National University of Science Technology Islamabad

School of Mechanical & Manufacturing Engineering

Artificial Intelligence

Assignment # 3

Submitted to:

Dr Yasir Ayaz

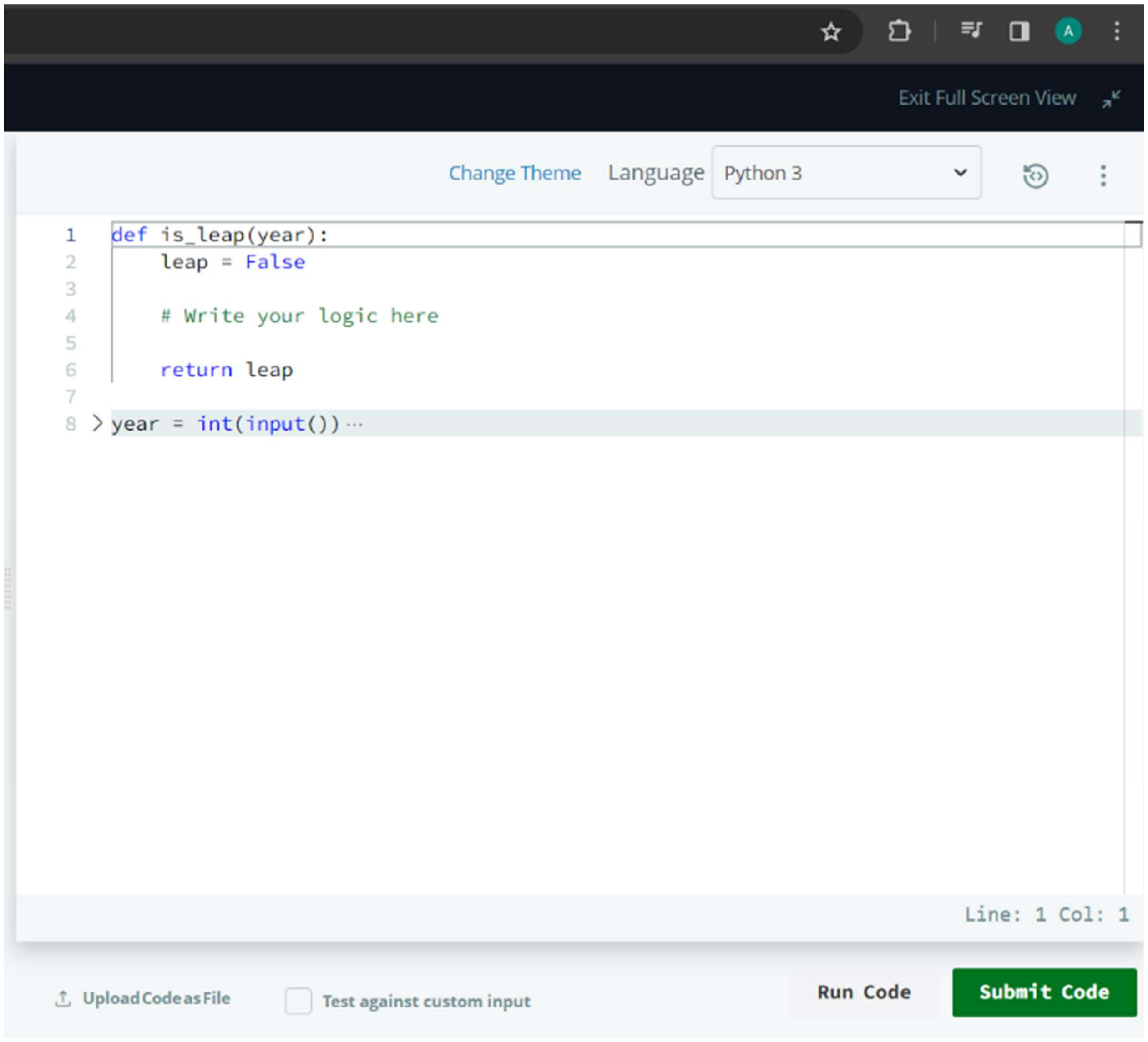
Submitted by:

Saeed Javaid

399756

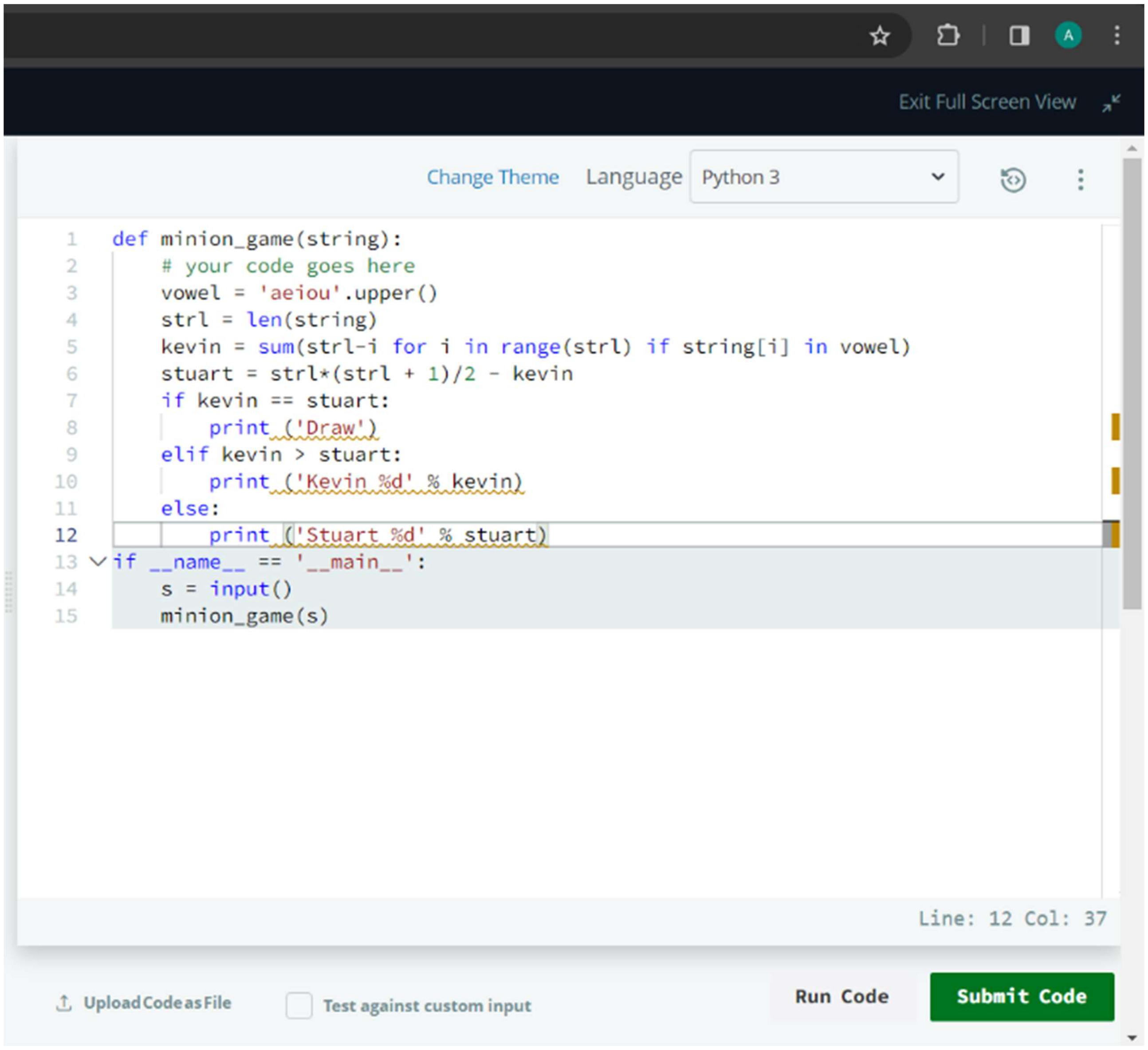
MS Mechanical Engineering

3rd semester



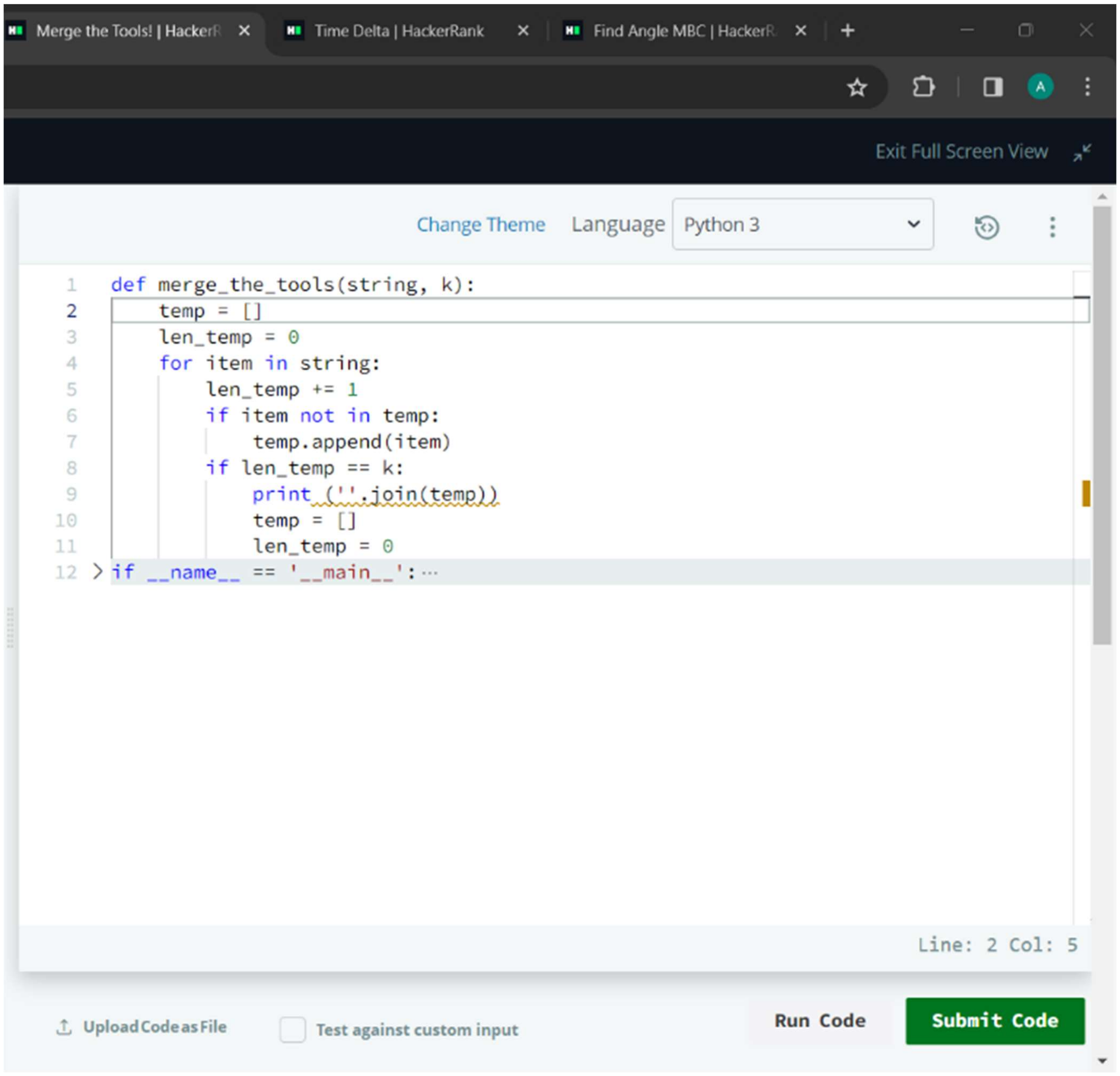
```python
def is_leap(year):
    leap = False

    # Write your logic here

    return leap

year = int(input())···
```

