School of Mechanical and Manufacturing Engineering (SMME)
National University of Sciences and Technology (NUST)

ASSIGNMENT # 3

Submitted by: Aina Zulfiqar

Registration #: 450192

Program: MS – Robotics & Intelligent Machine Engineering

Batch: Fall – 2023

Course: CSE 860 – Artificial Intelligence

Submitted to: Dr. Yasar Ayaz

Dated: January 1, 2024

PYTHON CHALLENGES

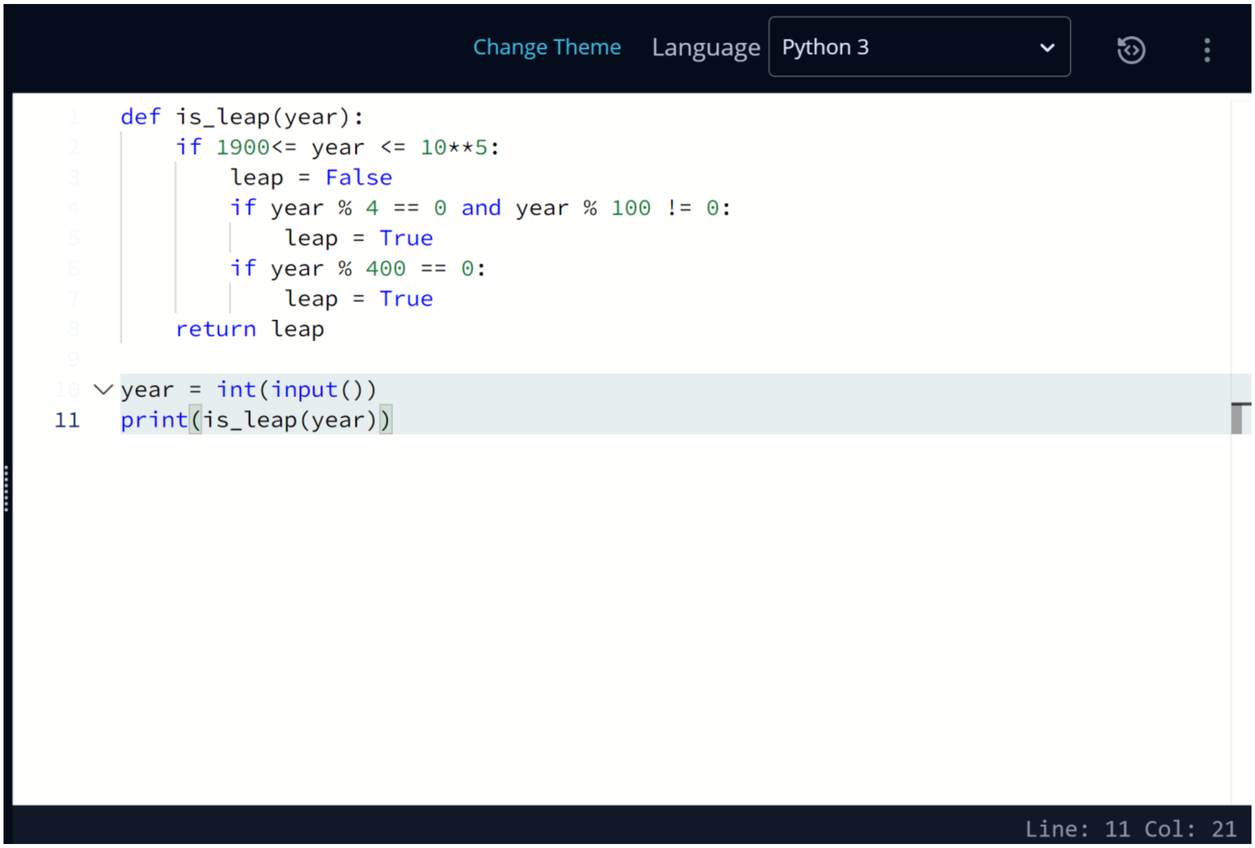
1 Medium Challenges

1.1 Write a Function

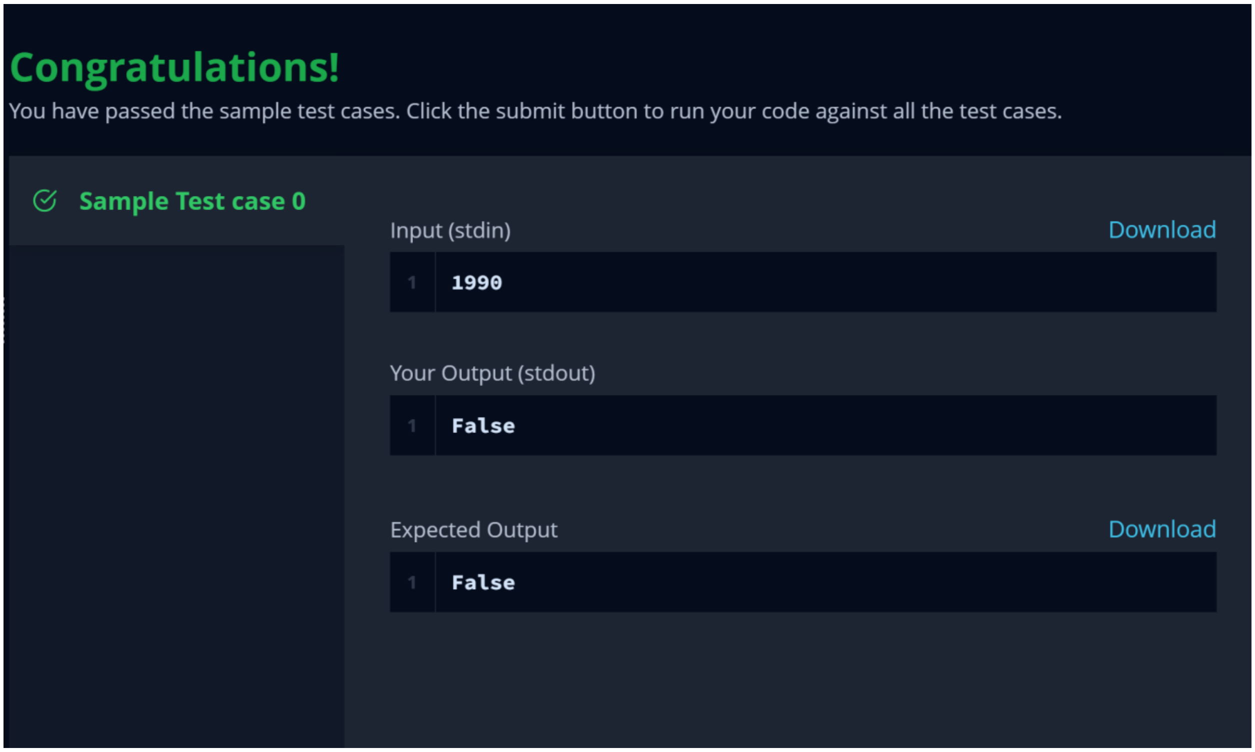
a. Code:

```
def is_leap(year):
  if 1900<= year <= 10**5:
    leap = False
    if year % 4 == 0 and year % 100 != 0:
      leap = True
    if year % 400 == 0:
      leap = True
  return leap

year = int(input())
print(is_leap(year))
```

b. Screenshots:



```
def is_leap(year):
    if 1900<= year <= 10**5:
        leap = False
        if year % 4 == 0 and year % 100 != 0:
            leap = True
        if year % 400 == 0:
            leap = True
    return leap

year = int(input())
print(is_leap(year))
```



