Python and SQL Practice

[**1141. User Activity for the Past 30 Days I**](https://leetcode.com/problems/user-activity-for-the-past-30-days-i/)

select to\_char(activity\_date, 'yyyy-mm-dd') as day, count(distinct(user\_id)) as active\_users

from Activity

where trunc(activity\_date) between to\_date('2019-07-27','yyyy-mm-dd') - 29

and to\_date('2019-07-27','yyyy-mm-dd')

group by activity\_date ;

-----------------------------------------------------------------------------------------------

<https://www.hackerearth.com/practice/basic-programming/input-output/basics-of-input-output/practice-problems/algorithm/split-house-547be0e9/>

*# name = input()                  # Reading input from STDIN*

*# print('Hi, %s.' % name)         # Writing output to STDOUT*

def h\_check(l):

    f = 0

    for i in range(len(l)-1):

        if l[i] == 'H' and l[i+1] == 'H':

            f = 1

            break

    if f == 1:

        return False

    else:

        return True

def fill(l):

    s = []

    for i in l:

        if i == '.':

            s.append('B')

        else:

            s.append(i)

    return "".join(s)

n = int(input())

v = input()

if h\_check(v):

    print('YES')

    print(fill(v))

else:

    print('NO')

[**1148. Article Views I**](https://leetcode.com/problems/article-views-i/)

select distinct(author\_id) as id from views where author\_id = viewer\_id order by author\_id;

[**1179. Reformat Department Table**](https://leetcode.com/problems/reformat-department-table/)

select id,

max((case when month = 'Jan' then revenue else null end)) as Jan\_Revenue,

max((case when month = 'Feb' then revenue else null end)) as Feb\_Revenue,

max((case when month = 'Mar' then revenue else null end)) as Mar\_Revenue,

max((case when month = 'Apr' then revenue else null end)) as Apr\_Revenue,

max((case when month = 'May' then revenue else null end)) as May\_Revenue,

max((case when month = 'Jun' then revenue else null end)) as Jun\_Revenue,

max((case when month = 'Jul' then revenue else null end)) as Jul\_Revenue,

max((case when month = 'Aug' then revenue else null end)) as Aug\_Revenue,

max((case when month = 'Sep' then revenue else null end)) as Sep\_Revenue,

max((case when month = 'Oct' then revenue else null end)) as Oct\_Revenue,

max((case when month = 'Nov' then revenue else null end)) as Nov\_Revenue,

max((case when month = 'Dec' then revenue else null end)) as Dec\_Revenue

from department

group by id;

/\* Write your PL/SQL query statement below \*/

select \* from department

PIVOT(

sum(revenue)

for month in (

'Jan' as Jan\_Revenue,

'Feb' as Feb\_Revenue,

'Mar' as Mar\_Revenue,

'Apr' as Apr\_Revenue,

'May' as May\_Revenue,

'Jun' as Jun\_Revenue,

'Jul' as Jul\_Revenue,

'Aug' as Aug\_Revenue,

'Sep' as Sep\_Revenue,

'Oct' as Oct\_Revenue,

'Nov' as Nov\_Revenue,

'Dec' as Dec\_Revenue

)

);

<https://www.hackerearth.com/practice/basic-programming/input-output/basics-of-input-output/practice-problems/algorithm/cartag-948c2b02/>

**Ali and Helping innocent people**

*# name = input()                  # Reading input from STDIN*

*# print('Hi, %s.' % name)         # Writing output to STDOUT*

def check\_tag(tag):

    l = [1,2,5,6]

    vowels = ['A','E','I','O','U','Y']

    f = 0

    for i in range(len(tag)-1):

        if i not in l:

            if ((int(tag[i]) + int(tag[i+1])) % 2) != 0:

                f = 1

        elif i == 2:

            if tag[i] in vowels:

                f = 1

    if f == 0:

        print("valid")

    else:

        print("invalid")

tag = input()

check\_tag(tag)

[**1661. Average Time of Process per Machine**](https://leetcode.com/problems/average-time-of-process-per-machine/)

<https://leetcode.com/problems/average-time-of-process-per-machine/description/>

/\* Write your PL/SQL query statement below \*/

with cte1 as(

    select \* from Activity where activity\_type = 'start'

), cte2 as(

    select \* from Activity where activity\_type = 'end'

)

-- select \* from start;

select cte1.machine\_id, round(sum(cte2.timestamp-cte1.timestamp)/count(cte1.machine\_id),3)  as processing\_time

from cte1 inner join cte2

on cte1.machine\_id = cte2.machine\_id

and cte1.process\_id = cte2.process\_id

group by cte1.machine\_id;

OR

select a1.machine\_id, round(avg(a2.timestamp-a1.timestamp), 3) as processing\_time

from Activity a1, Activity a2

where a1.machine\_id=a2.machine\_id and a1.process\_id=a2.process\_id

and a1.activity\_type='start' and a2.activity\_type='end'

group by a1.machine\_id;