Figure 1 Write a Function



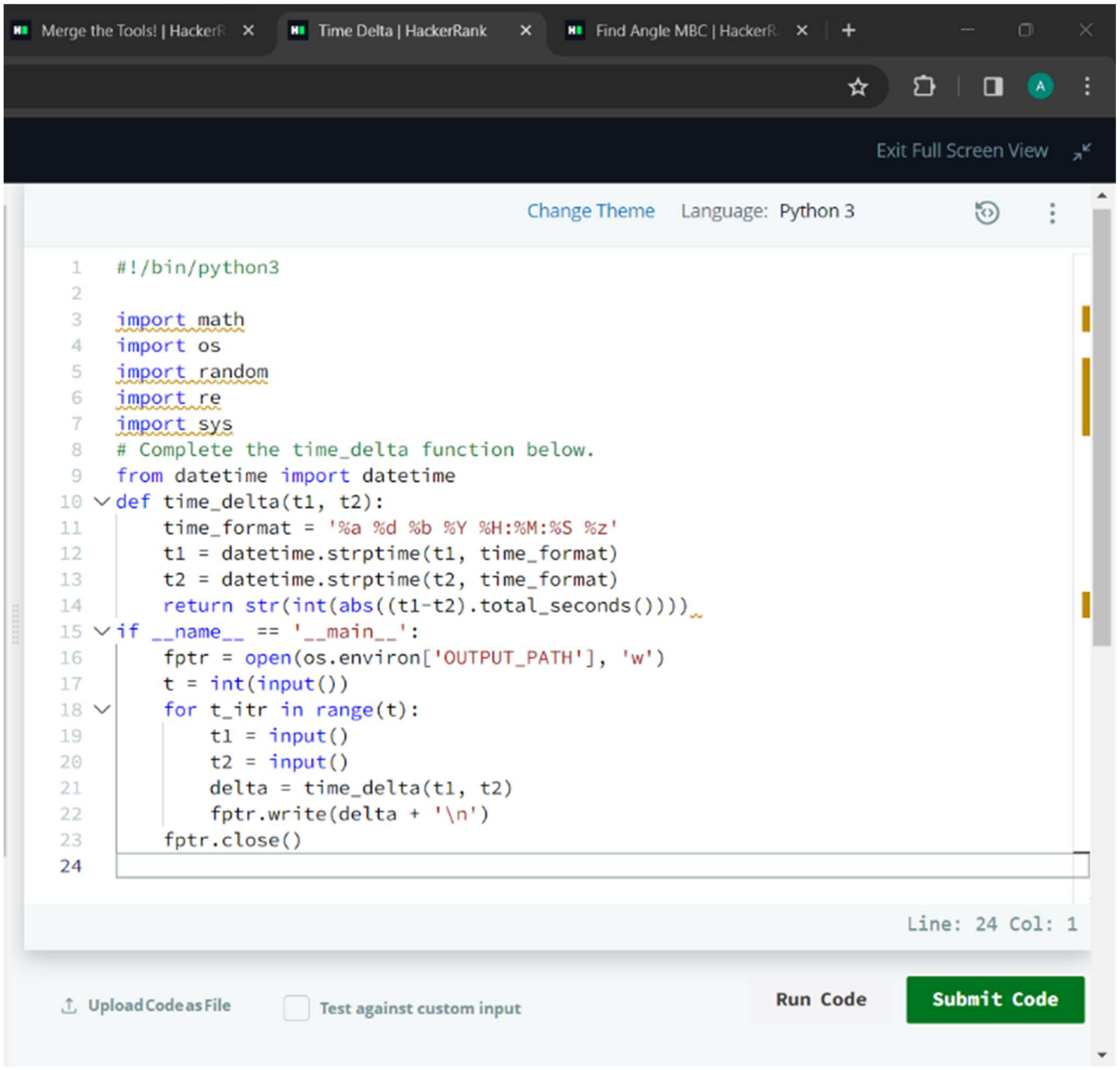
```python
def minion_game(string):
    # your code goes here
    vowel = 'aeiou'.upper()
    strl = len(string)
    kevin = sum(strl-i for i in range(strl) if string[i] in vowel)
    stuart = strl*(strl + 1)/2 - kevin
    if kevin == stuart:
        print ('Draw')
    elif kevin > stuart:
        print ('Kevin %d' % kevin)
    else:
        print ('Stuart %d' % stuart)
if __name__ == '__main__':
    s = input()
    minion_game(s)
```

Figure 2 The Minion Game



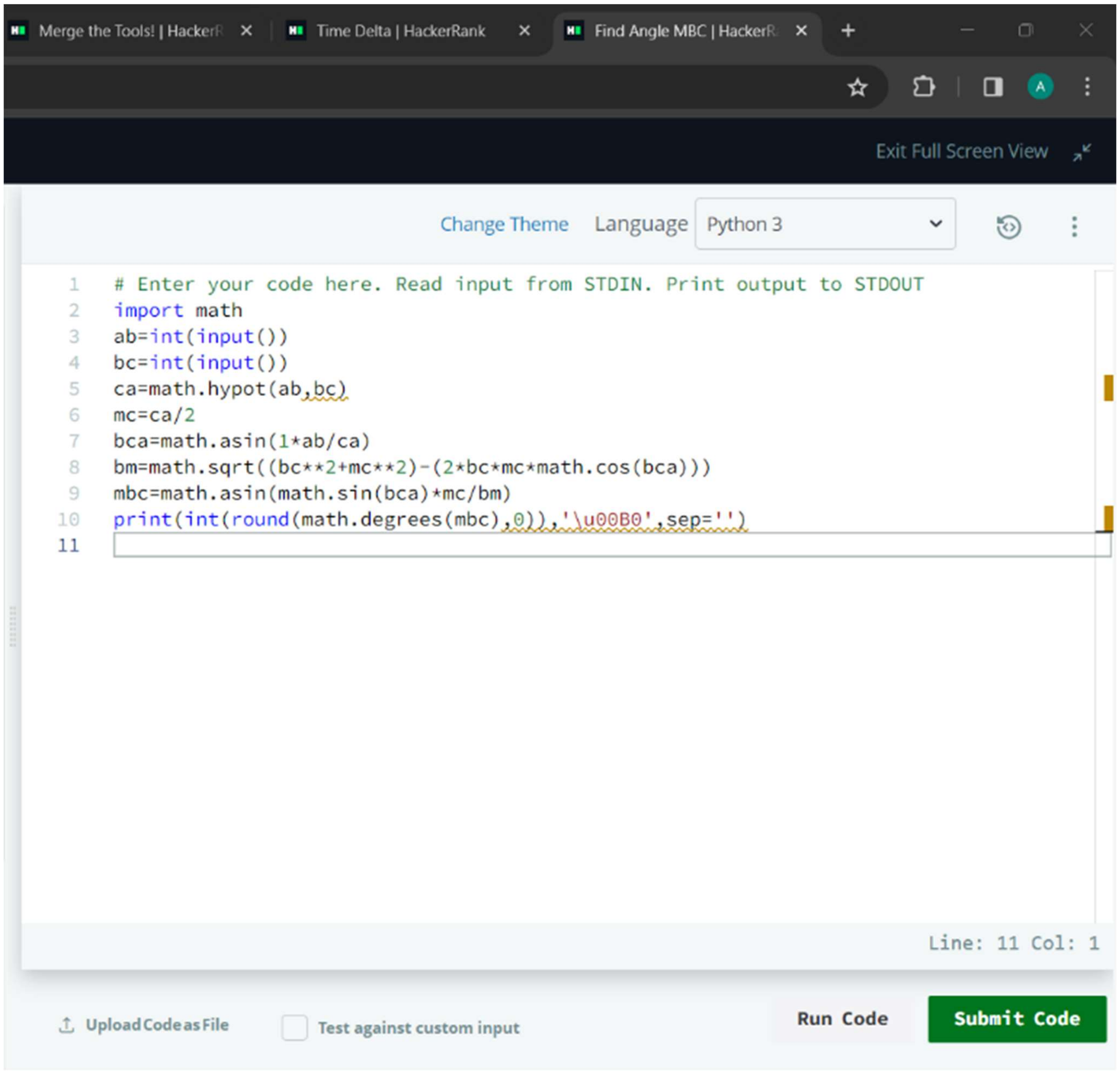
```python
def merge_the_tools(string, k):
    temp = []
    len_temp = 0
    for item in string:
        len_temp += 1
        if item not in temp:
            temp.append(item)
        if len_temp == k:
            print (''.join(temp))
            temp = []
            len_temp = 0
if __name__ == '__main__':...
```

Figure 3 Merge the Tools



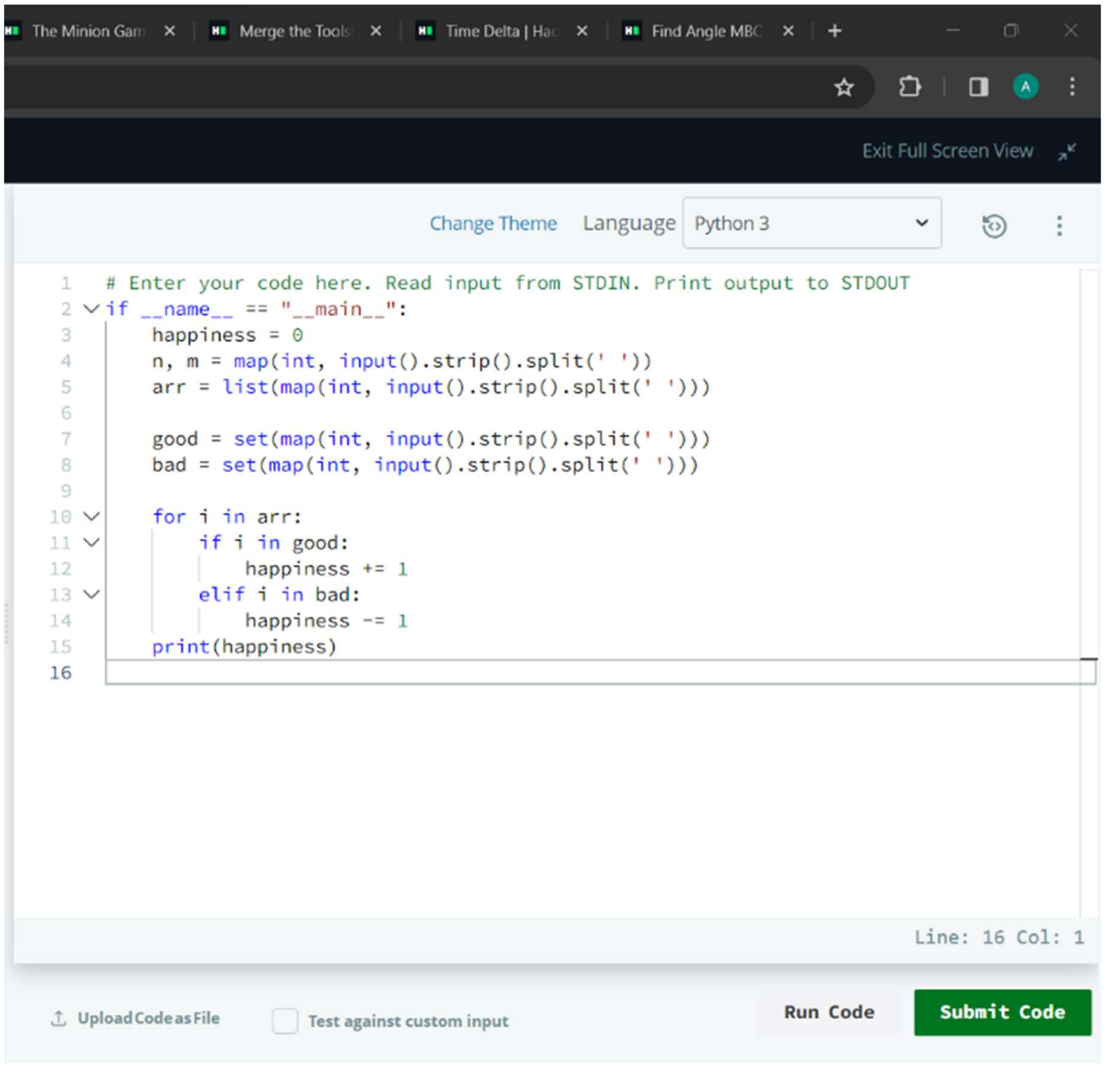
```python
#!/bin/python3

import math
import os
import random
import re
import sys
# Complete the time_delta function below.
from datetime import datetime
def time_delta(t1, t2):
    time_format = '%a %d %b %Y %H:%M:%S %z'
    t1 = datetime.strptime(t1, time_format)
    t2 = datetime.strptime(t2, time_format)
    return str(int(abs((t1-t2).total_seconds())))
if __name__ == '__main__':
    fptr = open(os.environ['OUTPUT_PATH'], 'w')
    t = int(input())
    for t_itr in range(t):
        t1 = input()
        t2 = input()
        delta = time_delta(t1, t2)
        fptr.write(delta + '\n')
    fptr.close()
```

