPosition[0]][newPosition[1]]=adjacentNode.cost
84                                  labeled.parent=currentNode
85
86                      if not encontrado:
87                          #agregamos el nodo a los etiquetados
88                          vectorOfLabeled.append(adjacentNode)
89                          costs[newPosition[0]][newPosition[1]]=adjacentNode.cost
90
91              vectorOfVisited.append(currentNode)
92              if(currentNode==endNode):
93                  break
94
95              #marcamos el nodo como visitado
96              visited[currentNode.position[0]][currentNode.position[1]]=1
97              vectorOfLabeled=sorted(vectorOfLabeled, key=lambda x: x.cost)
98
99          #de reversa mami para obtener el camino
100         #buscar mi destino dentro de los nodos visitados
101         for visitedNode in vectorOfVisited:
102             if visitedNode==endNode:
103                 endNode=visitedNode
104                 break
105
106         #una vez encontrado el que se marcó como final recorrer el grafo hacia atrás siguiendo los padres
107         while endNode is not None:
108             path.append(endNode.position)
109             endNode=endNode.parent
110         return path, visited, costs
111
112
```

Resultados





Conclusiones

No tuve complicaciones al momento de realizar la práctica

Referencias

Gabriel Martínez(2021)"Dijkstra aplicado a un mapa con obstaculos usando python con el paquete de pygames", en YouTube. Disponible en: https://www.youtube.com/watch?v=rS-fRggCQQo