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Input (stdin)

```
1990
```

Your Output (stdout)

```
False
```

Expected Output

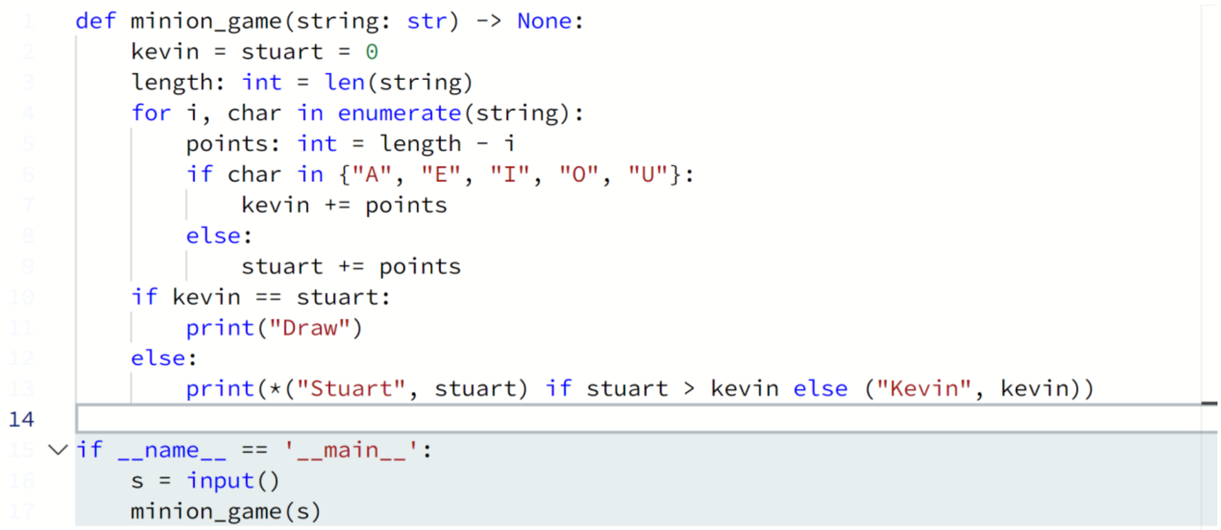
```
False
```

1.2 The Minion Game

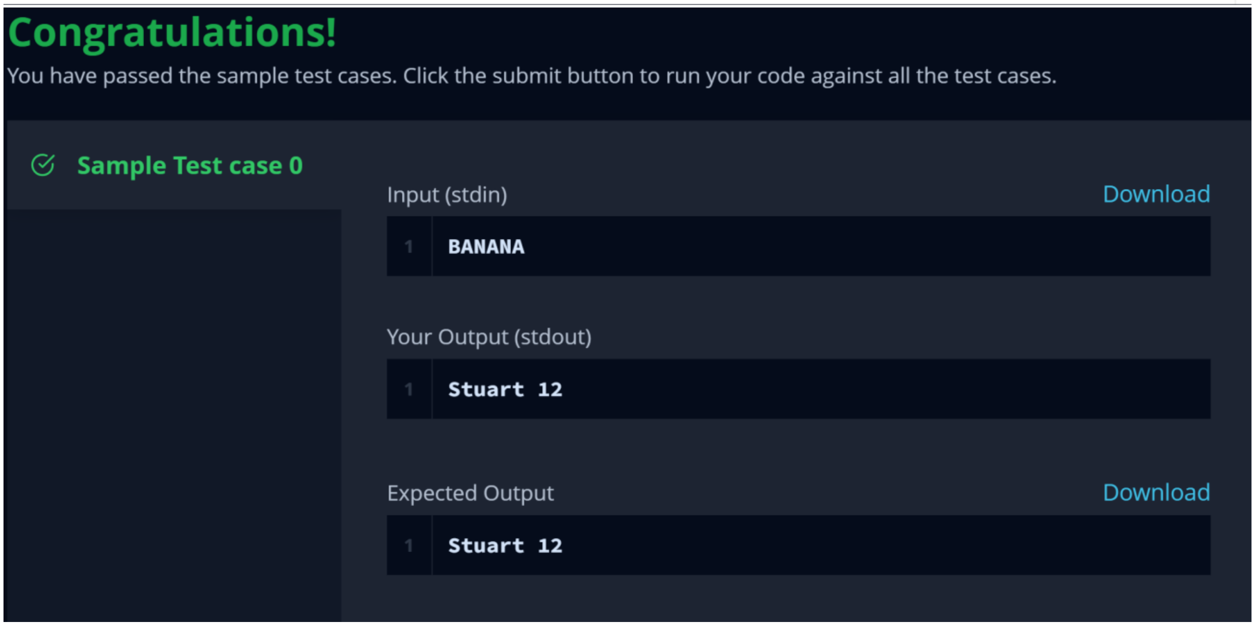
a. Code:

```
def minion_game(string: str) -> None:
    kevin = stuart = 0
    length: int = len(string)
    for i, char in enumerate(string):
        points: int = length - i
        if char in {"A", "E", "I", "O", "U"}:
            kevin += points
        else:
            stuart += points
    if kevin == stuart:
        print("Draw")
    else:
        print(*("Stuart", stuart) if stuart > kevin else ("Kevin", kevin))

if __name__ == '__main__':
    s = input()
    minion_game(s)
```