Figure 4 Time Delta



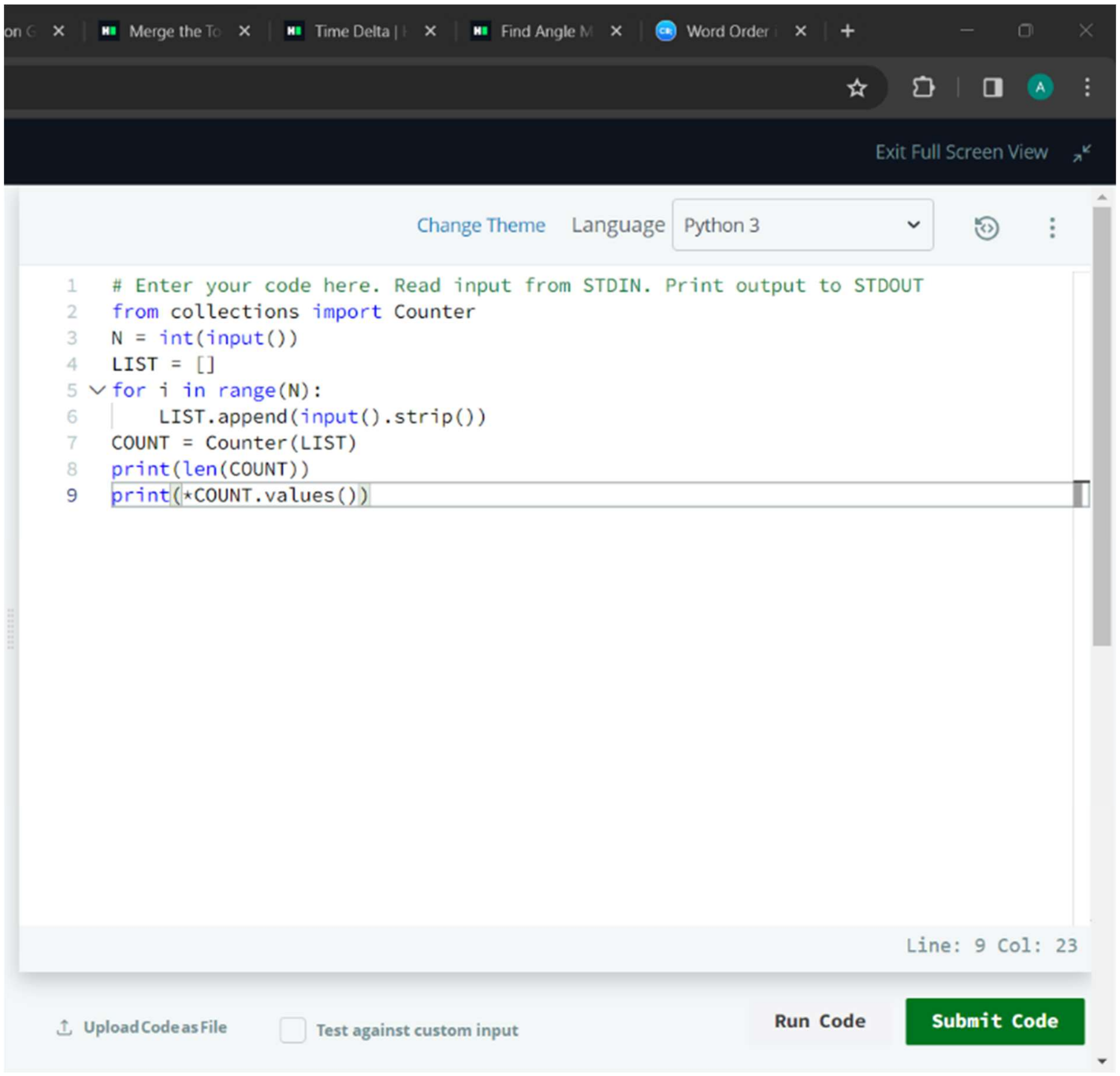
```python
# Enter your code here. Read input from STDIN. Print output to STDOUT
import math
ab=int(input())
bc=int(input())
ca=math.hypot(ab,bc)
mc=ca/2
bca=math.asin(1*ab/ca)
bm=math.sqrt((bc**2+mc**2)-(2*bc*mc*math.cos(bca)))
mbc=math.asin(math.sin(bca)*mc/bm)
print(int(round(math.degrees(mbc),0)),'\u00B0',sep='')
```

Figure 5 Find Angle MBC



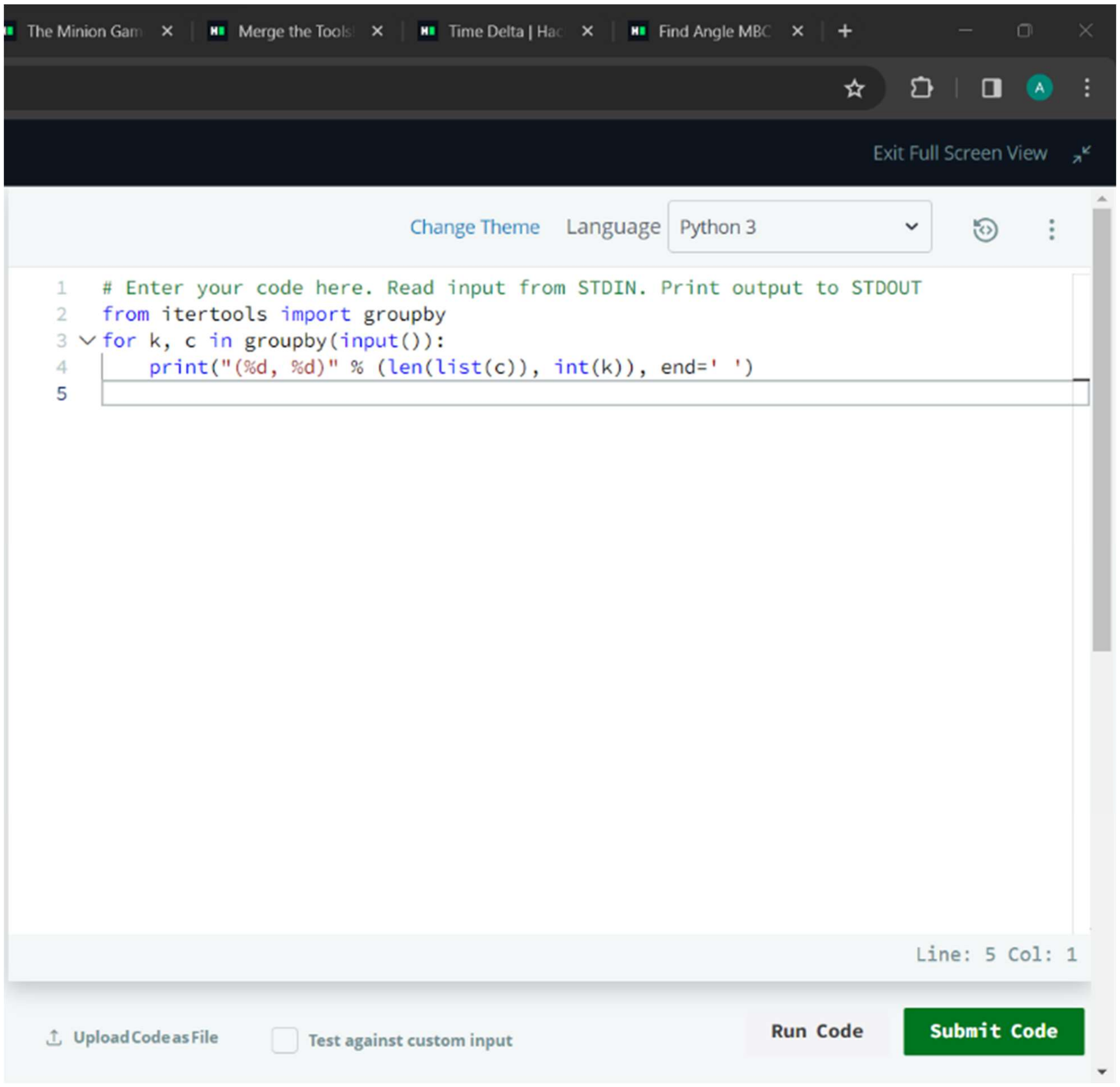
```python
# Enter your code here. Read input from STDIN. Print output to STDOUT
if __name__ == "__main__":
    happiness = 0
    n, m = map(int, input().strip().split(' '))
    arr = list(map(int, input().strip().split(' ')))

    good = set(map(int, input().strip().split(' ')))
    bad = set(map(int, input().strip().split(' ')))

    for i in arr:
        if i in good:
            happiness += 1
        elif i in bad:
            happiness -= 1
    print(happiness)

```

Figure 6 No Idea



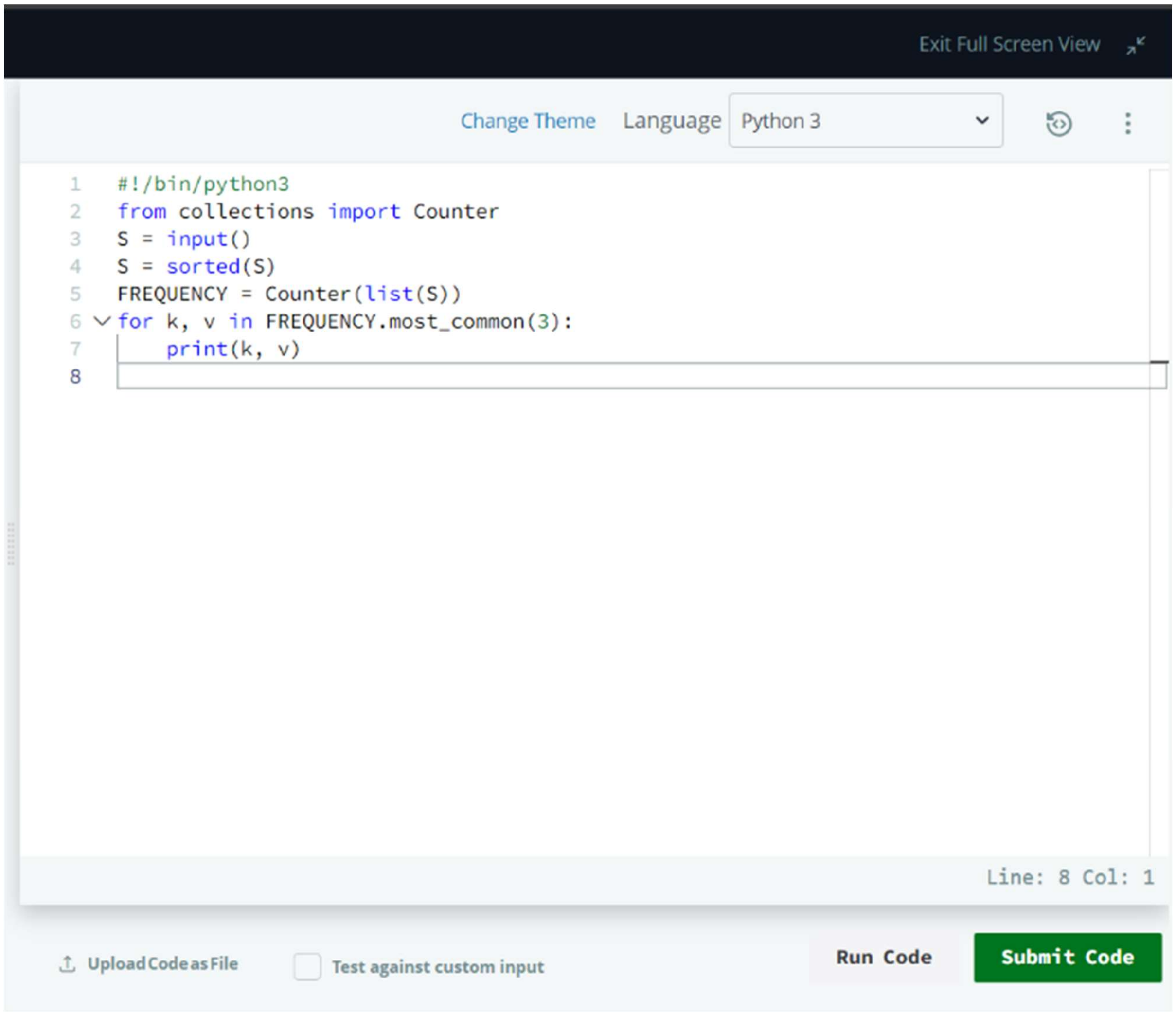
```python
# Enter your code here. Read input from STDIN. Print output to STDOUT
from collections import Counter
N = int(input())
LIST = []
for i in range(N):
    LIST.append(input().strip())
COUNT = Counter(LIST)
print(len(COUNT))
print(*COUNT.values())
```

