Annexes TP graphes (2/2)

1 Algorithme de Dijkstra dans un réseau carré

```
def DIJKSTRA_carre(n,dep,fin):
     G=dico_voisins(n)
     file=filePrio()
     file=put(dep,0,file)
     distances_origines={dep:0}
     for v in G:
          if v!=dep:
               distances_origines[v]=float("inf")
     parents={dep:None}
     chemin=[]
     plt.pause(0.1)
     carre(n)
     plt.plot(dep[0],dep[1],'bo')
     plt.plot(fin[0],fin[1],'bo')
     while not empty(file):
          s,d=get(file)
          if s!=fin:
               for v,delta in G[s]:
                    distance=distances_origines[s]+delta
                    if distance<distances_origines[v]:
                         distances_origines[v]=distance
                         file=put(v,distance,file)
                         parents[v]=s
                         plt.plot(v[0],v[1],"ro")
                         plt.pause(0.1)
          else:
               plt.plot(s[0],s[1],'bo')
               chemin=[fin]
               while s!=dep:
                    s=parents[s]
                    chemin=[s]+chemin
                    plt.plot(s[0],s[1],'bo')
                    plt.pause(0.1)
     return distances_origines[fin],chemin
```

2 Algorithme A* dans un réseau carré

```
def algoA(n,dep,fin):
    G=dico_voisins(n)
    distances_origines={}
    distances_origine={dep:0}
    for v in G:
        if v!=dep:
            distances_origine[v]=float("inf")
    h={} # heuristique
    for v in G:
        h[v]=euclide(v,fin) # distance de v à l'arrivée
    score={}
    score[dep]=h[dep]+distances_origine[dep] # distance origine+fin
    file=filePrio()
    file=put(dep,score[dep],file) # priorité au score le plus faible
    parents={dep:None}
    plt.pause(0.1)
    carre(n)
    plt.plot(dep[0],dep[1],'yo')
    plt.plot(fin[0],fin[1],'yo')
    while not empty(file):
        s,d=get(file) # sommet et distance au parent
        if s!=fin:
            for v,delta in G[s]:
                #on cherche le sommet le plus proche de l'origine...
                #...et le moins loin de l'arrivée
                distance=distances_origine[s]+delta
                if  distance<distances_origine[v]:
                    distances_origine[v]=distance
                    score[v]=distance+h[v]
                    file=put(v,score[v],file)
                    parents[v]=s
                    plt.plot(v[0],v[1],"ro")
                    plt.pause(0.1)
        else:
            plt.plot(s[0],s[1],'yo')
            chemin=[fin]
            while s!=dep:
                s=parents[s]
                chemin=[s]+chemin
                plt.plot(s[0],s[1],'yo')
                plt.pause(0.1)
            return distances_origine[fin],chemin
```