**Assignment 2 – PS06 - [Manufacture Footwear]**

**Group Name:** DSAD Group 70

**Using Python 3.7**

**Contribution Table:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl. No.** | **Name (as appears in Canvas)** | **ID NO** | **Contribution** |
| 1 | JATINDER KUMAR CHAURASIA | 2021c104115 | 100% |
| 2 | RAHUL NARESHRAO SHASTRI | 2021c104122 | 100% |
| 3 | SWAPNIL TIWARI | 2021c104116 | 100% |

**Design and Time complexity analysis:**

In this, we are following greedy method to solve the problem.

Greedy Method - Greedy methods makes the best choice at every step and attempts to find the optimal way to solve the whole problem.

**Algorithm:**

Read through the inputPS06.txt file and iterate over the lines

For each line l1:

prev= [] # to store the previous element which are iterated.

matched= 0

If prev=[] at first iteation:

prev=l1

Elseif last element added to prev == l1

prev= prev+l1

elseif last element added to prev not equal to l1

remove the last element from prev

if prev is empty after any of the iteration except first one

increment matched by 1

write the matched value to output file (outputPS06.txt)

**# How it is efficient:**

# by keeping in memory the previous items which are not added yet to packs, this is the best solution we found using greedy approach.

**# complexity of this solution is: O(n^2)**

We are using variables to store the previous state and iterating to next one to find the min packs we can make from shelf of LR assuming the shelf do have equal number of L and R.

* Variables to store the previous data that are not yet put in packs: **prev**
* Variables to store the packs number we can make after the full iteration: **matched**

Then using for loop iterating over the shelf, we are checking if there is suitable pack of LR

* **Conditions:**

If prev state is [] – then it will store the current element

If prev\_state[-1] =’L’ and current one is ‘L’ then the append the current to previous

If prev\_state[-1] =’R’ and current one is ‘R’ then the append the current to previous

If prev\_state[-1] = ‘L’ and current one is ‘R’ or opposite of it . then it will remove the last element from the prev and move to next one

If prev\_state = [] after any of the iteration , then increment the matched variable to 1.

verifying the inputs:

**Input(inputPS06.txt):**

LRLLRR  
LRRRRLLLRL  
LRLLLRRLRR  
LLLLRRRR

**Output(outputPS06.txt):**

2  
3  
2  
1

—----------------------------------------------------Thanks —--------------------------------------------------------------