Assignment-8

ELP 780 Software Lab

Pushpendra Singh Dahiya
2017EET2680
2017-19

A Report presented for the Assignment on

Experiment 8

Electrical Department
IIT DELHI
India

September 27, 2018

Contents

1 Problem Statement 1

1.1 Problem Statement

IIT Delhi, has just got the strongest computer. The professors in charge wants to check the computational capacity of the computer. So, they decided to create the problem which is to be given as an assignment to students. Can you help the professor to check the computation capability of the computer?

A valid cross is defined here as the two regions (horizontal and vertical) of equal lengths crossing over each other. These lengths must be odd, and the middle cell of its horizontal region must cross the middle cell of its vertical region. [1]

1.2 Assumptions

1. The input is provided from stdin

1.3 Algorithm

1. Take the inputs in m, n
2. Initialize s and arr of size m*n
3. Read the input characters in s
4. For each element in s, check if it forms a cross
5. Store the length of cross in the array arr
6. if the element is dull the size of cross is 0
7. Initialize two numbers for maximum and 2nd maximum
8. Find the maximum and second maximum and print them
9. exit.

1.4 Input and Output Format

- **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.

  Constraints

  1. 2 <= n <= 105
  2. 2 <= m <= 105

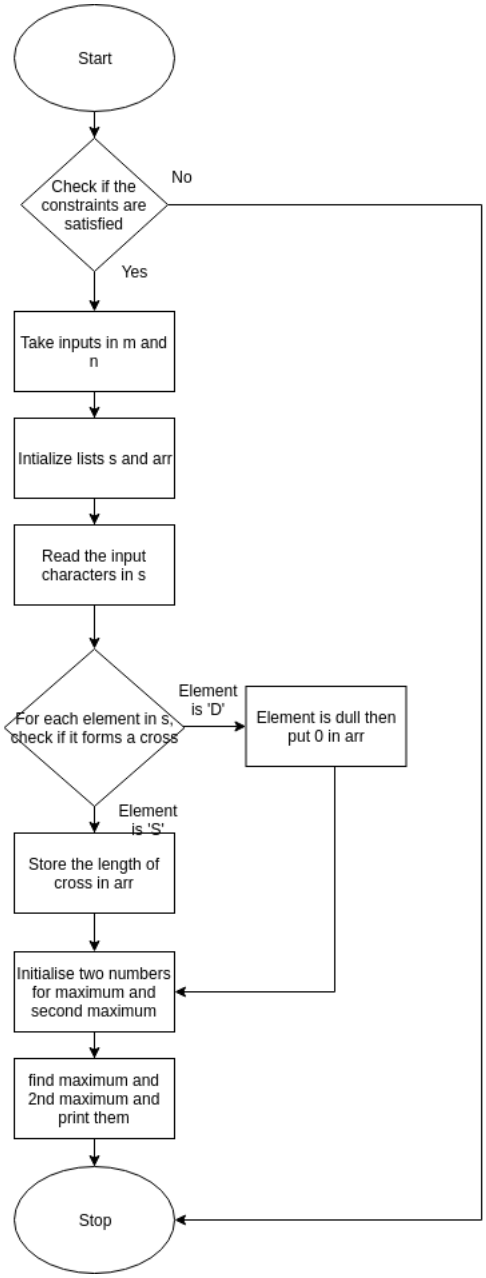
  Output format
  Find two valid crosses that can be drawn on smart cell of the grid, and return the dimension of both the crosses in the reverse sorted order(i.e. First Dimension should be the larger one and other should be smaller one).