Figure 7 Word Order



```python
# Enter your code here. Read input from STDIN. Print output to STDOUT
from itertools import groupby
for k, c in groupby(input()):
    print("(%d, %d)" % (len(list(c)), int(k)), end=' ')

```

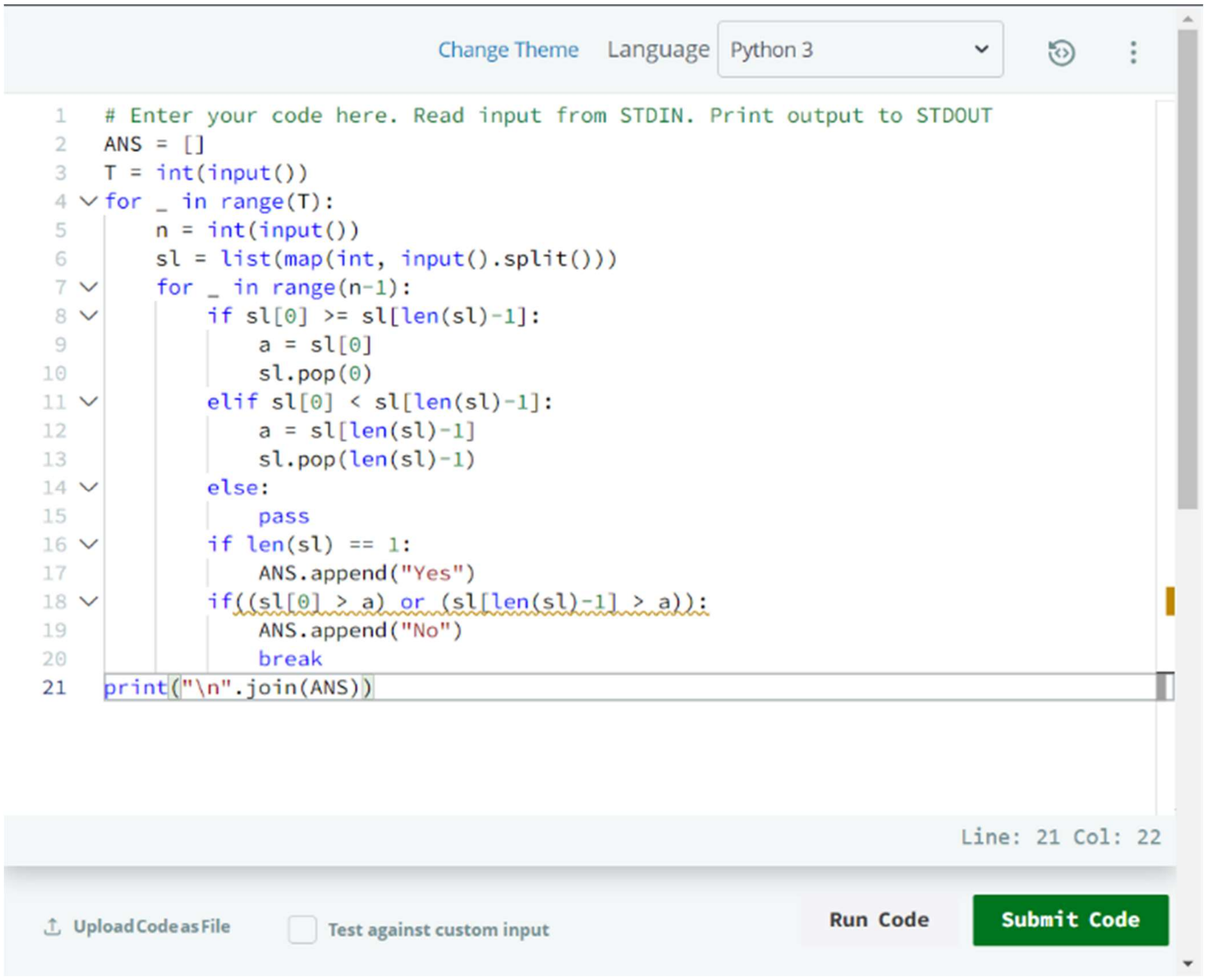
Figure 8 Complete the String



```python
#!/bin/python3
from collections import Counter
S = input()
S = sorted(S)
FREQUENCY = Counter(list(S))
for k, v in FREQUENCY.most_common(3):
    print(k, v)

```

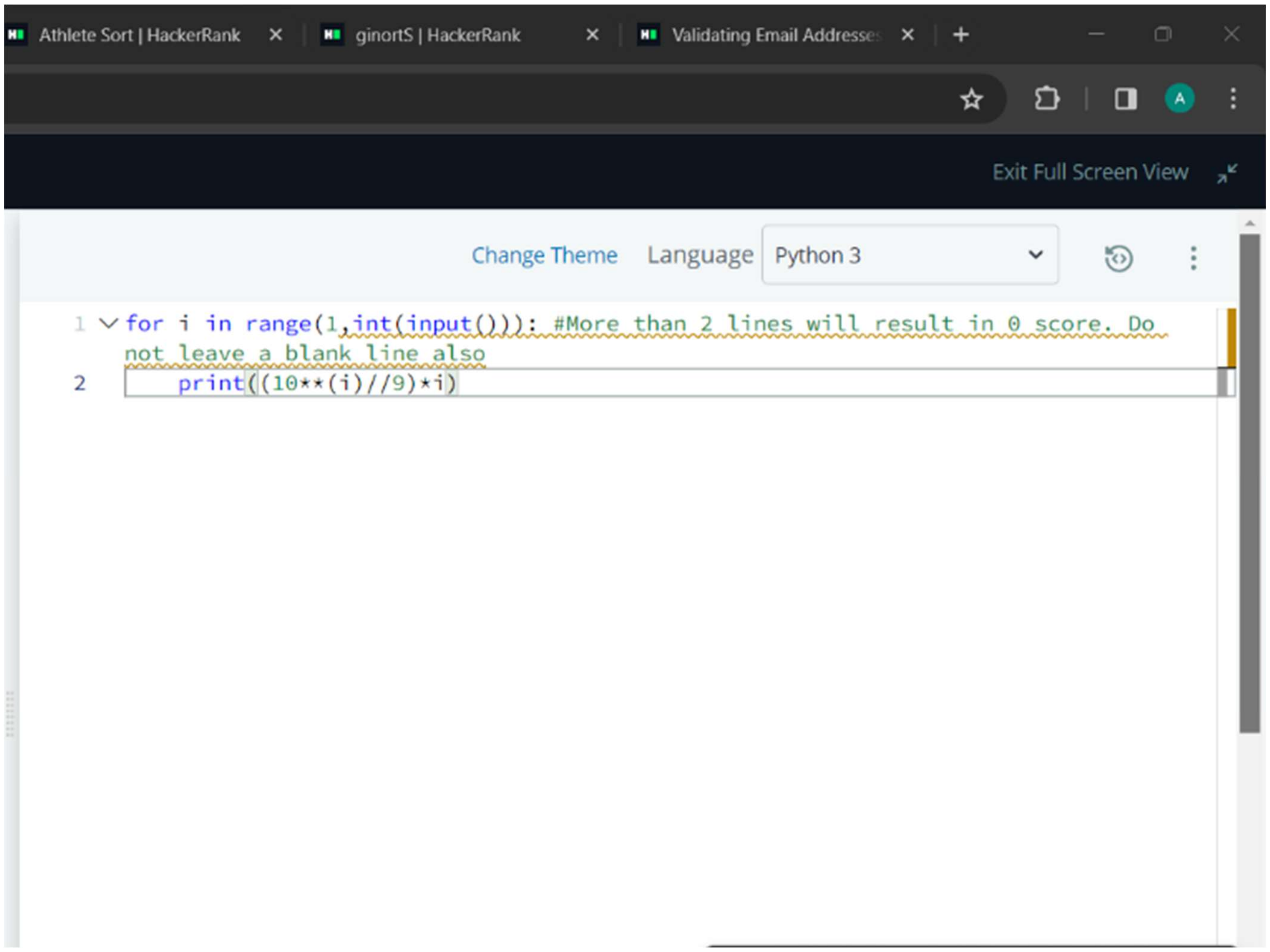
Line: 8 Col: 1

Figure 9 Company Logo



```python
# Enter your code here. Read input from STDIN. Print output to STDOUT
ANS = []
T = int(input())
for _ in range(T):
    n = int(input())
    sl = list(map(int, input().split()))
    for _ in range(n-1):
        if sl[0] >= sl[len(sl)-1]:
            a = sl[0]
            sl.pop(0)
        elif sl[0] < sl[len(sl)-1]:
            a = sl[len(sl)-1]
            sl.pop(len(sl)-1)
        else:
            pass
        if len(sl) == 1:
            ANS.append("Yes")
        if((sl[0] > a) or (sl[len(sl)-1] > a)):
            ANS.append("No")
            break
print("\n".join(ANS))
```

