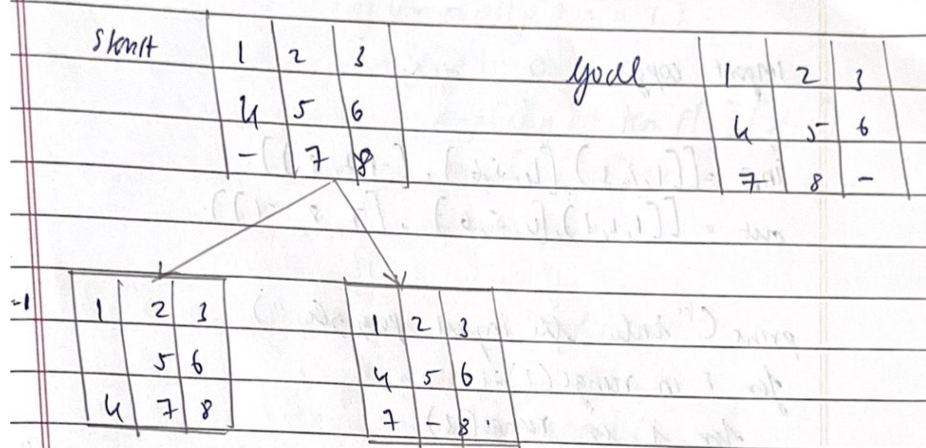
23/11/22. 8 PUZZLE - IDS

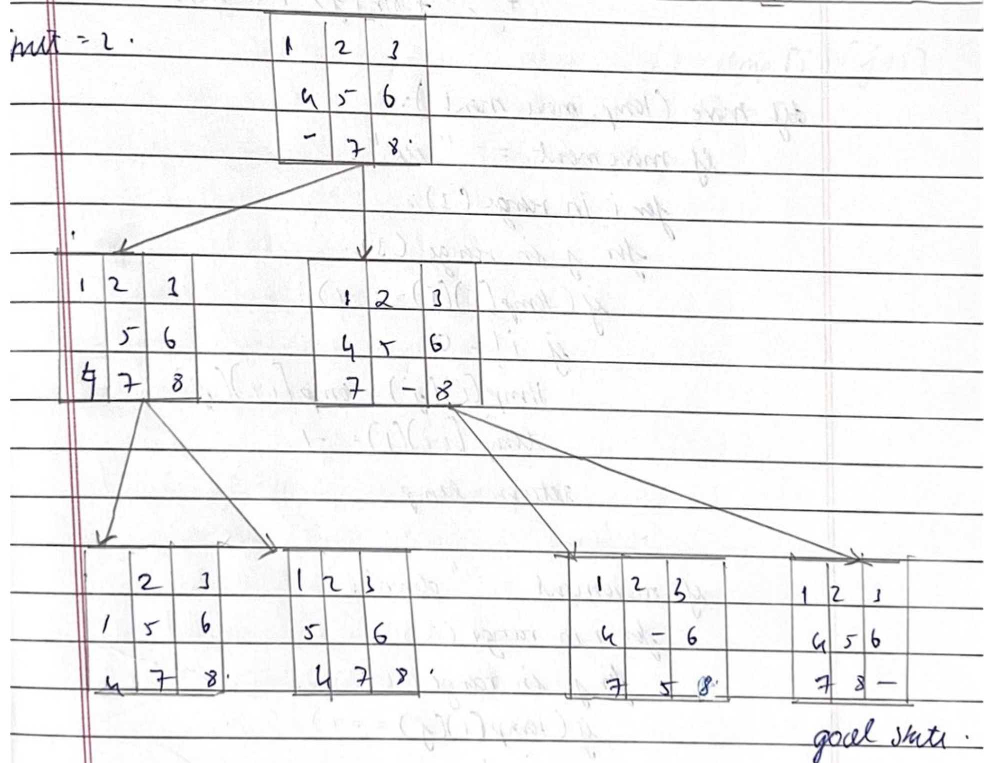
Aim - To implement 8 puzzle game using IDS algorithm.

