Assignment-8

ELP - 718 Telecom Software Laboratory

Arif Khan

2018JTM2242

2018-2020

A report presented for the assignment on Python Basic and Github

Bharti School Of

Telecommunication Technology and Management

IIT Delhi

India

September 27, 2018

Contents

1 Problem Statement-1

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.
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 Program Structure

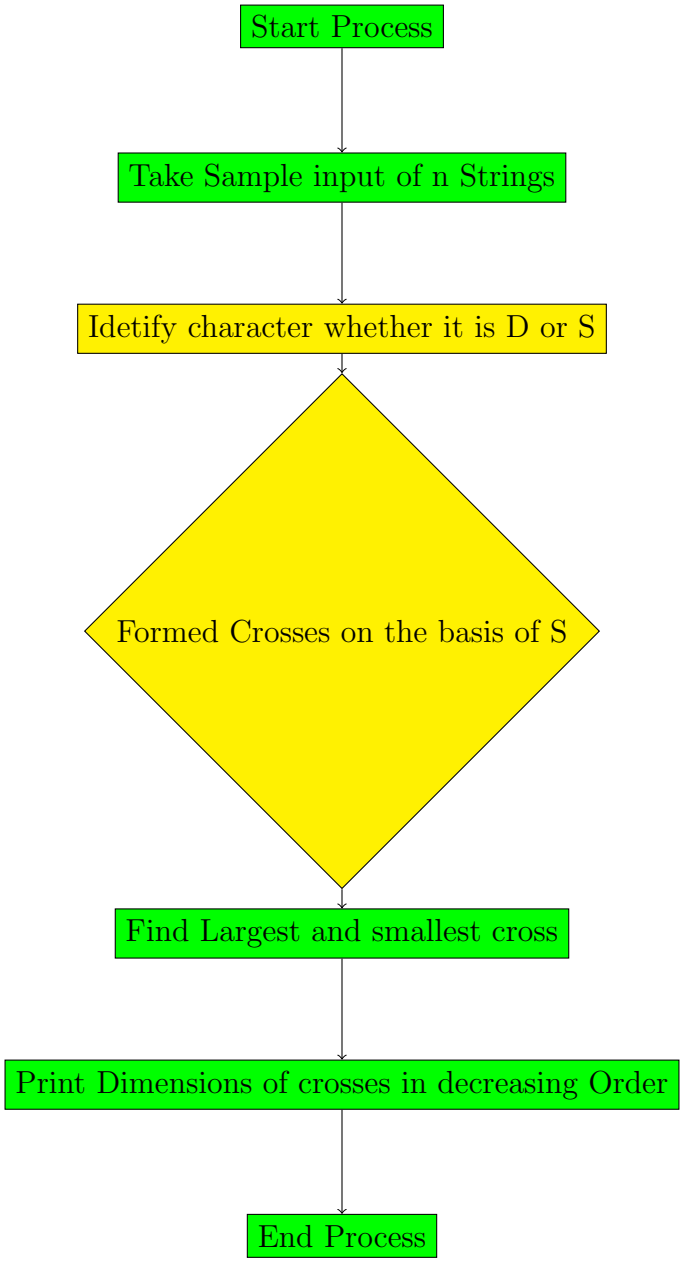
1.2 Difficulties

- We faced difficulties during overcome the problem of overlap of crosses.
- We faced difficulties during formation of cross.
- Faced problem to remove error .
- We faced problems during execute the code.

1.3 Algorithm

- Take N number of sample strings to formed crosses.
- Identify the character in each string whether it is DULL(D) or SMART(S).
- On the basis of "S" we create crosses.
- Find largest and smaller dimensions crosses.
- Print the Dimensions in Decreases order.

1.4 Flowchart



1.5 Input Format

Input is given through a text file.

1.6 Output Format

Output is taken through user commmand line argument.

2 Problem Statement-2

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 Program Structure

```
> ###### this is the second .py file ############
>
> ######## write your code here ##########
> #rotate function
> def rotate(lst,x):
```

```
>      copy = list(lst)
>      for i in range(len(lst)):
>          if x<0:
>              lst[i+x] = copy[i]
>          else:
>              lst[i] = copy[i-x]
>
>
> #Create 3 groups
> g1="abcdefghi"
> g2="jklmnopqr"
> g3="stuvwxyz_"
>
> c1 =[]
> c2 =[]
> c3 =[]
> index1=[]
> index2=[]
> index3=[]
>
> #get key vakue from user
> k1,k2,k3 = list(map(int,input().split()))
>
> #get string
> msg = input()
```

```
> msg_list = list(msg)
> print(msg_list)
>
> #now compair g1 in string and copy similaar char into s1
> for i in range(0,len(msg)):
>     if msg_list[i] in g1:
>         c1.append(msg_list[i])
>         index1.append(i)
>
>     elif msg_list[i] in g2:
>         c2.append(msg_list[i])
>         index2.append(i)
>     elif msg_list[i] in g3:
>         c3.append(msg_list[i])
>         index3.append(i)
>
>
>
> #rotate c1,c2,c3
> rotate(c1,k1)
> rotate(c2,k2)
> rotate(c3,k3)
>
>
>
```

```
> #get decrypted msg
> p=q=r=0
> for i in range(0,len(msg)+1):
>     if i in index1:
>         msg_list[i]=c1[p]
>         p+=1
>     elif i in index2:
>         msg_list[i]=c2[q]
>         q+=1
>     elif i in index3:
>         msg_list[i]=c3[r]
>         r+=1
>
> print(msg_list)
>
> for i in msg_list[:]:
>     print (i, end ='')
>
> print(\n)
```

2.2 Algorithm

- Take N number of sample strings.
- We divide all alphabets in K1,K2 AND K3 keys.
- For encryption we rotate characters left corresponding to Key.
- For Decryption we rotate characters Right corresponding to Key.

- Print the Decrypted string.

2.3 Input Format

2.4 Output Format

2.5 Difficulties

- We faced difficulties during overcome the problem of repetation of character.
- We faced difficulties during Decryption and Encryption of strings.
- Faced problem to remove error .
- We faced problems during execute the code.

2.6 Flowchart