**SOFT COMPUTING**

**END TREM PRACTICAL 2020**

**AIM**: Write a program to implement Depth Limited Search Algorithm in Python programming Language.

**CODE:**

ADJ = {}

ADJ['S'] = ['2', '6']

ADJ['2'] = ['S', '3']

ADJ['3'] = ['2','8']

ADJ['G'] = ['10']

ADJ['6'] = ['S', '11']

ADJ['8'] = ['3', '13']

ADJ['10'] = ['G', '15']

ADJ['11'] = ['6', '12']

ADJ['12'] = ['11', '13', '17']

ADJ['13'] = ['8', '12']

ADJ['15'] = ['10', '20']

ADJ['17'] = ['12','22']

ADJ['19'] = ['20', '24']

ADJ['20'] = ['15','19']

ADJ['21'] = ['22']

ADJ['22'] = ['17','21','23']

ADJ['23'] = ['22', '24']

ADJ['24'] = ['19','23']

print ("adj",ADJ)

# keep track of visited nodes

visited = {str(i) : False for i in range(1,26)}

visited['S'] = False

visited['G'] = False

def dls(start, goal,limit):

depth = 0

OPEN=[]

CLOSED=[]

OPEN.append(start)

visited["S"] = True

while OPEN != []: # Step 2

if depth<=limit:

current = OPEN.pop()

if current == goal:

print("Goal Node Found")

return True

else:

lst = successors(current)

for i in lst:

# try to visit a node in future, if not already been to it

if(not(visited[i])):

OPEN.append(i)

visited[i] = True

depth +=1

else:

print("Not found within depth limit")

return False

print(OPEN)

#print("node visited",i,sep='>',end='\n')

return False

def successors(city):

return ADJ[city]

def test():

start = 'S'

goal = 'G'

limit=int(input("ENTER THE DEPTH LIMIT"))

print("Starting a dls from \n[ " + start+" ]")

print(dls(start, goal,limit))

if \_\_name\_\_ == "\_\_main\_\_":

test()

