**Task 1**

def covid(file):

mat = []

ys = []

dfs\_counts = []

with open(file, 'r') as f:

for line in f.readlines():

t = line.split(' ')

t[-1] = t[-1].replace('\n', '')

t[-1] = t[-1].replace('\t\t', '')

mat.append(t)

for row in range(len(mat)):

for column in range(len(mat[row])):

if mat[row][column] == 'Y':

ys.append((row, column))

# dfs starting from all pairs and keeping the count in dfs\_count

for p in ys:

stack = [p]

visited = [p]

ys.remove(p)

while len(stack) > 0:

current = stack.pop(-1)

# checking up

if (current[0]-1, current[1]) in ys and (current[0]-1, current[1]) not in visited:

stack.append((current[0]-1, current[1]))

visited.append((current[0]-1, current[1]))

ys.remove((current[0]-1, current[1]))

# checking down

if (current[0]+1, current[1]) in ys and (current[0]+1, current[1]) not in visited:

stack.append((current[0]+1, current[1]))

visited.append((current[0]+1, current[1]))

ys.remove((current[0]+1, current[1]))

# checking left

if (current[0], current[1]-1) in ys and (current[0], current[1]-1) not in visited:

stack.append((current[0], current[1]-1))

visited.append((current[0], current[1]-1))

ys.remove((current[0], current[1]-1))

# checking right

if (current[0], current[1]+1) in ys and (current[0], current[1]+1) not in visited:

stack.append((current[0], current[1]+1))

visited.append((current[0], current[1]+1))

ys.remove((current[0], current[1]+1))

# checking bottom right

if (current[0]+1, current[1]+1) in ys and (current[0]+1, current[1]+1) not in visited:

stack.append((current[0]+1, current[1]+1))

visited.append((current[0]+1, current[1]+1))

ys.remove((current[0]+1, current[1]+1))

# checking bottom left

if (current[0]+1, current[1]-1) in ys and (current[0]+1, current[1]-1) not in visited:

stack.append((current[0]+1, current[1]-1))

visited.append((current[0]+1, current[1]-1))

ys.remove((current[0]+1, current[1]-1))

# checking upper left

if (current[0]-1, current[1]-1) in ys and (current[0]-1, current[1]-1) not in visited:

stack.append((current[0]-1, current[1]-1))

visited.append((current[0]-1, current[1]-1))

ys.remove((current[0]-1, current[1]-1))

# checking upper right

if (current[0]-1, current[1]+1) in ys and (current[0]-1, current[1]+1) not in visited:

stack.append((current[0]-1, current[1]+1))

visited.append((current[0]-1, current[1]+1))

ys.remove((current[0]-1, current[1]+1))

dfs\_counts.append(len(visited))

print(max(dfs\_counts))

covid('input 2.txt')

**Task 2**

def alien(file):

mat = []

aliens = []

humans = []

bfs\_counts = []

with open(file, 'r') as f:

lines = f.readlines()

for line in range(2, len(lines)):

t = lines[line].split(' ')

t[-1] = t[-1].replace('\n', '')

t[-1] = t[-1].replace('\t\t', '')

mat.append(t)

for row in range(len(mat)):

for column in range(len(mat[row])):

if mat[row][column] == 'A':

aliens.append((row, column))

if mat[row][column] == 'H':

humans.append((row, column))

visited = []

# bfs from every alien regions

for p in aliens:

queue = [p]

count = 0

while len(queue) > 0:

current = queue.pop(0)

human\_found = False

# checking up

if (current[0]-1, current[1]) in humans and (current[0]-1, current[1]) not in visited:

queue.append((current[0]-1, current[1]))

visited.append((current[0]-1, current[1]))

human\_found = True

# checking down

if (current[0]+1, current[1]) in humans and (current[0]+1, current[1]) not in visited:

queue.append((current[0]+1, current[1]))

visited.append((current[0]+1, current[1]))

human\_found = True

# checking left

if (current[0], current[1]-1) in humans and (current[0], current[1]-1) not in visited:

queue.append((current[0], current[1]-1))

visited.append((current[0], current[1]-1))

human\_found = True

# checking right

if (current[0], current[1]+1) in humans and (current[0], current[1]+1) not in visited:

queue.append((current[0], current[1]+1))

visited.append((current[0], current[1]+1))

human\_found = True

if human\_found == True:

count += 1

bfs\_counts.append(count)

print('Time:', str(max(bfs\_counts)), 'minutes')

if len(humans) == len(visited):

print("No one survived")

else:

print(str(len(humans)-len(visited)), 'survived')

alien("Question2 input1.txt")