N- Queens Exp No : 1 1. N- Queens Problem Using Backtracking in def 1852fe (board, row, wor, n): python. Aim: - Write a program to solve N-Queens problem with python code. olde Frent Jalle Algorithm: for (1-, 1-, wor) = pmor) gis ai i i vob i) Start ii) Placing empty space - Create an NXN board filled. iii) call check it placing Queen is safe. No Queen, is in the Same tow. No Other is in the Same diagonal. No other queen is in the same lower diagonal. iv) If a safe position is found, place the queen ('i') and move to next column. using reculsion. v) If all Queen are Successfully placed the algorithm, return True, offully placed the Blution is bound. Indicating that the vi) If placing the Queen is the auxent Column does not lead to solution Agonthm Backtlacks to the previous Column and tries mext row. def solve No (n):

[ (n) sprior il- vod n\* [0]] = brood : also 1904 ( board, 0, 0) = delice: print ("Solution ales not enerts") Inny

schoon take

```
Program: int & see your meldod snow
 def issafe (board, row, w/n):
   for i in range (col):
     Jeturn False
 dor ij in Zip (range (row, -1, -1) range
  ochen False prisold ti does de
 for is in zip (range (now, n, -1) range
   lif board Cij [j] == 1.
    detran True
  def solve NO Util (board, col, n):
  if color=n: even bono (1,1) no
      Jeturn Tous noisluse pridu
   for i fin range (n).
 bon-1 non Col-h):
     board [i] [co 1]=1
   if she No util (board, col +1, n) == True
    return True
                 Horthm Back Hack
     board Ci J [w] J=000 board on
  Jetum False
  def solve No (n):
     board = [[o]* n for -in range (n)]
     it blue NOUTI (board, 0, 1) == delse:
      print ("Solution does not exists")
    Xturn False
```

for i in board: print (i] ochum Tome dept titale tomoralami of n = int (input ("enter n value: ")
dolne NO(n) is postible before bocktracking Output: :matroplAv Enter n value: 5 [0,0,0,0] tilo loso, 1,000 of milet and of the Land of to feet troude of vi [0,1,0,0,0] ii) Rush the starting node [0,0,0,0,1] to [0,0] the took of 10,0,0] Result: dasto at most about jet at das (v vii) For lack adjacent unvinted neighbour with Mark the new glabour as visited xi) Push the unwishted reighteur unt the Stack. en show that all sectable to spel (ix des (xi

Depth First Scarch

Exp No:2

Exp No:2 Depth First Search

Aim:

To implement depth first search (PF8) to travelse a graph & explore all Vertices by visiting as far alongs Branch as possible before backtracking.

## Algorithm:

- i) Start
- ii) Initialize an empty stack and a list to keep track of visited nodes,

Enter in value is

- ii) Rush the Starting node sonto stack is mas
- iv) while the stack is not empty, repeat
- v) Pop the top node from the stack.
- Vi) Print or process the popped mode
- vii) For lach adjacemt unvisited neighbi
- viii) Mark the neighbour as visited vi) Push the unvisited neighbour onto the Stack.
- xii) Répeat until all reachable node ale visited.
- ix) Sop.

```
Program:
 def dbs (graph, Stark):
                                 · mit
   Stack = [Start]
   Visited = Ret () melding por setous
while stack:
   node = Stack by
   if hode not in visited:
   Print (node end = ")
    Visited.add (node)
  for neighbor in graph [hode]:
     if neighbor not in visited:
     Stack. append (neighbou)
 graph = d
 A' & [B' C']

B' C']

The standard of enemy standard (V
 parc, [ E. E. Parlon 10
                   I the annount
   D' C To descript a file
ETETO ( Somet 21 notale)
. If the state x, y has been grisity to fore
    Skip to the next iteration.
               340 State At ALVOM.
Print (" DFS Traversal Starting from rode'A': )
Output: DF8 Travellal Start brown ade X1
         ABE ACFBE'D
                 ( o): Fpit Hamet
                  Jomph Juga: LX o
```

Novell divit Atgal