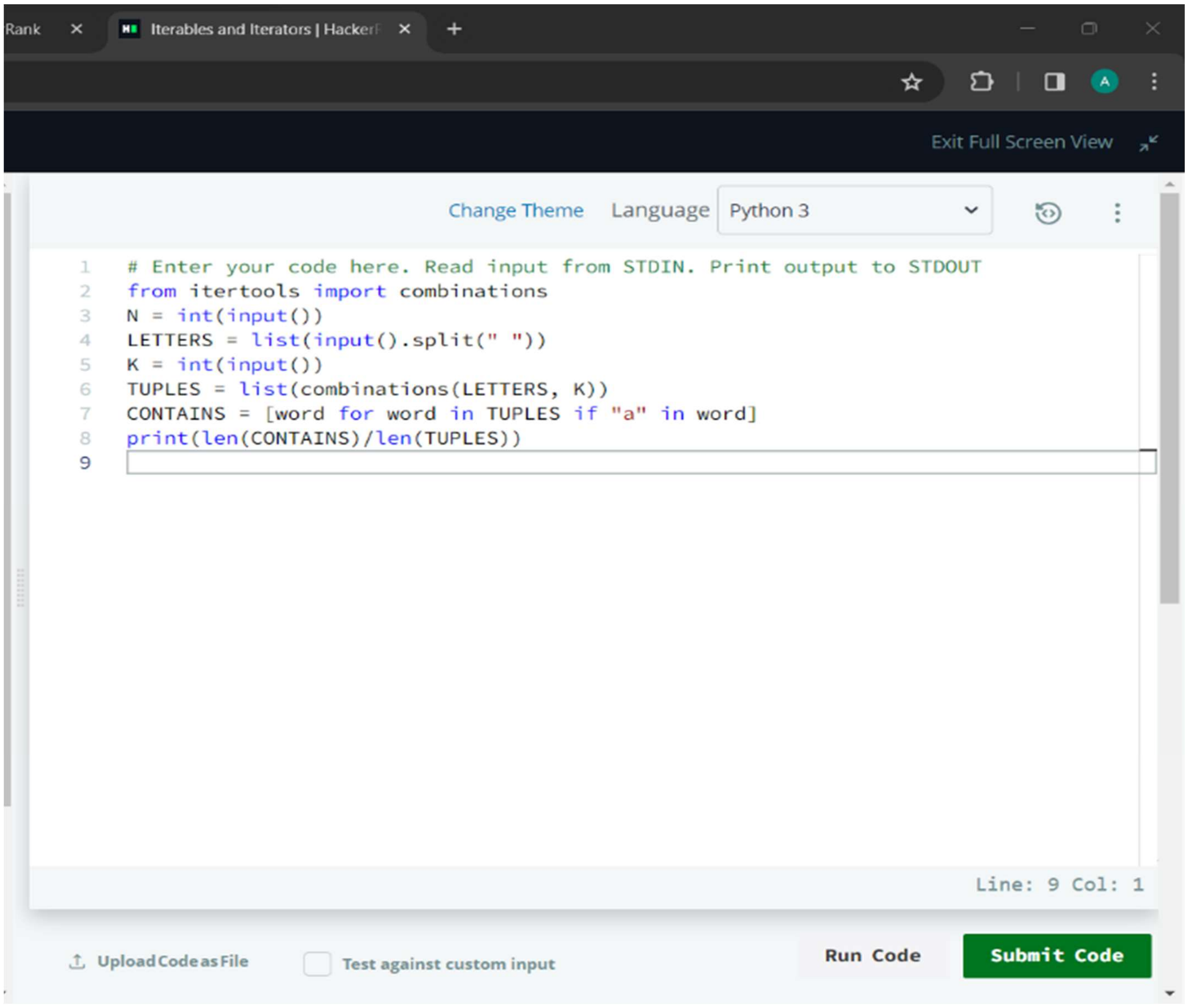
Figure 10 Piling Up



Python 3

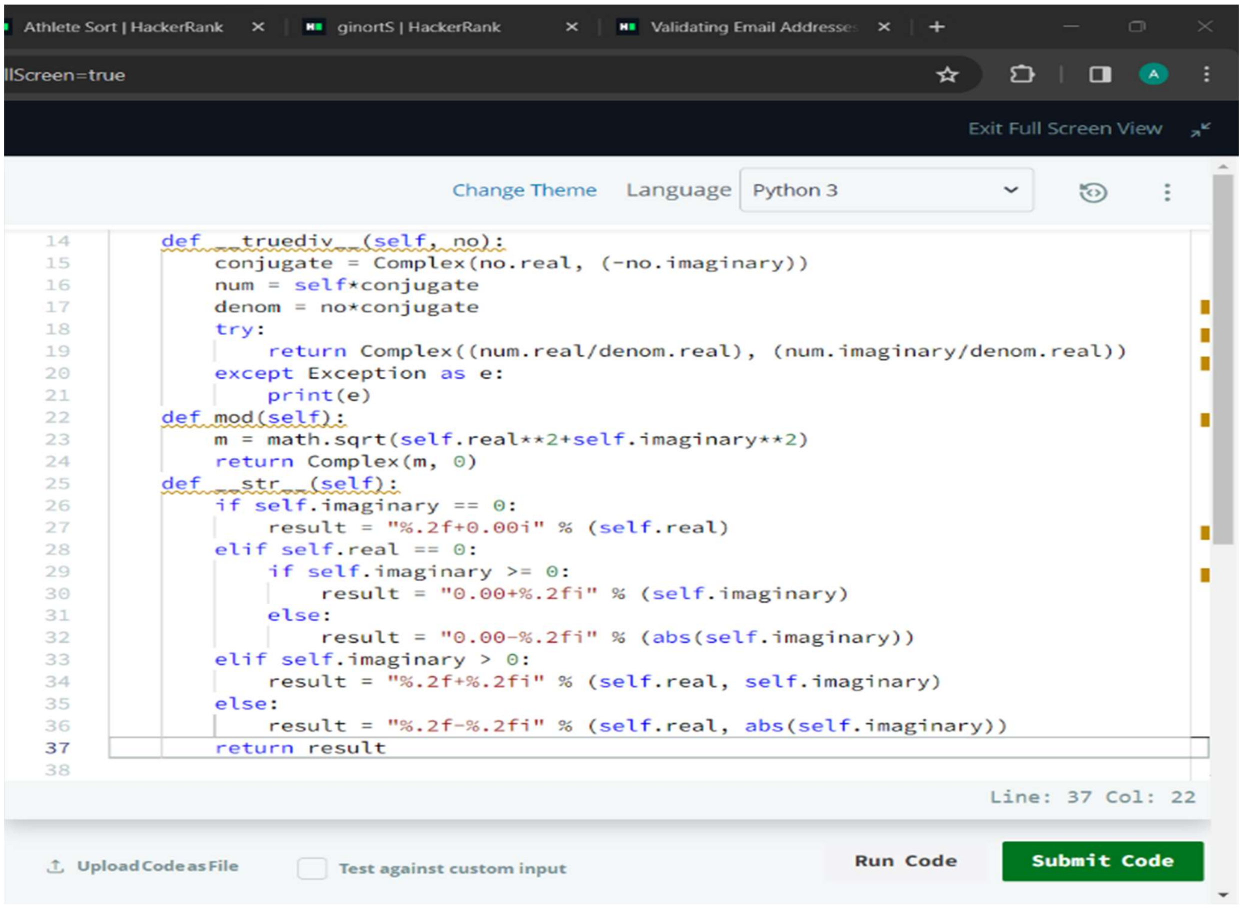
```
for i in range(1,int(input())): #More than 2 lines will result in 0 score. Do not leave a blank line also
    print((10**(i)//9)*i)
```

Figure 11 Traingle Quest



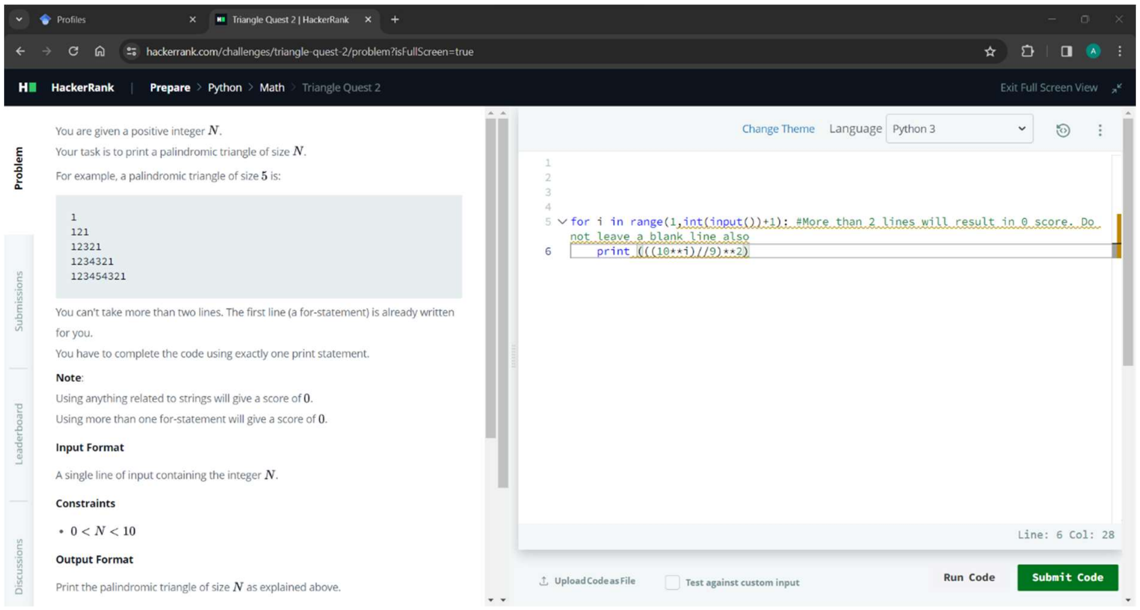
```python
# Enter your code here. Read input from STDIN. Print output to STDOUT
from itertools import combinations
N = int(input())
LETTERS = list(input().split(" "))
K = int(input())
TUPLES = list(combinations(LETTERS, K))
CONTAINS = [word for word in TUPLES if "a" in word]
print(len(CONTAINS)/len(TUPLES))
```

Figure 12 Iterables & Iterators



```python
    def __truediv__(self, no):
        conjugate = Complex(no.real, (-no.imaginary))
        num = self*conjugate
        denom = no*conjugate
        try:
            return Complex((num.real/denom.real), (num.imaginary/denom.real))
        except Exception as e:
            print(e)
    def mod(self):
        m = math.sqrt(self.real**2+self.imaginary**2)
        return Complex(m, 0)
    def __str__(self):
        if self.imaginary == 0:
            result = "%.2f+0.00i" % (self.real)
        elif self.real == 0:
            if self.imaginary >= 0:
                result = "0.00+%.2fi" % (self.imaginary)
            else:
                result = "0.00-%.2fi" % (abs(self.imaginary))
        elif self.imaginary > 0:
            result = "%.2f+%.2fi" % (self.real, self.imaginary)
        else:
            result = "%.2f-%.2fi" % (self.real, abs(self.imaginary))
        return result
```

Figure 13 Classes: Dealing with Complex Numbers



hackerrank.com/challenges/triangle-quest-2/problem?isFullScreen=true

HackerRank | Prepare > Python > Math > Triangle Quest 2

You are given a positive integer N.

Your task is to print a palindromic triangle of size N.

For example, a palindromic triangle of size 5 is:

```
1
121
12321
1234321
123454321
```

You can't take more than two lines. The first line (a for-statement) is already written for you.

You have to complete the code using exactly one print statement.

Note:

Using anything related to strings will give a score of 0.

Using more than one for-statement will give a score of 0.

Input Format

A single line of input containing the integer N.

Constraints

- $0 < N < 10$

Output Format

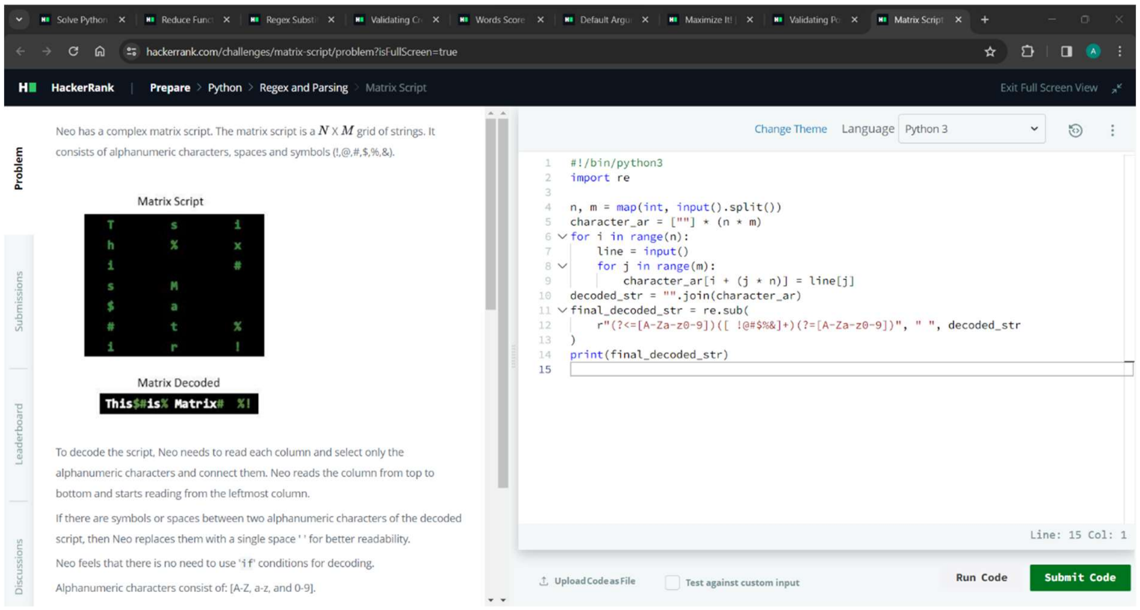
Print the palindromic triangle of size N as explained above.

Language: Python 3

```python




for i in range(1,int(input())+1): #More than 2 lines will result in 0 score. Do not leave a blank line also
    print (((10**i)//9)**2)
```

Line: 6 Col: 28

Figure 14 Traingle Quest 2



hackerrank.com/challenges/matrix-script/problem?isFullScreen=true

HackerRank | Prepare > Python > Regex and Parsing > Matrix Script

Neo has a complex matrix script. The matrix script is a N X M grid of strings. It consists of alphanumeric characters, spaces and symbols (!,@,#,$,%,&).

Matrix Script

```
T s i
h % x
i   #
s M
$ a
# t %
i r !
```

Matrix Decoded

```
This$#is% Matrix# %!
```

To decode the script, Neo needs to read each column and select only the alphanumeric characters and connect them. Neo reads the column from top to bottom and starts reading from the leftmost column.

If there are symbols or spaces between two alphanumeric characters of the decoded script, then Neo replaces them with a single space ' ' for better readability.

Neo feels that there is no need to use 'if' conditions for decoding.