**OUTPUT SNAPSHOTS:**

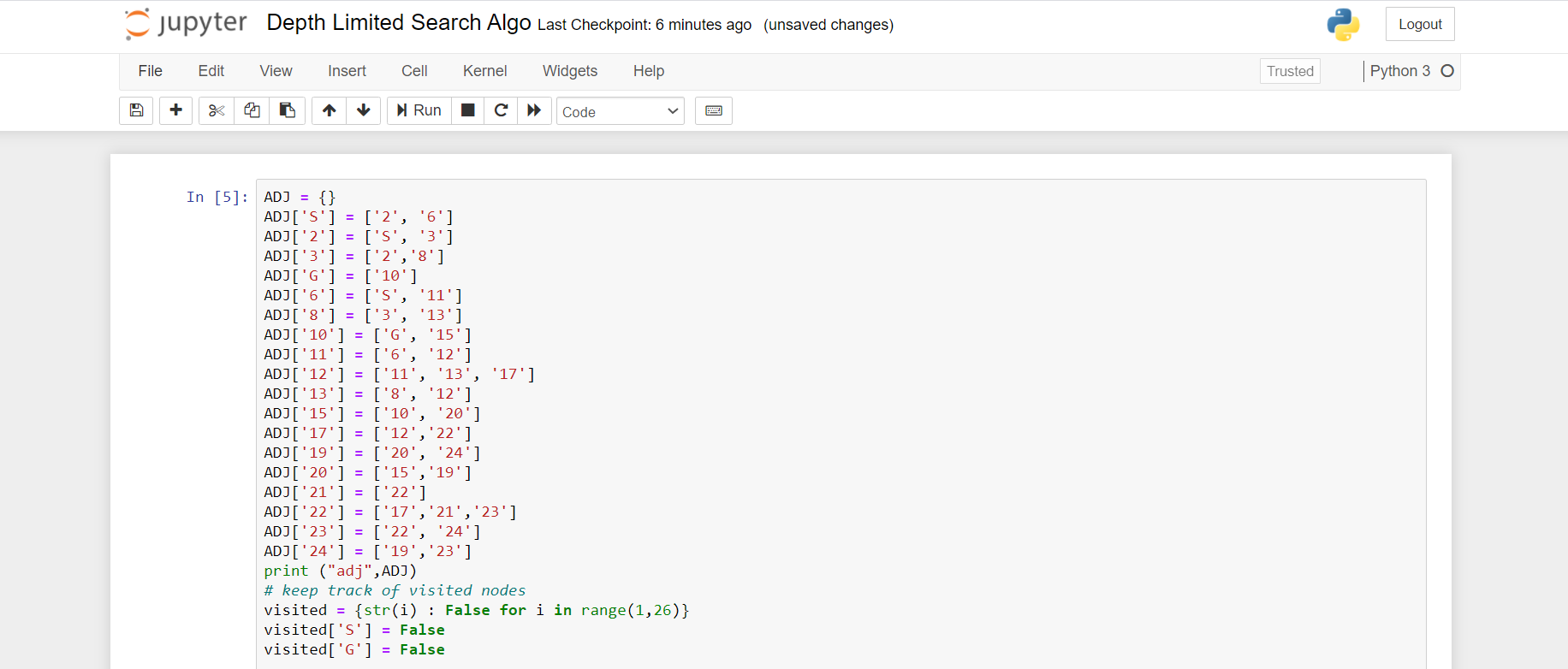


Figure : Code

