CSE422 Lab 1

```
##############
# Question 1
##############

def dfs(row,column,count):

    #checking (x, y+1) right position for child

    try:

        if table[row][column+1]=='Y':

            if  str(row)+str(column+1) in visited :
                pass

            else:
                visited.append(str(row)+str(column+1))
                Stack.append(str(row)+str(column+1))
                count=count+1
                count=dfs(row,column+1,count)
                Stack.pop()

    except IndexError:
        pass


    #checking (x+1, y-1) diagonal for child

    try:

        if  column!=0:

            if table[row+1][column-1]=='Y':

                if  str(row+1)+str(column+1) in visited :
                    pass

                else:
                    visited.append(str(row+1)+str(column-1))
                    Stack.append(str(row+1)+str(column-1))
                    count=count+1
                    count=dfs(row+1,column-1,count)
                    Stack.pop()
```

```
except IndexError:
    pass


#checking (x+1, y) down position for child

try:

    if table[row+1][column]=='Y':

        if  str(row+1)+str(column) in visited :
            pass

        else:
            visited.append(str(row+1)+str(column))
            Stack.append(str(row+1)+str(column))
            count=count+1
            count=dfs(row+1,column,count)
            Stack.pop()

except IndexError:
    pass


#checking (x+1, y+1) diagonal position for child

try:

    if table[row+1][column+1]=='Y':

        if  str(row+1)+str(column+1) in visited :
            pass

        else:
            visited.append(str(row+1)+str(column+1))
            Stack.append(str(row+1)+str(column+1))
            count=count+1
            count=dfs(row+1,column+1,count)
            Stack.pop()

except IndexError:
    pass

return  count
```

```
#reading file

file = open("D:\Downloads\Spring  2022\Spring 2022
submissions\CSE422\CSE422 Lab\Lab1\input.txt")
lines_List = file.read().splitlines()
file.close()

table = [[]]

#storing values in the table

for i in range(len(lines_List)):

    for j in range(len(lines_List[0])):

        if (lines_List[i][j] != " "):

            table[i].append(lines_List[i][j])

    # removing extra row from table

    if (i != len(lines_List)-1):

        table.append([])


visited=[]
Stack=[]
infected=[]

# traverse the table

for row   in  range(len(table)):

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

        count=0

        #if find Y in the table applying DFS

        if(table[row][column]=='Y'):

            if str(row)+str(column) in visited :
                pass

            else:
                visited.append(str(row)+str(column))
                Stack.append(str(row)+str(column))
                count=count+1
                count=dfs(row,column,count)
                infected.append(count)
                Stack.pop()


print("maximum infected area : ",max(infected))


""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
==================================================================================
--------- Question 1 End ---------------------------------------------------------
==================================================================================
""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""


##############
# Question 2
##############

def bfs():

    level_visited=-1

    #when stack not empty

    while   len(stack)!=0:

        stack_length=len(stack)

        #visiting all nodes of a level

        for i   in  range(stack_length):

            current_node=stack.pop(0)

            row=int(current_node[0])

            column=int(current_node[1])
```

```
#checking position (x-1,y) ↑ for child

try:

    # if there is Human
    # put 'A' in that position in table and
    # push position in stack

    if  row!=0:

        if  table[row-1][column]=='H':

            table[row-1][column]='A'

            stack.append(str(row-1)+str(column))

except IndexError:
    pass


#checking position (x,y-1) ← for child

try:

    # if there is Human
    # put 'A' in that position in table and
    # push position in stack

    if  column!=0:

        if  table[row][column-1]=='H':

            table[row][column-1]='A'

            stack.append(str(row)+str(column-1))

except IndexError:
    pass


#checking position (x+1,y) ↓ for child

try:

    # if there is Human
```

```
                # put 'A' in that position in table and
                # push position in stack

                if  table[row+1][column]=='H':

                    table[row+1][column]='A'

                    stack.append(str(row+1)+str(column))

            except IndexError:
                pass


            #checking position (x,y+1) → for child

            try:

                # if there is Human
                # put 'A' in that position in table and
                # push position in stack

                if  table[row][column+1]=='H':

                    table[row][column+1]='A'

                    stack.append(str(row)+str(column+1))

            except IndexError:
                pass

        level_visited=level_visited+1

    return  level_visited


#reading file

file = open("D:\Downloads\Spring  2022\Spring 2022
submissions\CSE422\CSE422 Lab\Lab1\input.txt")
lines_List = file.read().splitlines()
file.close()

table = [[]]

#storing values in the table
```

```
for i in range(int(lines_List[0])):

    for j in range(len(lines_List[2])):

        if (lines_List[i+2][j] != " "):

            table[i].append(lines_List[i+2][j])

    # removing extra row from table

    if (i != int(lines_List[0])-1):

        table.append([])



stack=[]


# traverse the table

for row   in  range(len(table)):

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

        #finding all A in table as node 0
        #storing them in the stack

        if(table[row][column]=='A'):

            if str(row)+str(column) not in stack :

                stack.append(str(row)+str(column))

            else:
                pass


print("Time: ",bfs()," minutes")

# finding survived Humans, if any

survived=0
```

```
for row in range(len(table)):

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

        if  table[row][column]=='H':
            survived=survived+1

if  survived>0:

    print(survived," survived")

else:

    print("No one survived")
```