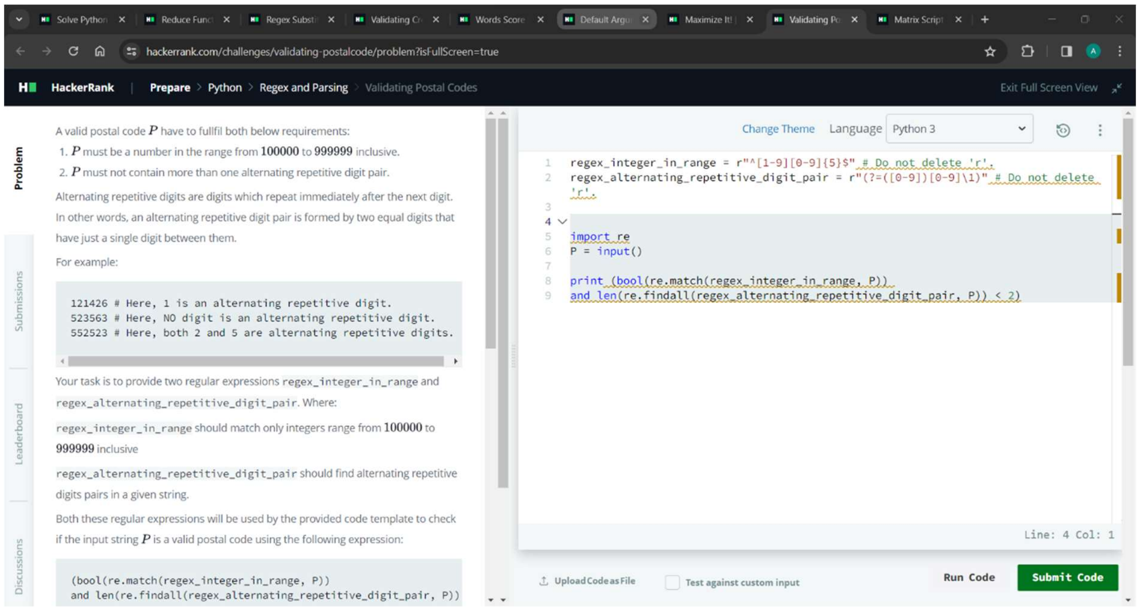
Alphanumeric characters consist of: [A-Z, a-z, and 0-9].

Language: Python 3

```python
#!/bin/python3
import re

n, m = map(int, input().split())
character_ar = [""] * (n * m)
for i in range(n):
    line = input()
    for j in range(m):
        character_ar[i + (j * n)] = line[j]
decoded_str = "".join(character_ar)
final_decoded_str = re.sub(
    r"(?<=[A-Za-z0-9])([ !@#$%&]+)(?=[A-Za-z0-9])", " ", decoded_str
)
print(final_decoded_str)

```

Line: 15 Col: 1

Figure 15 Matrix Script


hackerrank.com/challenges/validating-postalcode/problem?isFullScreen=true
HackerRank
Prepare > Python > Regex and Parsing > Validating Postal Codes
Exit Full Screen View
Problem
Submissions
Leaderboard
Discussions
Change Theme
Language
Python 3
Line: 4 Col: 1
Upload Code as File
Test against custom input
Run Code
Submit Code

A valid postal code P have to fullfil both below requirements:

1. P must be a number in the range from 100000 to 999999 inclusive.
2. P must not contain more than one alternating repetitive digit pair.

Alternating repetitive digits are digits which repeat immediately after the next digit. In other words, an alternating repetitive digit pair is formed by two equal digits that have just a single digit between them.

For example:

```
121426 # Here, 1 is an alternating repetitive digit.
523563 # Here, NO digit is an alternating repetitive digit.
552523 # Here, both 2 and 5 are alternating repetitive digits.
```

Your task is to provide two regular expressions `regex_integer_in_range` and `regex_alternating_repetitive_digit_pair`. Where:

`regex_integer_in_range` should match only integers range from 100000 to 999999 inclusive

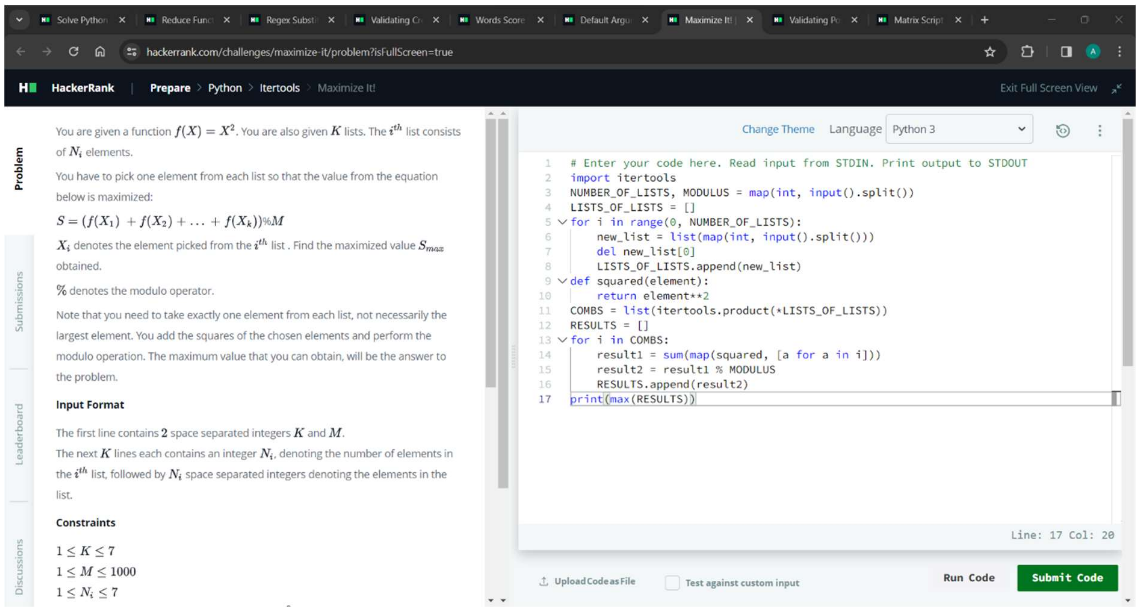
`regex_alternating_repetitive_digit_pair` should find alternating repetitive digits pairs in a given string.

Both these regular expressions will be used by the provided code template to check if the input string P is a valid postal code using the following expression:

```
(bool(re.match(regex_integer_in_range, P))
and len(re.findall(regex_alternating_repetitive_digit_pair, P))
```

```python
regex_integer_in_range = r"^[1-9][0-9]{5}$" # Do not delete 'r'.
regex_alternating_repetitive_digit_pair = r"(?=([0-9])[0-9]\1)" # Do not delete 'r'.


import re
P = input()

print (bool(re.match(regex_integer_in_range, P))
and len(re.findall(regex_alternating_repetitive_digit_pair, P)) < 2)
```

Figure 16 Validating Postal Codes



You are given a function $f(X) = X^2$. You are also given K lists. The i^{th} list consists of N_i elements.

You have to pick one element from each list so that the value from the equation below is maximized:

$S = (f(X_1) + f(X_2) + \ldots + f(X_k))\%M$

X_i denotes the element picked from the i^{th} list . Find the maximized value S_{max} obtained.

$\%$ denotes the modulo operator.

Note that you need to take exactly one element from each list, not necessarily the largest element. You add the squares of the chosen elements and perform the modulo operation. The maximum value that you can obtain, will be the answer to the problem.

Input Format

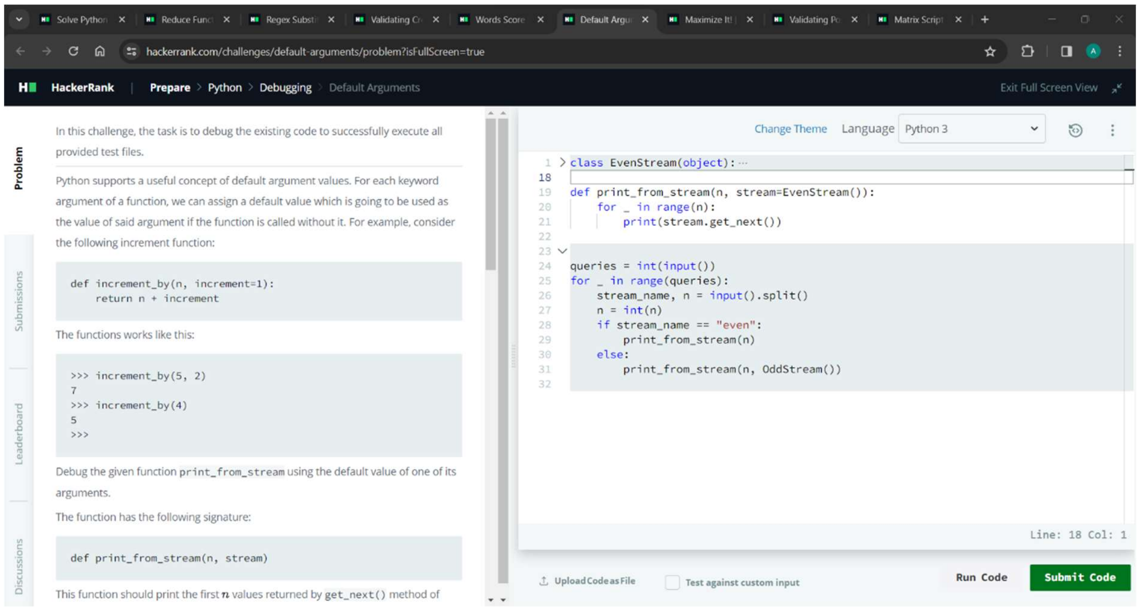
The first line contains 2 space separated integers K and M.
The next K lines each contains an integer N_i, denoting the number of elements in the i^{th} list, followed by N_i space separated integers denoting the elements in the list.

Constraints

$1 \leq K \leq 7$
$1 \leq M \leq 1000$
$1 \leq N_i \leq 7$

```python
# Enter your code here. Read input from STDIN. Print output to STDOUT
import itertools
NUMBER_OF_LISTS, MODULUS = map(int, input().split())
LISTS_OF_LISTS = []
for i in range(0, NUMBER_OF_LISTS):
    new_list = list(map(int, input().split()))
    del new_list[0]
    LISTS_OF_LISTS.append(new_list)
def squared(element):
    return element**2
COMBS = list(itertools.product(*LISTS_OF_LISTS))
RESULTS = []
for i in COMBS:
    result1 = sum(map(squared, [a for a in i]))
    result2 = result1 % MODULUS
    RESULTS.append(result2)
print(max(RESULTS))
```

Figure 17 Maximize It


hackerrank.com/challenges/default-arguments/problem?isFullScreen=true
HackerRank | Prepare > Python > Debugging > Default Arguments
Exit Full Screen View
Problem
Submissions
Leaderboard
Discussions
Change Theme
Language Python 3
Line: 18 Col: 1
Upload Code as File
Test against custom input
Run Code
Submit Code

In this challenge, the task is to debug the existing code to successfully execute all provided test files.

Python supports a useful concept of default argument values. For each keyword argument of a function, we can assign a default value which is going to be used as the value of said argument if the function is called without it. For example, consider the following increment function:

```
def increment_by(n, increment=1):
    return n + increment
```

The functions works like this:

```
>>> increment_by(5, 2)
7
>>> increment_by(4)
5
>>>
```

Debug the given function `print_from_stream` using the default value of one of its arguments.

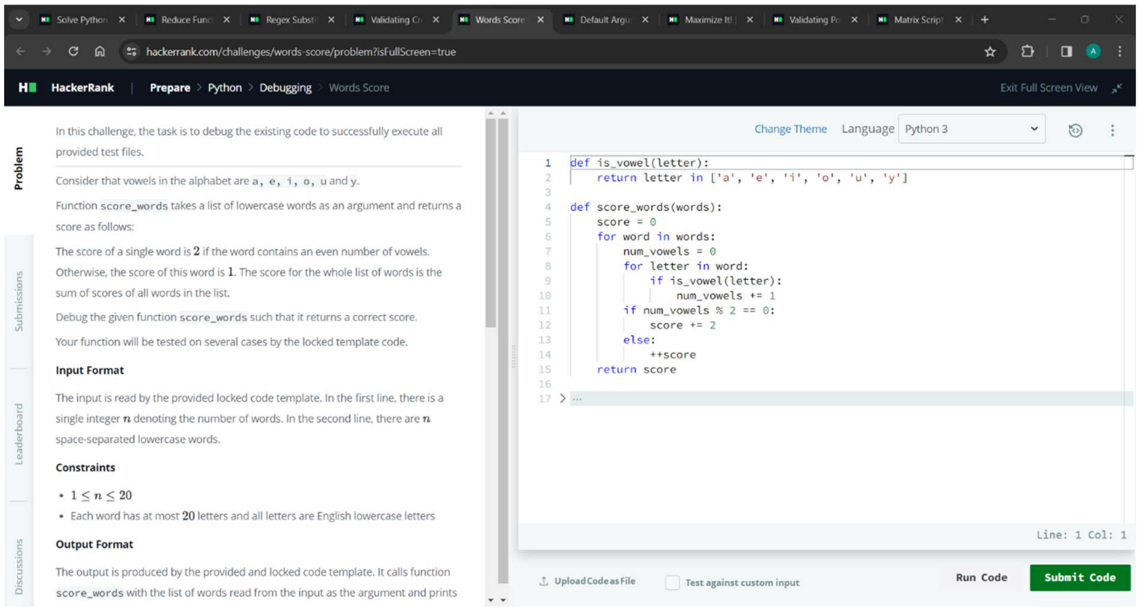
The function has the following signature:

```
def print_from_stream(n, stream)
```

This function should print the first n values returned by `get_next()` method of

```
class EvenStream(object): ···

def print_from_stream(n, stream=EvenStream()):
    for _ in range(n):
        print(stream.get_next())


queries = int(input())
for _ in range(queries):
    stream_name, n = input().split()
    n = int(n)
    if stream_name == "even":
        print_from_stream(n)
    else:
        print_from_stream(n, OddStream())
```

Figure 18 Default Arguments


hackerrank.com/challenges/words-score/problem?isFullScreen=true
HackerRank | Prepare > Python > Debugging > Words Score
Exit Full Screen View
Problem
Submissions
Leaderboard
Discussions
Change Theme
Language Python 3
Line: 1 Col: 1
Upload Code as File
Test against custom input
Run Code
Submit Code

In this challenge, the task is to debug the existing code to successfully execute all provided test files.