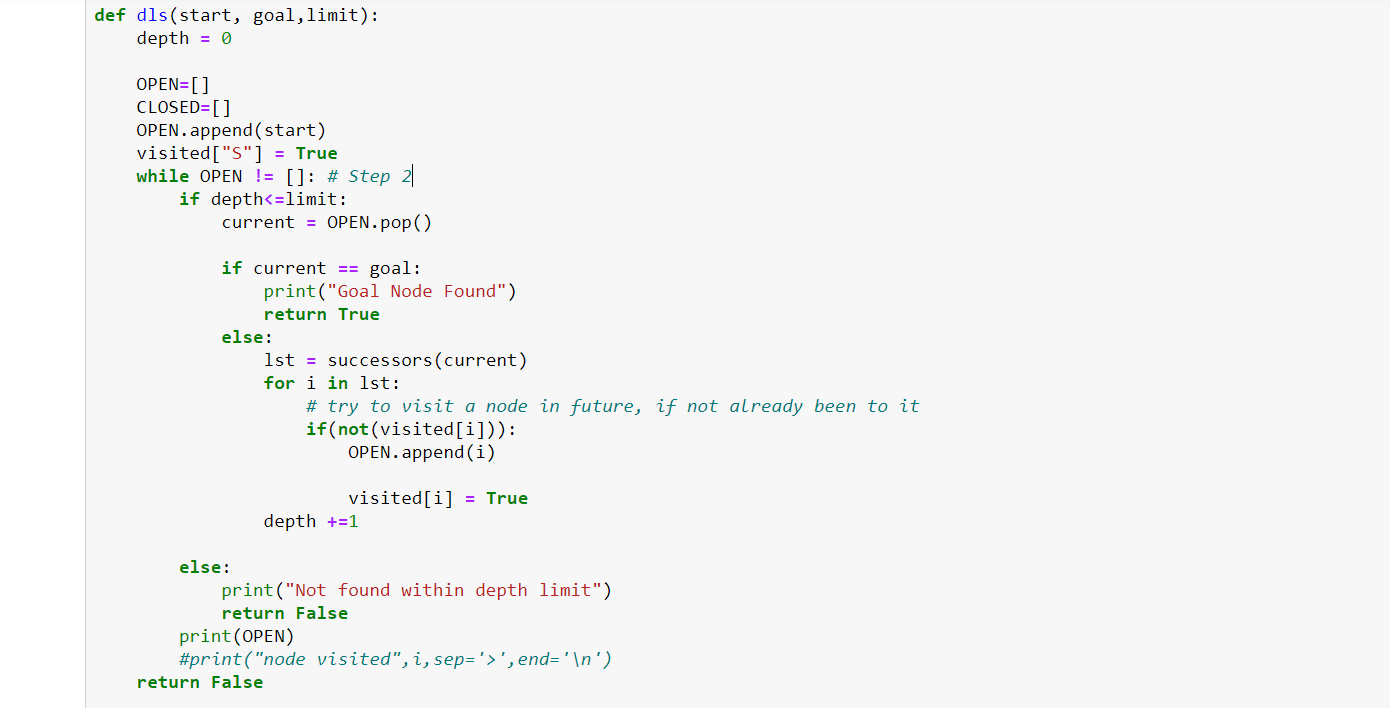


Figure : code

****

Figure : Code

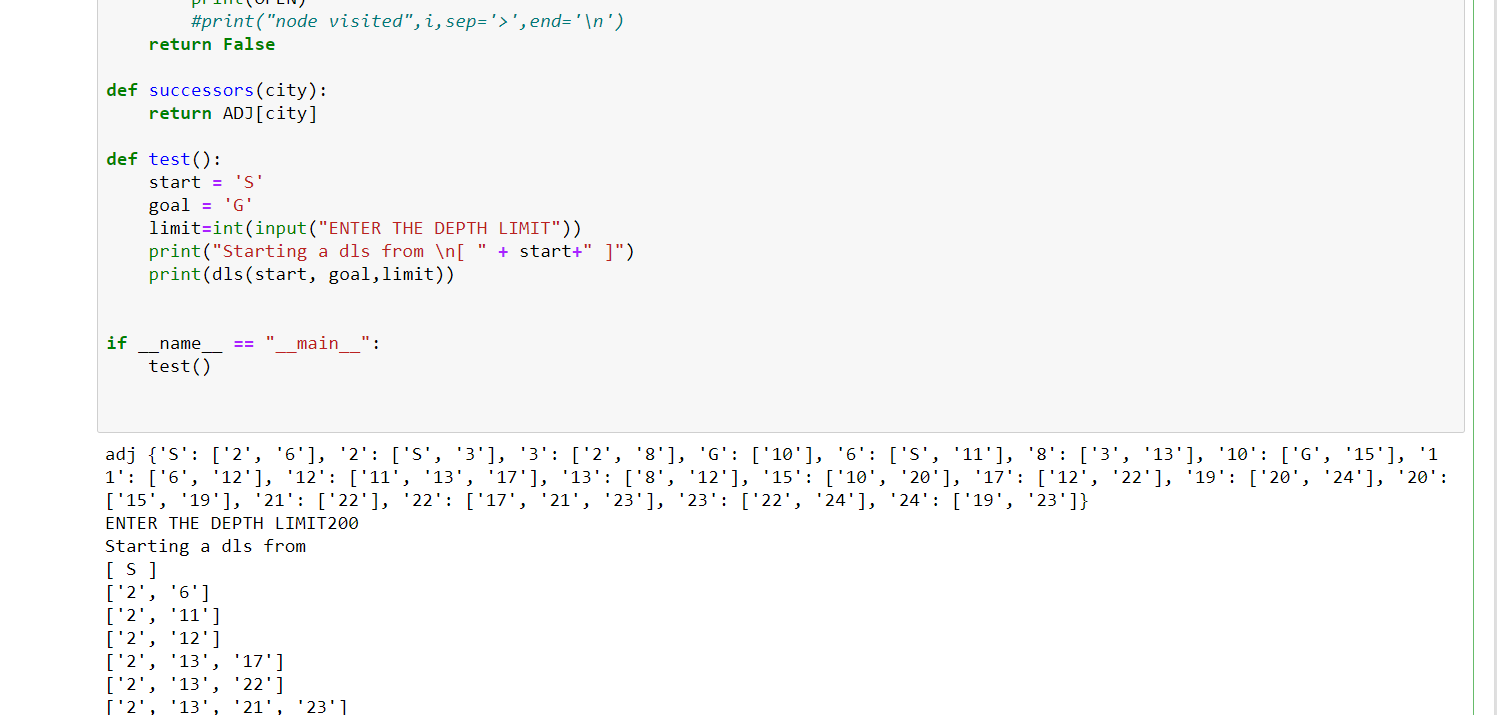
