Assignment-8

ELP-780 Software Lab

Indian Institute of Technology

Name:	*Entry No:*
PALAKH SHANGLE	2017EET2292

September 27, 2018

Contents

1 Problem Statement 1

Statement: Find the two largest valid crosses that can be drawn on smart cells in the grid, and return two integers denoting the dimension of the each of the two largest valid crosses. In the above diagrams, our largest crosses have dimension of 1, 5 and 9 respectively
Note: The two crosses cannot overlap, and the dimensions of each of the valid crosses should be maximal

1.1 Input Format

- The first line contains two space-separated integers, n and m.
  Each of the next lines n contains a string of m characters where each character is either S (Smart) or D (Dull). These strings represent the rows of the grid. If the jth character in the ith line is S, then (i,j) is a cell smart. Otherwise it's a dull cell.
- **Sample Input:**
  5 6
  SSSSSS
  SDDDSD
  SSSSSS
  SSDDSD
  SSSSSS

1.2 Constraints

- 2 <= n <= 105
- 2 <= m <= 105

1.3 Output Format

- Find two valid crosses that can be drawn on smart cell of the grid, and return the dimension of both the crosses in the reverse sorted order(i.e. First Dimension should be the larger one and other should be smaller one).

- **Sample Output:**
  5 1

1.4 Algorithm

1. Each line of the textfile is read and depending upon the plan type, a new file is created
2. Then the user chooses from the 3 plan types
3. For each file type the average call duration is displayed. Then the total cost is calculated if the user wants

1.5 Implementation

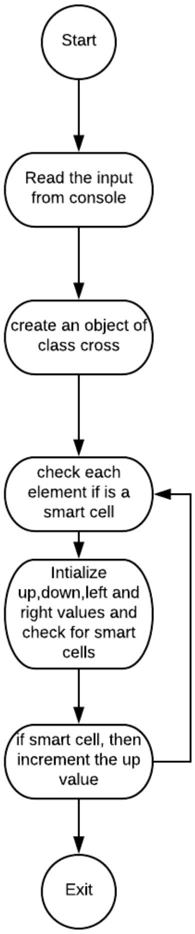
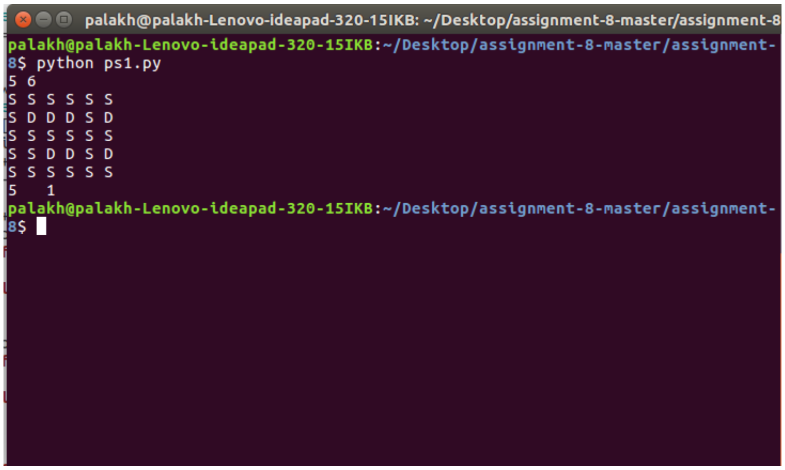

Start
Read the input from console
create an object of class cross
check each element if is a smart cell
Intialize up,down,left and right values and check for smart cells
if smart cell, then increment the up value
Exit

Figure 1: Flowchart of first program

1.6 Screenshots



palakh@palakh-Lenovo-ideapad-320-15IKB: ~/Desktop/assignment-8-master/assignment-8

```
palakh@palakh-Lenovo-ideapad-320-15IKB:~/Desktop/assignment-8-master/assignment-
8$ python ps1.py
5 6
S S S S S S
S D D D S D
S S S S S S
S S D D S D
S S S S S S
5   1
palakh@palakh-Lenovo-ideapad-320-15IKB:~/Desktop/assignment-8-master/assignment-
8$ 
```