Consider that vowels in the alphabet are `a`, `e`, `i`, `o`, `u` and `y`.

Function `score_words` takes a list of lowercase words as an argument and returns a score as follows:

The score of a single word is 2 if the word contains an even number of vowels. Otherwise, the score of this word is 1. The score for the whole list of words is the sum of scores of all words in the list.

Debug the given function `score_words` such that it returns a correct score.

Your function will be tested on several cases by the locked template code.

Input Format

The input is read by the provided locked code template. In the first line, there is a single integer n denoting the number of words. In the second line, there are n space-separated lowercase words.

Constraints

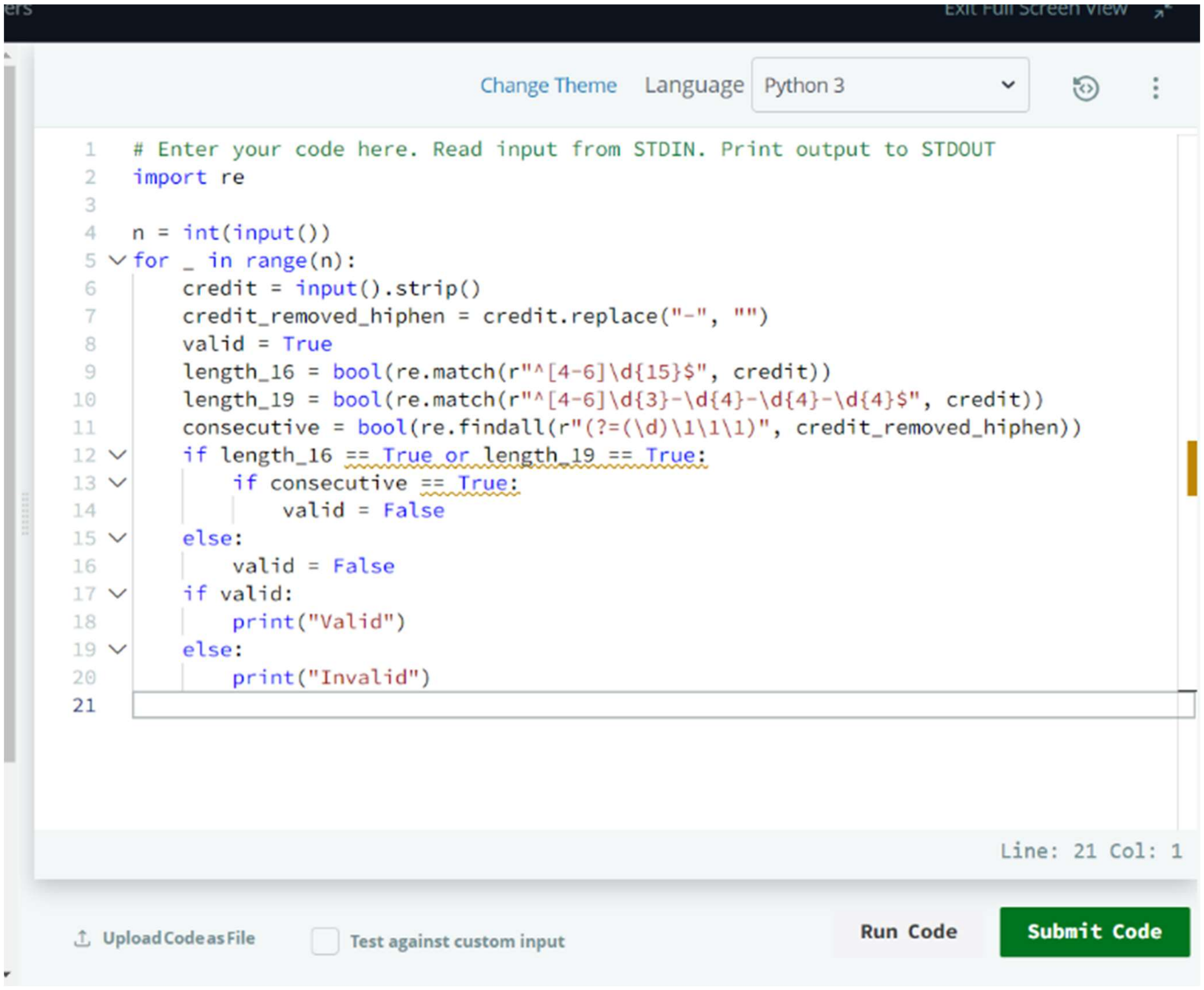
- $1 \leq n \leq 20$
- Each word has at most 20 letters and all letters are English lowercase letters

Output Format

The output is produced by the provided and locked code template. It calls function `score_words` with the list of words read from the input as the argument and prints

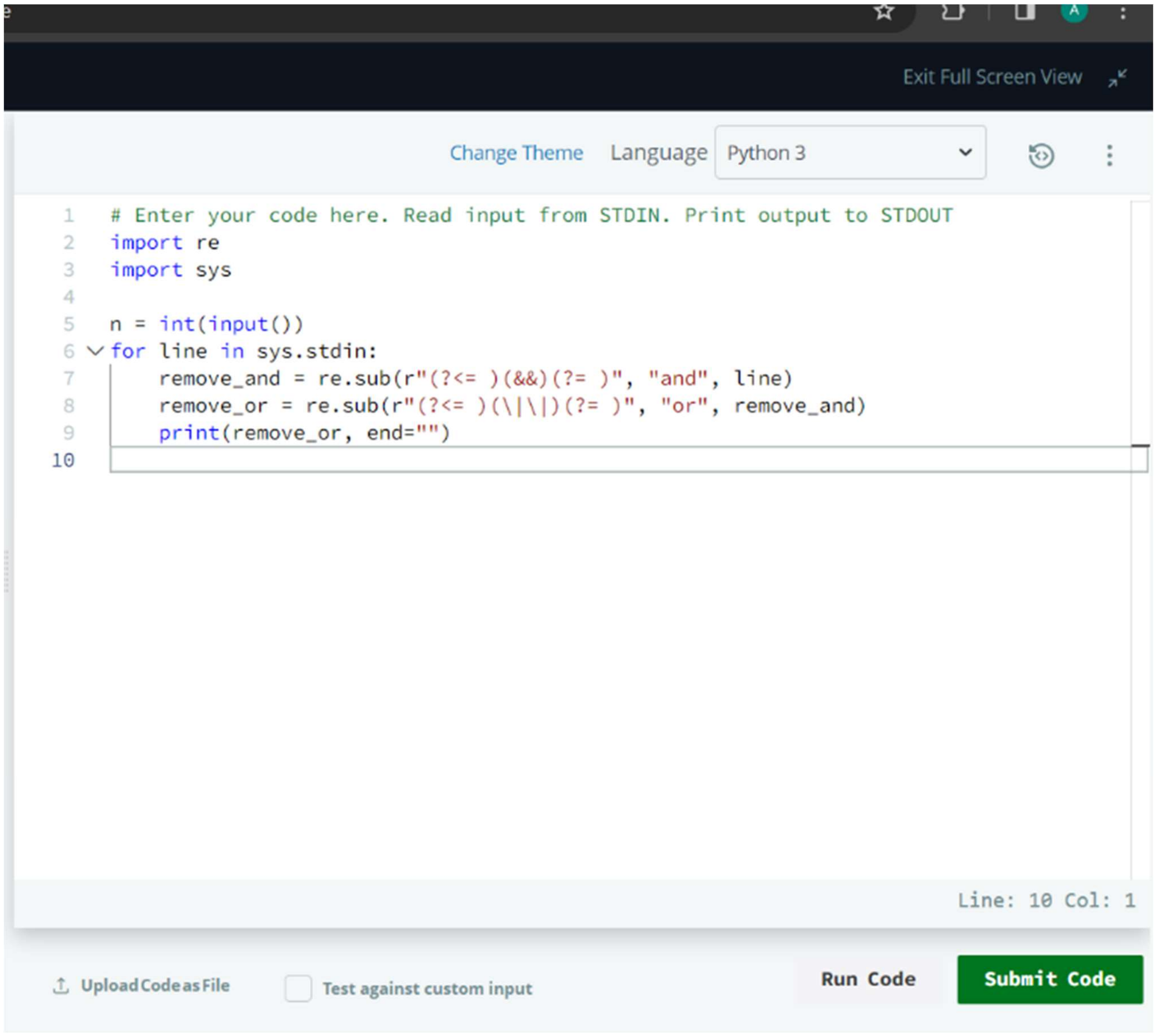
```
def is_vowel(letter):
    return letter in ['a', 'e', 'i', 'o', 'u', 'y']

def score_words(words):
    score = 0
    for word in words:
        num_vowels = 0
        for letter in word:
            if is_vowel(letter):
                num_vowels += 1
        if num_vowels % 2 == 0:
            score += 2
        else:
            ++score
    return score

···
```

Figure 19 Word Score



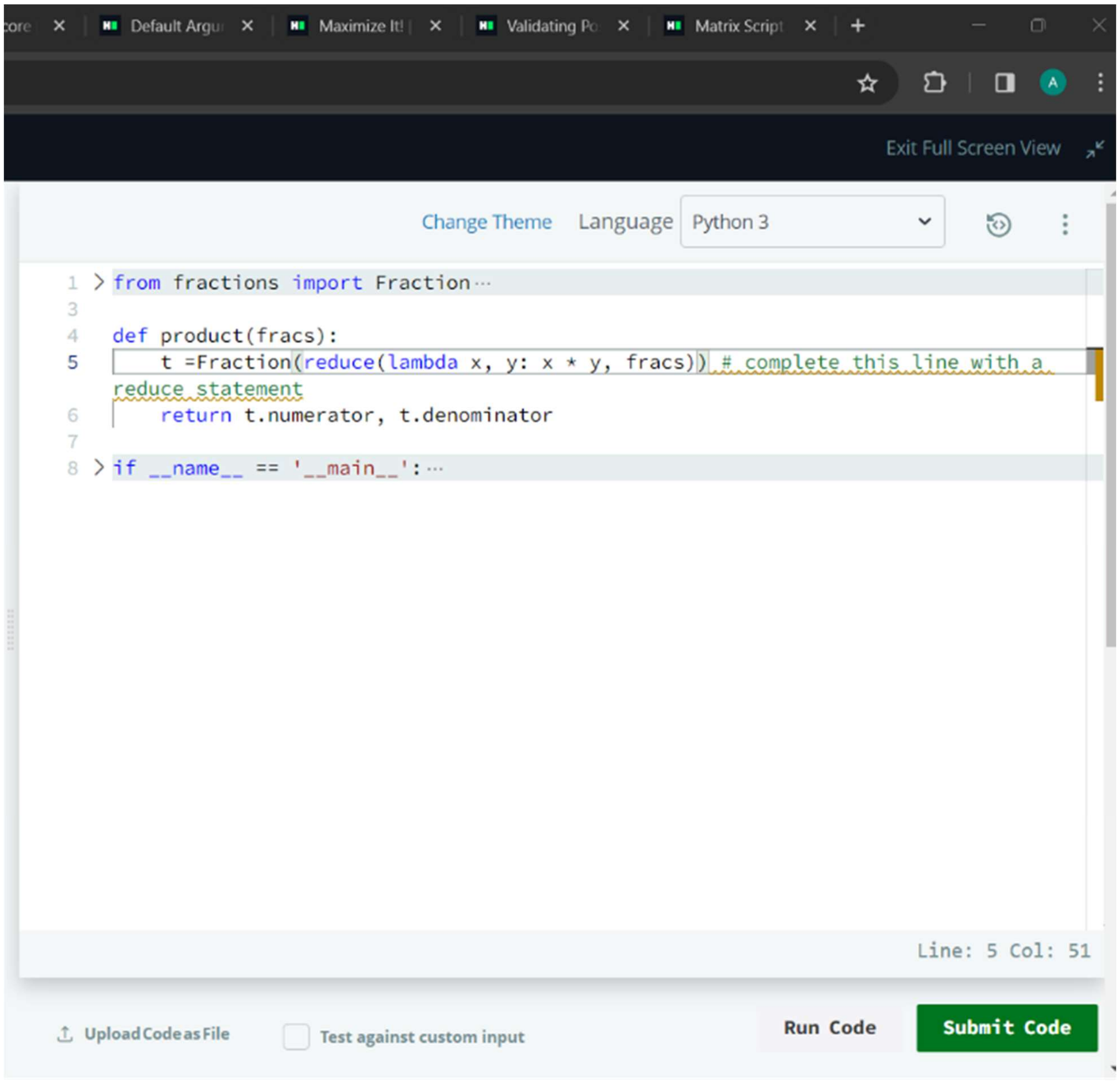
```python
# Enter your code here. Read input from STDIN. Print output to STDOUT
import re

n = int(input())
for _ in range(n):
    credit = input().strip()
    credit_removed_hiphen = credit.replace("-", "")
    valid = True
    length_16 = bool(re.match(r"^[4-6]\d{15}$", credit))
    length_19 = bool(re.match(r"^[4-6]\d{3}-\d{4}-\d{4}-\d{4}$", credit))
    consecutive = bool(re.findall(r"(?=(\d)\1\1\1)", credit_removed_hiphen))
    if length_16 == True or length_19 == True:
        if consecutive == True:
            valid = False
    else:
        valid = False
    if valid:
        print("Valid")
    else:
        print("Invalid")
```