1.5 Test Cases

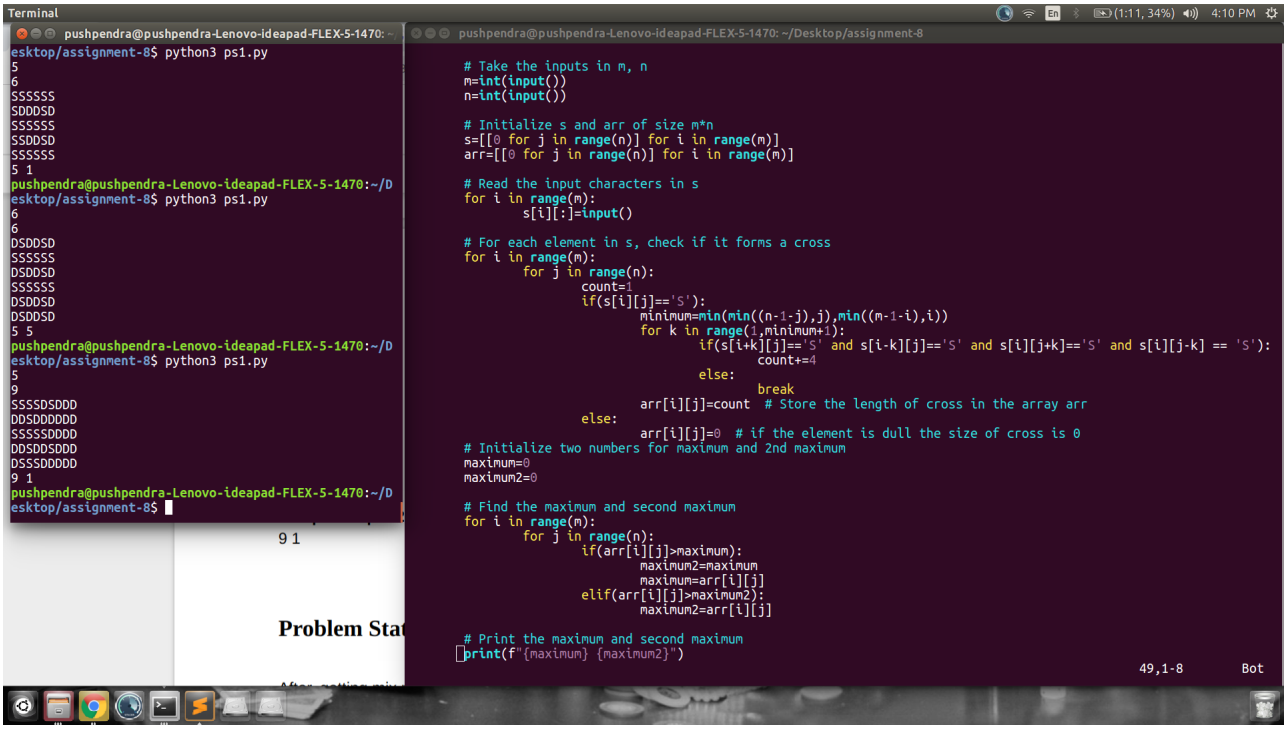
Input
python3 ps1.py
5
6
SSSSSS
SDDDSD
SSSSSS
SSDDSD
SSSSSS

Output
5 1

1.6 Flowchart



1.7 Screenshots



1.8 Difficulties /Issues faced

NONE

2 Problem Statement 2

2.1 Problem 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]

2.2 Assumptions

Input is provided from stdin.

2.3 Algorithm

1. Take the inputs in k1, k2, k3 and s
2. Initailize 3 empty list for the 3 range of characters
3. Initialize list for output string
4. Check the character belong to which range
5. Rotate the elements of list by the value given in k
6. Place the rotated elements in the out list
7. Print the out list by converting it to a string
8. Exit

2.4 Input and Output Format

- **Input format**
  All input strings comprises of only lowercase English alphabets and underscores(_).

  Constraints

  1. 1 <= Length of the string <=150
  2. 1 <= ki <=150 (i=1,2,3)