Figure 2: Screenshot of Problem 1

2 Problem Statement 2

Statement: After, getting mix results of valid crosses, professors decided to test the computation abilities on one more problem. This time professors wanted to test the decryption capabilities of the computer. Encryption of a message requires three keys, k1, k2, and k3. The **26 letters of English and underscore are divided in three groups, [a-i] form one group, [j-r] a second group, and everything else ([s-z] and underscore) the third group**. Within each group the letters are rotated left by ki positions in the message. Each group is rotated independently of the other two. Decrypting the message means doing a right rotation by ki positions within each group.

2.1 Input Format

- **Sample Input**
  - All input strings comprises of only lowercase English alphabets and underscores()
    2 3 4
    dikhtkoreytecocsusrswehas

2.2 Constraints

- 1<=Length of the string<=150
- 1, =ki<=150(i=1,2..)

2.3 Output Format

- **Sample Output**
  For each encrypted message, the output is a single line containing the decrypted string
  hardworkisthekeytosuccess

2.4 Algorithm

- The inputs are read from the console
- Then 3 lists are maintained for each of the group

- Each character is read from the input line, and checked which list it belongs to, and accordingly it is added to a new list
- The new list is then rotated
- Then according to the new rotated list the output is printed

2.5 Implementation

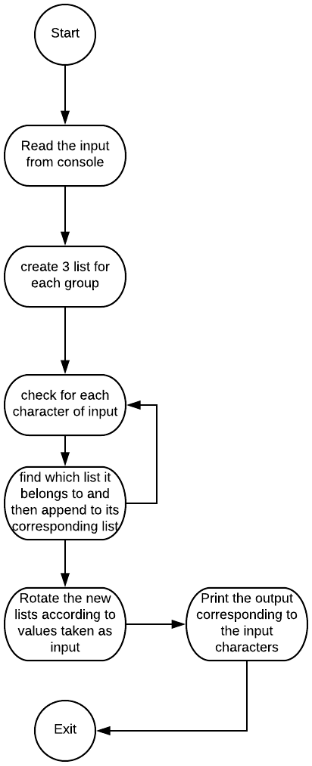
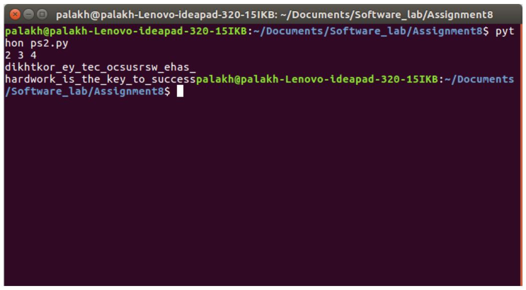

Start
Read the input from console
create 3 list for each group
check for each character of input
find which list it belongs to and then append to its corresponding list
Rotate the new lists according to values taken as input
Print the output corresponding to the input characters
Exit

Figure 3: Flowchart of second program

2.6 Screenshots



palakh@palakh-Lenovo-ideapad-320-15IKB: ~/Documents/Software_lab/Assignment8

```
palakh@palakh-Lenovo-ideapad-320-15IKB:~/Documents/Software_lab/Assignment8$ pyt
hon ps2.py
2 3 4
dikhtkor_ey_tec_ocsusrsw_ehas_
hardwork_is_the_key_to_successpalakh@palakh-Lenovo-ideapad-320-15IKB:~/Documents
/Software_lab/Assignment8$
```