Figure 20 Validating Credit Card Numbers



```python
# Enter your code here. Read input from STDIN. Print output to STDOUT
import re
import sys

n = int(input())
for line in sys.stdin:
    remove_and = re.sub(r"(?<= )(&&)(?= )", "and", line)
    remove_or = re.sub(r"(?<= )(\|\|)(?= )", "or", remove_and)
    print(remove_or, end="")
```

Figure 21 Regex Substitution

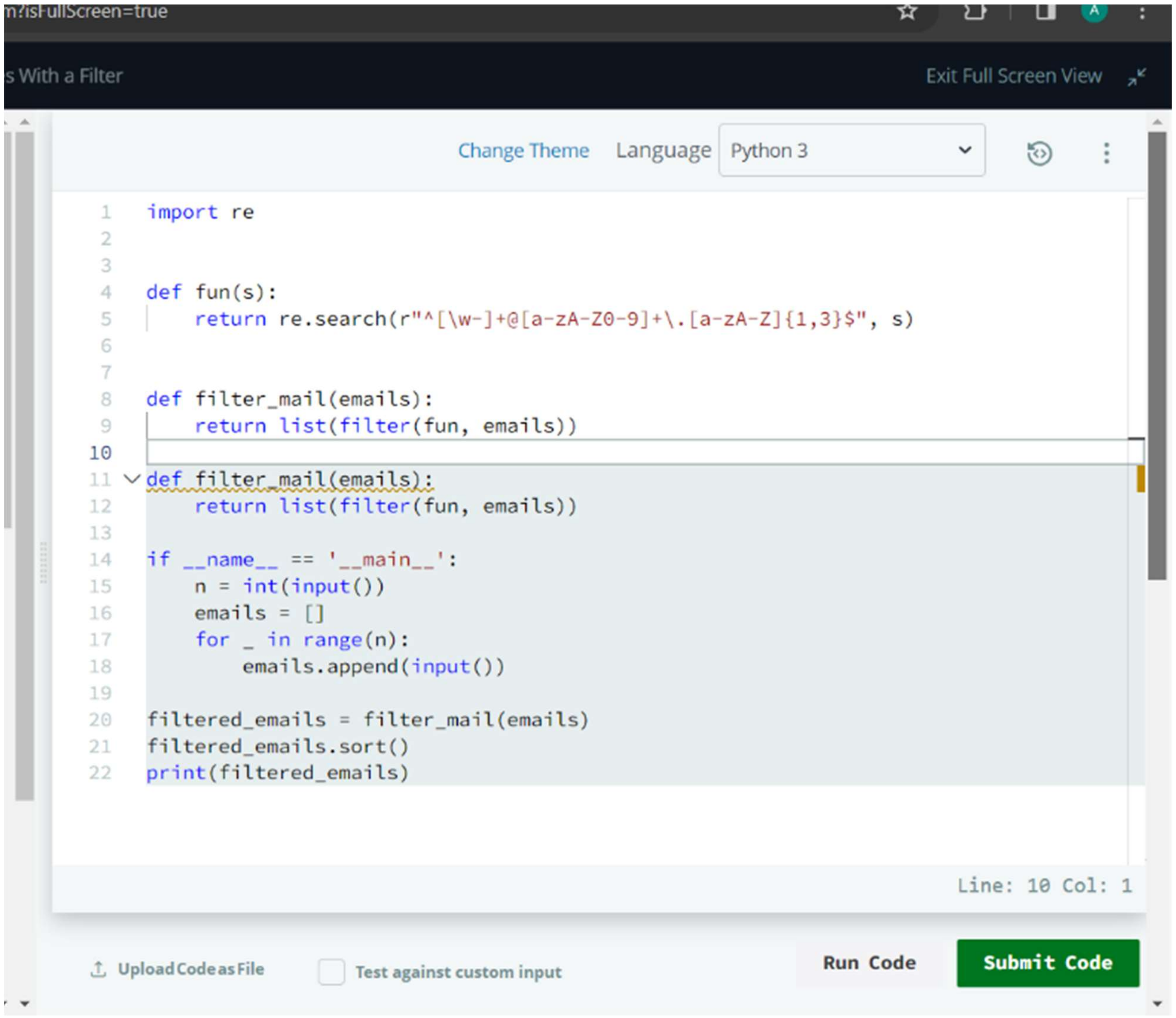


Language: Python 3

```
from fractions import Fraction···

def product(fracs):
    t =Fraction(reduce(lambda x, y: x * y, fracs)) # complete this line with a reduce statement
    return t.numerator, t.denominator

if __name__ == '__main__':···
```

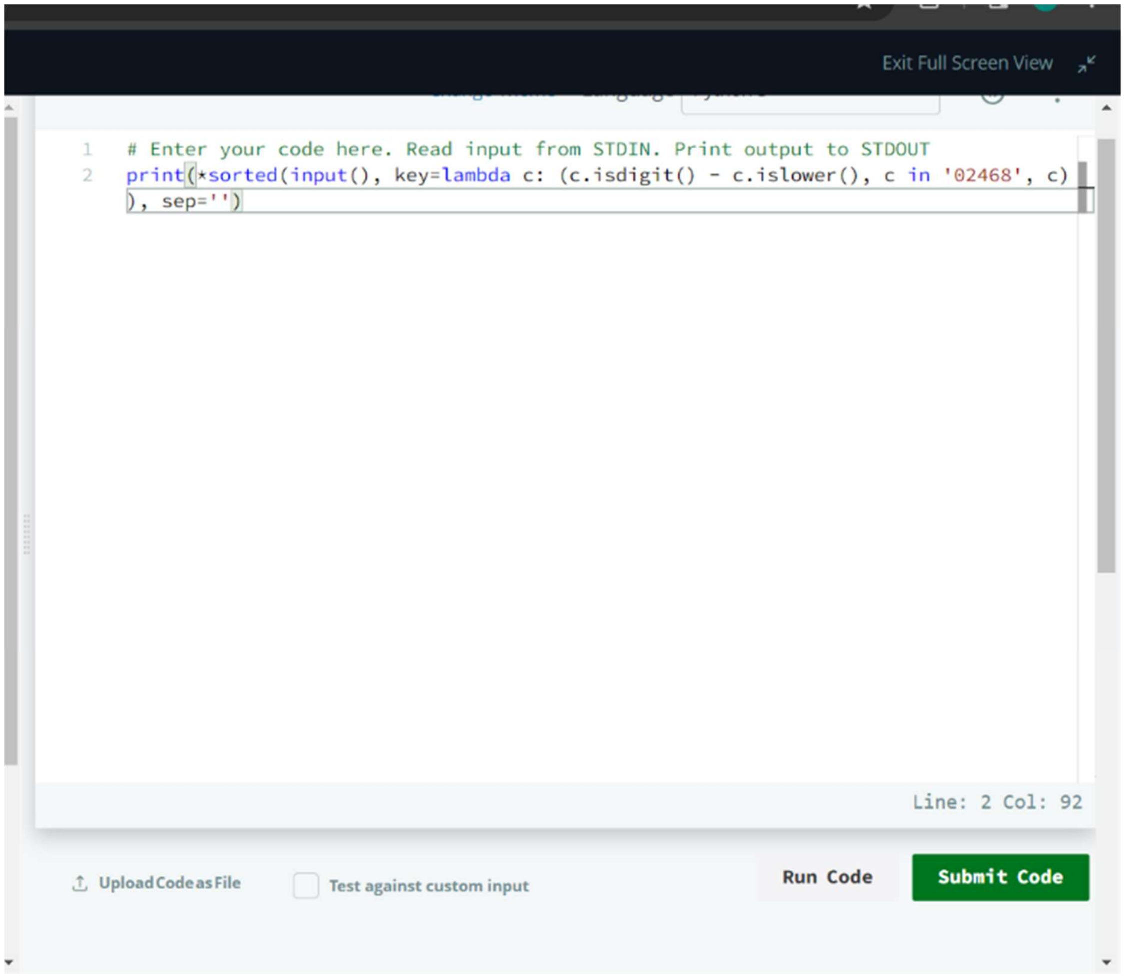
Line: 5 Col: 51

Figure 22 Reduce Function



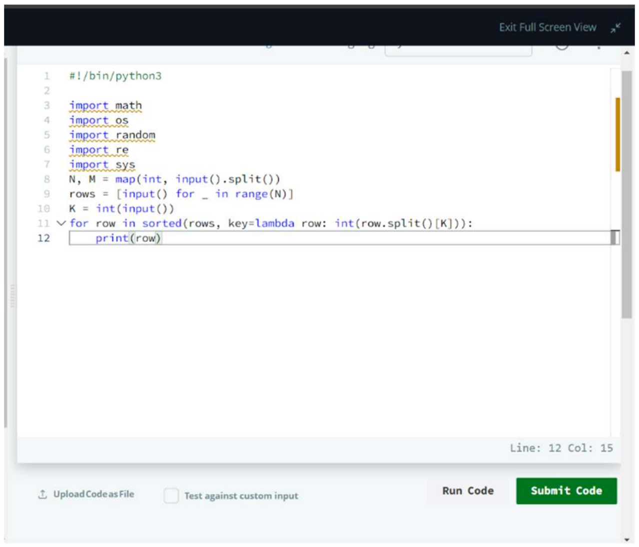
```python
import re


def fun(s):
    return re.search(r"^[\w-]+@[a-zA-Z0-9]+\.[a-zA-Z]{1,3}$", s)


def filter_mail(emails):
    return list(filter(fun, emails))

def filter_mail(emails):
    return list(filter(fun, emails))

if __name__ == '__main__':
    n = int(input())
    emails = []
    for _ in range(n):
        emails.append(input())

filtered_emails = filter_mail(emails)
filtered_emails.sort()
print(filtered_emails)
```

Figure 23 Validating Email Addresses with a Filter



```python
# Enter your code here. Read input from STDIN. Print output to STDOUT
print(*sorted(input(), key=lambda c: (c.isdigit() - c.islower(), c in '02468', c)
), sep='')
```

Figure 24 ginorts



```python
#!/bin/python3

import math
import os
import random
import re
import sys
N, M = map(int, input().split())
rows = [input() for _ in range(N)]
K = int(input())
for row in sorted(rows, key=lambda row: int(row.split()[K])):
    print(row)
```

Figure 25 Athlete Sort