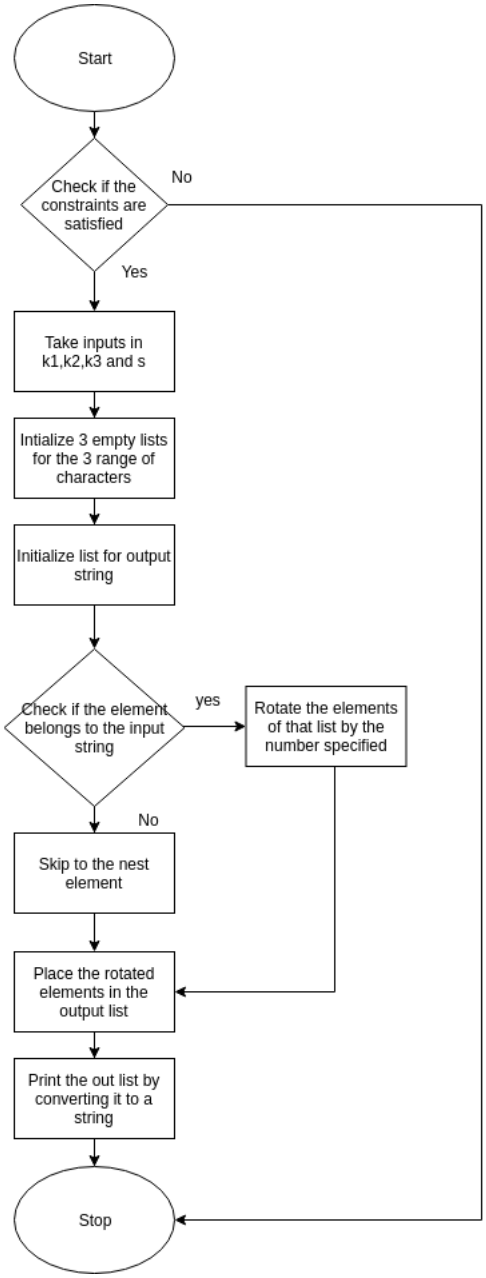
- **Output format**
  For each encrypted message, the output is a single line containing the decrypted string.

2.5 Test Cases

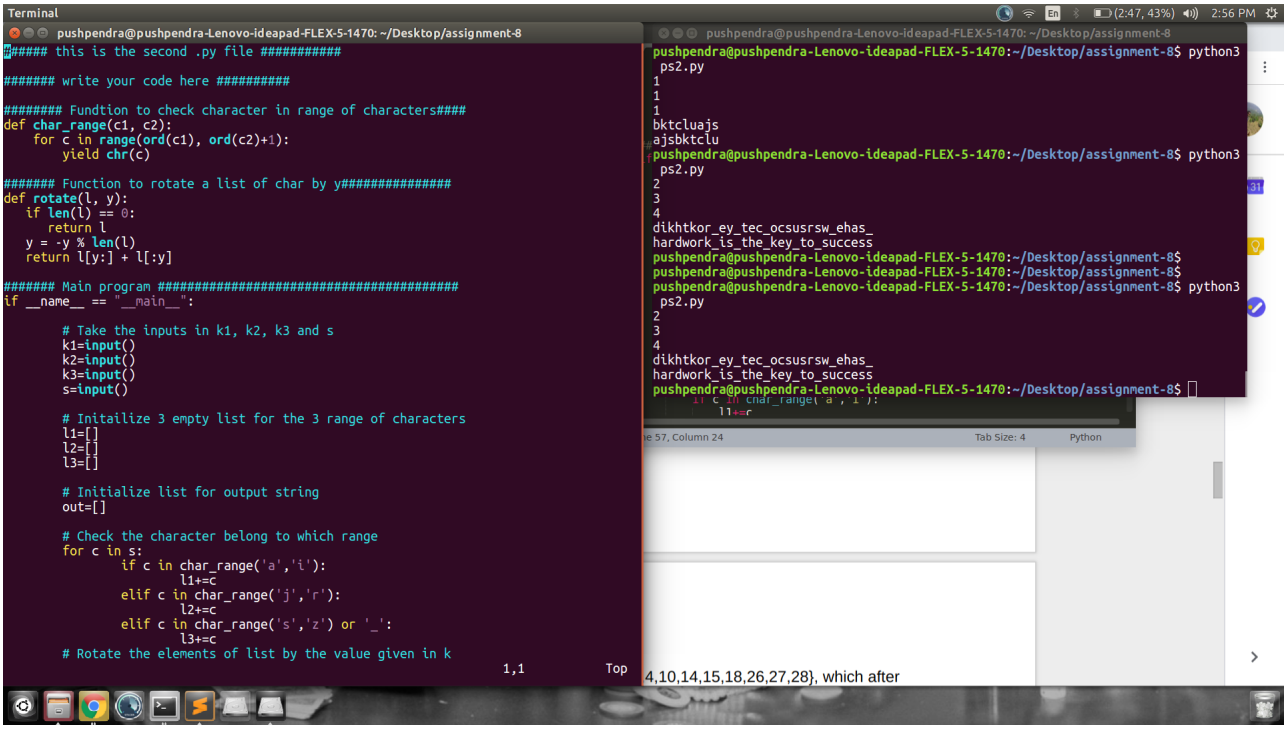
Input
2 3 4
dikhtkor_ey_tec_ocsusrsw_ehas_