****

Figure : output

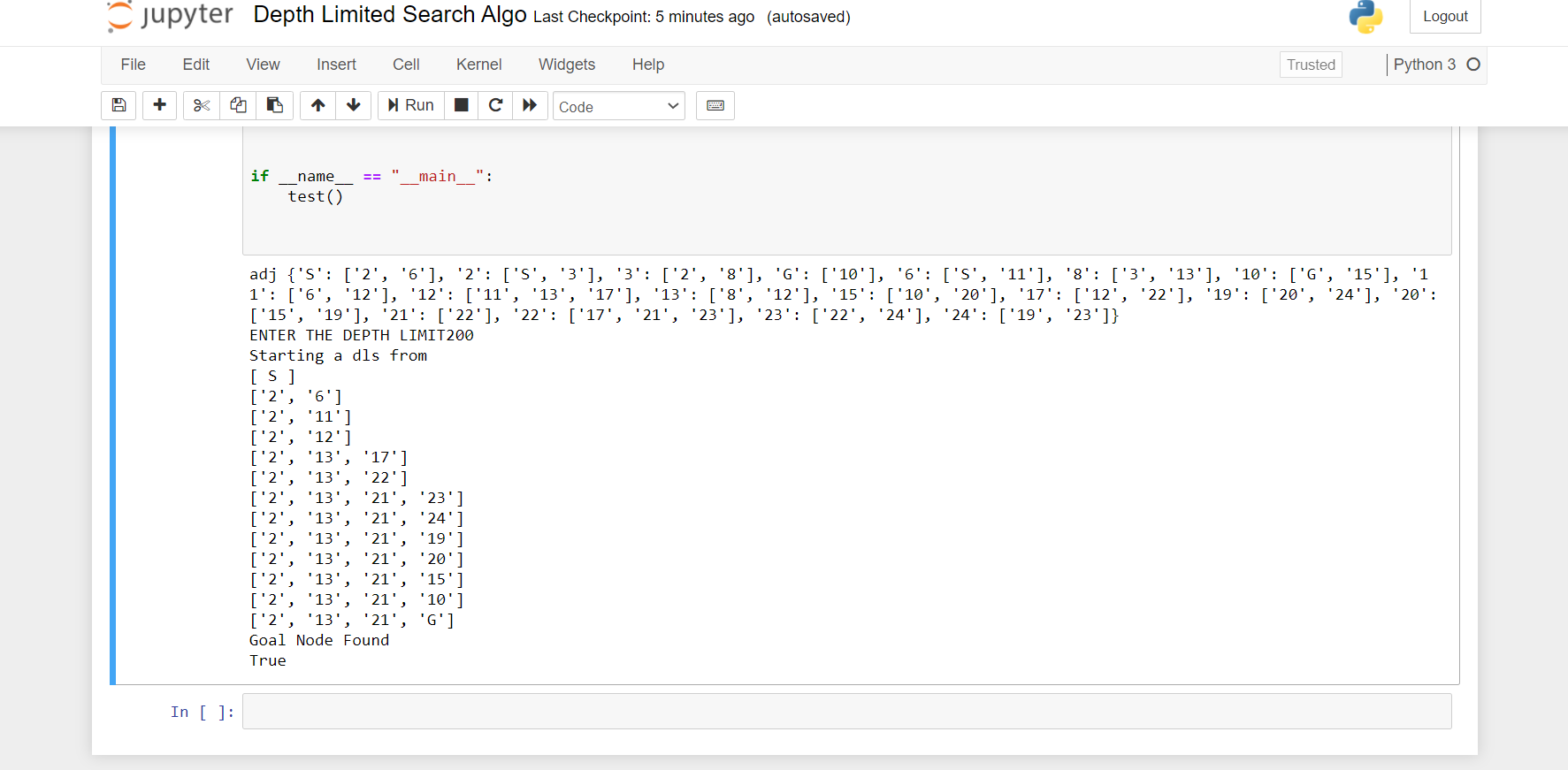
****

Figure : output