Algorithm:

```
IDS(root, goal, depth limit)
    for depth = 0 to depth_limit
        if (DFS(root, goal, depth))
            return True
return false
```

State space tree





code:

```
import copy

inp = [[1,2,3],[4,5,6], [-1,7,8]].
out = [[1,2,3],[4,5,6] , [7, 8, -1]].

print ("Enter the input puzzle ")
for i in range(3):
    for j in range(3):
        inp[i][j] = int (input ("Enter number at "+str(i)
                    +" , "+str(j) +" ->"))


def move (temp, move ment ):
    if movement == "up":
        for i in range (3):
            for j in range (3):
                if (temp[i][j] == -1):
                    if i != 0
                        temp(i)(j) = temp [i-1][j]
                        temp [i-1][j] = -1
                return temp

    if movement == "down":
        for i in range (3) :
            for j in range (3):
                if (temp[i][j] == -1)
                    if i != 2 :
                        temp[i][j] = temp[i+1][j]
                        temp[i+1][j] = -1
                return temp

    if movement == "left":
        for i in range(3):
            for j in range(3):
                if (temp[i][j] == -1):
                    if j != 0:
                        temp[i][j] = temp[i][j-1]
                        temp[i][j-1] = -1
                return temp.

        if movement == "right":
            for i in range(3):
                for j in range(3):
                    if (temp[i][j] == -1):
                        if j != 2
                            temp[i][j] = temp[i][j+1]
                            temp[i][j+1] = -1
                    return temp.

def ids():
    global inp
    global out
    global flag

    for limit in range(100):
        print('LIMIT -> '+ str(limit))
        stack=[]
        inpx = [inp, "none"]
        stack.append(inpx)
        level = 0
        while(True):
            if len(stack) == 0:
                break
            puzzle = stack.pop(0)
            if level <= limit:
                print(str(puzzle[1]) + " --> " + str(puzzle[0]))
                if(puzzle[0] == out):
```

```
        print ("Found")
        print ('PATH COST  =' +str (level))
        flag = True
        return
    else:
        level = level +1
        if (puzzle[1]!= "down"):
            temp = copy.deepcopy(puzzle[0])
            up = move (temp, "up")
            if (up != puzzle[0]):
                upx = [up, "up"]
                stack.insert (0, upx)

        if (puzzle[1]!= "right"):
            temp = copy.deepcopy(puzzle[0])
            left = move(temp, "left")
            if (left != puzzle[0]):
                leftx = [left, "left")
                stack.insert (0, leftx)

        if (puzzle[1] != "up"):
            temp = copy.deepcopy(puzzle[0])
            down = move(temp, "down")
            if ( down != puzzle[0]):
                downx = [down, "down")
                stack.insert (0, downx)

        if (puzzle[1]!= "left"):
            temp = copy.deepcopy(puzzle[0])
            right = move(temp, "right")
            if (right != puzzle[0]) :
                rightx = [right, "right")
                stack.insert (0, right)
```

classmate
Date
Page

ids() .

O/P: Enter input puzzle.

```
Enter number at   0,0 → 1
Enter number at   0,1 → 2
Enter number at   0,2 → 3
Enter number at   1,0 → 4
Enter number at   1,1 → 5
Enter number at   1,2 → 6
Enter number at   2,0 → -1
Enter number at   2,1 → 7
Enter number at   2,2 → 8

        - IDS -
Limit -> 0
none -> [[1,2,3], [4,5,6], [-1,7,8]].
Limit -> 1
none -> [[1,2,3], [4,5,6], [-1,7,8]].
right -> [[1,2,3], [4,5,6], [7,-1,8]].
limit -> 2
none -> [[1,2,3], [4,5,6], [-1,7,8]]
right -> [[1,2,3], [4,5,6], [7,-1,8]]
right -> [[1,2,3], [4,5,6], [7,8,-1]]
found
path cost = 2 .
```