Output
hardwork_is_the_key_to_success

2.6 Flowchart



2.7 Screenshots



2.8 Difficulties /Issues faced

To join the list back to string

3 Appendix

3.1 Appendix-A: code for ps1

```
###### this is the first .py file ###########

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


############# Main Function ##################
if __name__ == "__main__":

        # Take the inputs in m, n
        m=int(input())
        n=int(input())

        # Initialize s and arr of size m*n
        s=[[0 for j in range(n)] for i in range(m)]
        arr=[[0 for j in range(n)] for i in range(m)]

        # Read the input characters in s
        for i in range(m):
                s[i][:]=input()

        # For each element in s, check if it forms a cross
        for i in range(m):
                for j in range(n):
                        count=1
                        if(s[i][j]=='S'):
                                minimum=min(min((n-1-j),j),min((m-1-i),
                                for k in range(1,minimum+1):
                                        if(s[i+k][j]=='S' and s[i-k][j]=
                                                count+=4
                                        else:
                                                break
                                arr[i][j]=count  # Store the length of
                        else:
                                arr[i][j]=0  # if the element is dull th
```

```
# Initialize two numbers for maximum and 2nd maximum
maximum=0
maximum2=0

# Find the maximum and second maximum
for i in range(m):
        for j in range(n):
                if(arr[i][j]>maximum):
                        maximum2=maximum
                        maximum=arr[i][j]
                elif(arr[i][j]>maximum2):
                        maximum2=arr[i][j]

# Print the maximum and second maximum
print(f"{maximum} {maximum2}")
```

3.2 Appendix-B: code for ps2.l

```
###### this is the second .py file ###########

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

######## Fundtion to check character in range of characters####
def char_range(c1, c2):
    for c in range(ord(c1), ord(c2)+1):
        yield chr(c)

####### Function to rotate a list of char by y###############
def rotate(l, y):
   if len(l) == 0:
      return l
   y = -y % len(l)
   return l[y:] + l[:y]

####### Main program ##########################################
if __name__ == "__main__":

        # Take the inputs in k1, k2, k3 and s
        k1=input()
        k2=input()
        k3=input()
        s=input()

        # Initailize 3 empty list for the 3 range of characters
        l1=[]
        l2=[]
        l3=[]

        # Initialize list for output string
        out=[]

        # Check the character belong to which range
        for c in s:
                if c in char_range('a','i'):
```

```
                l1+=c
        elif c in char_range('j','r'):
                l2+=c
        elif c in char_range('s','z') or '_':
                l3+=c
# Rotate the elements of list by the value given in k
l1=rotate(l1,int(k1))
l2=rotate(l2,int(k2))
l3=rotate(l3,int(k3))

# Place the rotated elements in the out list
for c in s:
        if c in char_range('a','i'):
                out.append(l1.pop(0))
        elif c in char_range('j','r'):
                out.append(l2.pop(0))
        elif c in char_range('s','z') or '_':
                out.append(l3.pop(0))

#Print the out list by converting it to a string
print(''.join(out))
```

References

[1] Python 3.7.1rc1 documentation. https://docs.python.org/3/.

[2] Git Tutorial. https://www.atlassian.com/git/tutorials.