**GRAPH – Algorithms**

**Breadth first search(v):**

q = Queue(maxsize = self.GetNrVertices())

prev = {}

dist = {}

visited = []

q.put(s)

visited.append(s)

dist[s] = 0

while not q.isEmpty() do

x = q.get()

for y in self.Dout[x] do

if y not in visited then

q.put(y)

visited.append(y)

dist[y] = dist[x] + 1

prev[y] = x

return visited (is a list with all accessible vertices from vertex v)

**Depth first search(v, visited):**

if visited is None:

visited = []

prev = {}

dist = {}

visited.append(v)

for i in self.Dout[v]:

if i not in visited:

dist[y] = dist[x] + 1

prev[y] = x

self.DFT(I, visited)

return visited

**Dijkstra algorithm(start, final):**

if visited is None:

dist = {}

prev = {start:None}

q = []

dist[start] = 0

heapq.heappush(q, (dist[start], start))

while len(q) > 0:

var, x = heapq.heappop(q)

if (var == dist[x]):

for y in self.Dout[x]:

if y not in prev or dist[y] > dist[x] + self.DCost[(x, y)]:

prev[y] = x

dist[y] = dist[x] + self.DCost[(x, y)]

heapq.heappush(q, (dist[y], y))

if self.GetPath(prev, start) is None:

return None

else:

return self.GetPath(prev, start), dist[strat]