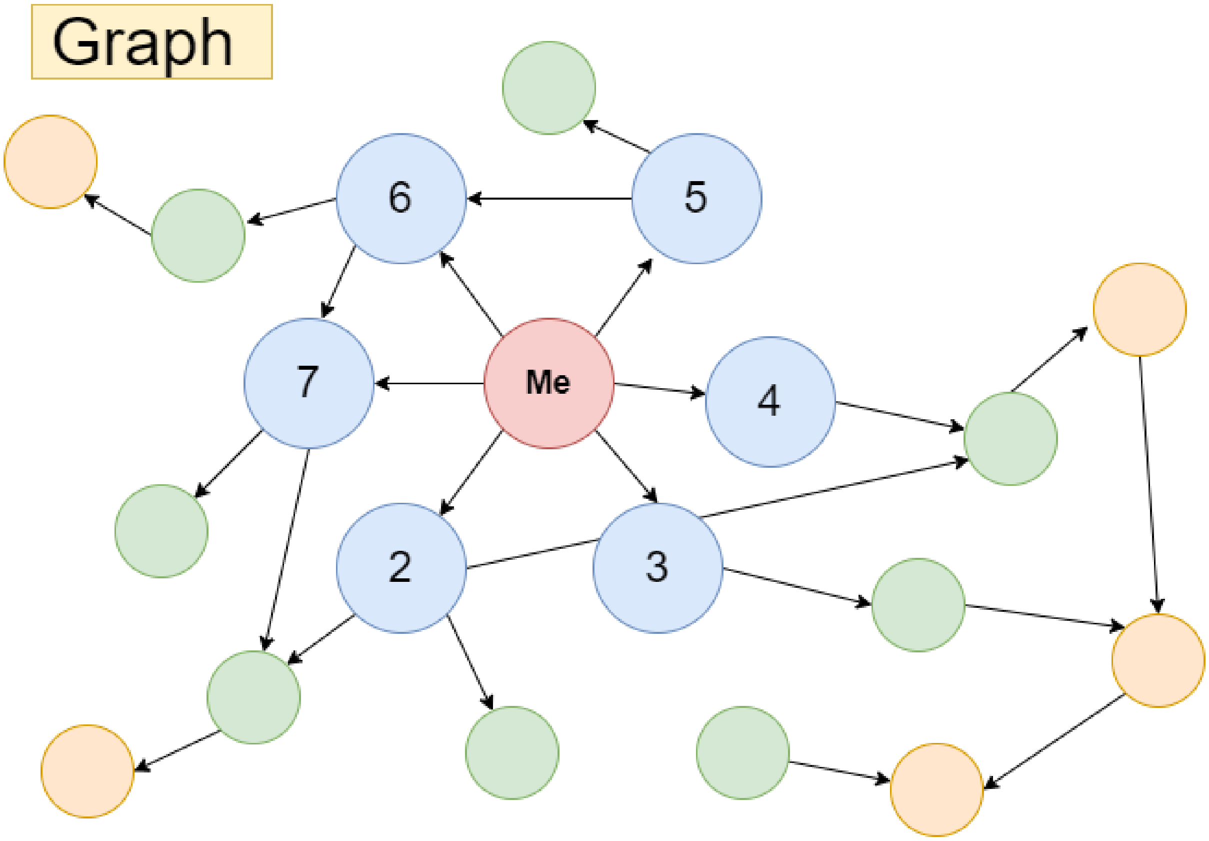

Example of graph

The overall complexity : Primary (O(E+Vlog V)

Bouns (O(V+E))

Analysis:

Code	*Time Complexity*
for i in dff["follwer"]: * follwers.add(i)* *for i in dff["influncer"]:* * influncers.add(i)*	*o(E) for both loops "add() function is O(1) because Python's set data structure is implemented as a hash table ”*
dff.drop_duplicates(inplace=True)	*o(E) which o small that we can’t reach to it*

d=dict()
for f,i in zip(dff["follwer"],dff["influncer"]):
    if i in d:
        d[i].follwers.append(f)
        d[i].nof+=1
    else:
 	
        d[i]=node(1,[f],-math.inf)
	O(E) : All this
	
	check = o(1) as d is a dictionary not a list

for key in d:
    if d[key].nof>maxi:
        famous=key
        maxi=d[key].nof
	O(v)

sd=sorted(d.items(),key=lambda pair:-pair[1].nof)	O(v log v) It's sort by comparsion

for k in d:
    d[k].state=-math.inf
 	
for n in q:
    d[n].state=-1
 	
for follwer in q:
    sus[follwer]=-1
	Each For will be :O(V)

for n in q:
     for gc in d[n].follwers:
       if gc in sus:
 	
           if sus[gc]!=-1:
               sus[gc]+=1
       else:
           sus[gc]=1
	It's all :O(V+E)
	
	It's directed graph so ,O(v) + O(E)
	=O(V+E)

for k in sus:
    if sus[k]>=metric:
 	
follow_those_people.append((k,sus[k]))
)
	O(V)