Figure 4: Screenshot of Problem 2

3 Appendix

3.1 Code-1

```

out=[]

class cross:  #class definition
  def __init__(self, data):
    self.data = data
    self.top = 0
    self.down = 0
    self.left = 0
    self.right = 0

x = input()     # reading the values of n,m
row = int(x.split(” ”)[0])
col = int(x.split(” ”)[1])

mat=[[0 for i in range(col)]for j in range(row)]

for i in range(row):
  mat[i]=input().split(” ”) #reading cross values and storing it in a matrix

for i in range(row):
    for j in range(col):  #for each element assigning data members of class
        mat[i][j]= cross(mat[i][j])



for i in range(row):
  for j in range(col):
    if(mat[i][j].data == ’S’):  #iterating loop over only smart cells
      up = i-1        #if smart cell then top will be above element
      d = i+1         #if smart cell then top will be
```

```
       below element
    l = j-1          #if smart cell then top will be
       leftmost element and starting loop
    r = j+1          #if smart cell then right will
       be nexr right and starting loop
    for up in range(i-1,-1,-1):
      if(mat[up][j].data != 'D'): #iterating till
         not getting dull cell
        mat[i][j].top=mat[i][j].top+1 #if smart cell
           then increment value
      else:
        break

    for down in range(i+1,row):
      if(mat[down][j].data != 'D'):      #iterating
         till not getting dull cell
        mat[i][j].down=mat[i][j].down+1   #if smart
           cell then increment value
      else:
        break

    for l in range(j-1,-1,-1):
      if(mat[i][l].data != 'D'):         #iterating
         till not getting dull cell
        mat[i][j].left=mat[i][j].left+1   #if smart
           cell then increment value
      else:
        break

    for r in range(j+1,col):
      if(mat[i][r].data != 'D'):         #iterating
         till not getting dull cell
        mat[i][j].right=mat[i][j].right+1 #if smart
           cell then increment value
      else:
        break

    temp=min(mat[i][j].top,mat[i][j].down,mat[i][j].left,mat[i][j].rig
            #finding min
        out.append(temp);

n=len(out)
out.sort()
print(out[n-1]*4+1," ",out[n-2]*4+1)  #printing
    largest and 2nd largest

####### write your code here ##########
```

ps1.py

3.2 Code-2

```

def rotate(l, n):     #Function to rotate the list
    return l[-n:] + l[:-n]

k=input()          #Taking input from user
k1=int(k.split(" ")[0])
k2=int(k.split(" ")[1])
k3=int(k.split(" ")[2])
encrypt=input()

list1=['a','b','c','d','e','f','g','h','i'] # defining 3 list for each group
list2=['j','k','l','m','n','o','p','q','r']
list3=['s','t','u','v','w','x','y','z','_']
out=[]

list4=[]
list5=[]
list6=[]
for i in range(len(encrypt)):   #reading the input character by character
  #print(encrypt[i])
  if encrypt[i] in list1: #checking character belongs to which list , accordingly adding it
    list4.append(encrypt[i])
  if encrypt[i] in list2:
    list5.append(encrypt[i])
  if encrypt[i] in list3:
    list6.append(encrypt[i])


list4=rotate(list4,k1)       #list rotated
list5=rotate(list5,k2)
list6=rotate(list6,k3)

l1=0
```

```
l2=0
l3=0

for i in range(len(encrypt)):
  if encrypt[i] in list1:   #checking if character belongs to list1
    #print(list4[l1])
    out.append(list4[l1]) #if yes then appending rotated char value
    l1=l1+1          #Incrementing the value of counter to access the elemnet of rotated list
    #print("l1",l1)
  if encrypt[i] in list2:   #checking if character belongs to list2
    #print(list5[l2])
    out.append(list5[l2]) #if yes then appending rotated char value
    l2=l2+1          #Incrementing the value of counter to access the elemnet of rotated list
    #print("l2",l2)
  if encrypt[i] in list3:   #checking if character belongs to list3
    #print(list6[l3])
    out.append(list6[l3]) #if yes then appending rotated char value
    l3=l3+1          #Incrementing the value of counter to access the elemnet of rotated list
    #print("l3",l3)

for i in range(len(out)):
  print(out[i],end="")
###### write your code here ##########
```

ps2.py