Algorithms & Data Structures

Time Complexity Analysis of
the Sliding Tile Puzzle by Al
Sweigart

import random, sys: basic operation: 1= θ(1)

BLANK = " ": basic operation: 1= θ(1)

def main()...: 1* [1+1+1+n(1+1+1+1(1+1)]= 3 + 5n = θ(n)

def getNewBoard()...: 1 [1]= 1 basic operation = θ(1)

def displayBoard(board)...: 1 [1+1+1] = 3 basic operations = θ(1)

def findBlankSpace(board)...: 1 [n*n*1*1]=n^2 = θ(n^2)

def askForPlayerMove(board)...: 1 [2+3+3+3+3 + n(1+1+1+1(1)+1(1))]= 14 + 5n = θ(n)

def makeMove(board, move)...: 1 [2+1(2) + 1(2) + 1(2) + 1(2)] = 10 basic operations = θ (1)

def makeRandomMove(board)...: 1[2 + 1 + 1(1) + 1(1) + 1(1)+ 1(1) + 1] = 8 basic operations = θ(1)

def getNewPuzzle(moves=200)...: 1[n(1) + 1] = n+1 = θ(n)

if __name__ == '__main__': 1 basic operation = θ(1)
    main() : 1 basic operation = θ(1)

Tn(n)= 1+ 1+ n + 1 + n^2 + 1 + n + 1 + 1 + n +1 + 1 + 1 + 1
T(n)= 1 + n + n + n + n^2

θ(n^2)

The program's Time Complexity is Quadratic= θ(n^2).

Group 1

June 5, 2022