b. Screenshots:



```python
def minion_game(string: str) -> None:
    kevin = stuart = 0
    length: int = len(string)
    for i, char in enumerate(string):
        points: int = length - i
        if char in {"A", "E", "I", "O", "U"}:
            kevin += points
        else:
            stuart += points
    if kevin == stuart:
        print("Draw")
    else:
        print(*("Stuart", stuart) if stuart > kevin else ("Kevin", kevin))

if __name__ == '__main__':
    s = input()
    minion_game(s)
```



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

✓ **Sample Test case 0**

Input (stdin)

```
BANANA
```

Your Output (stdout)

```
Stuart 12
```

Expected Output

```
Stuart 12
```

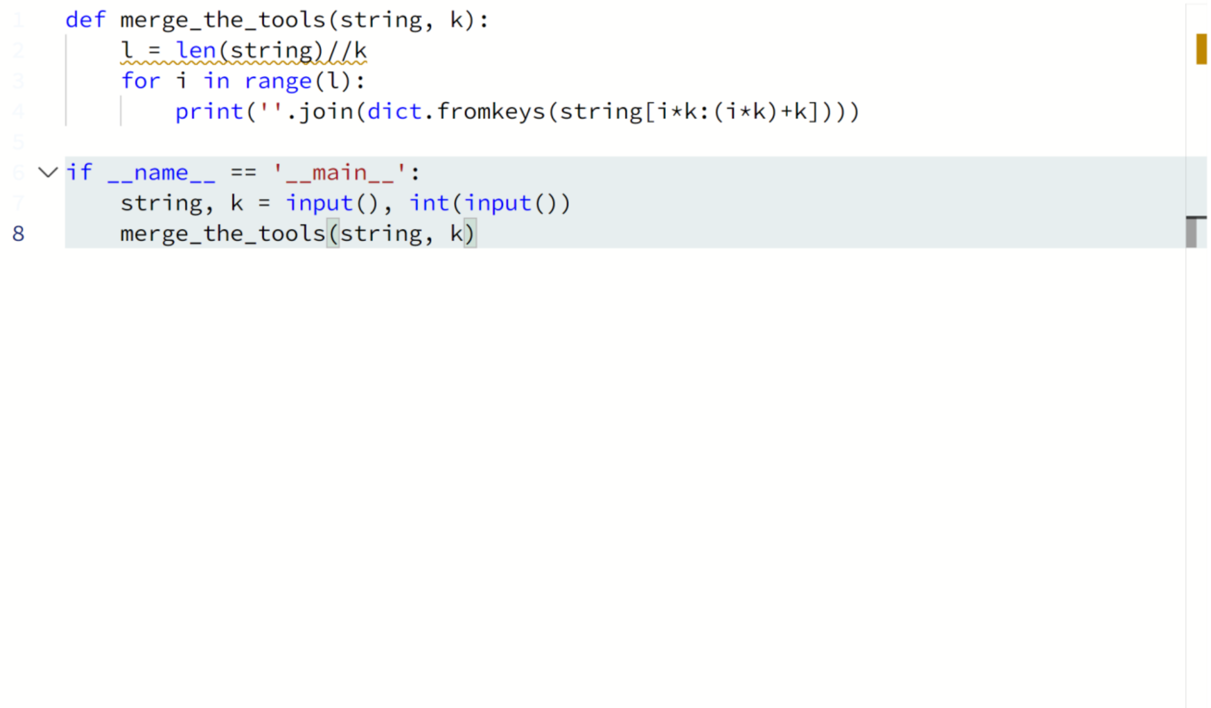
1.3 Merge the Tools!

a. Code:

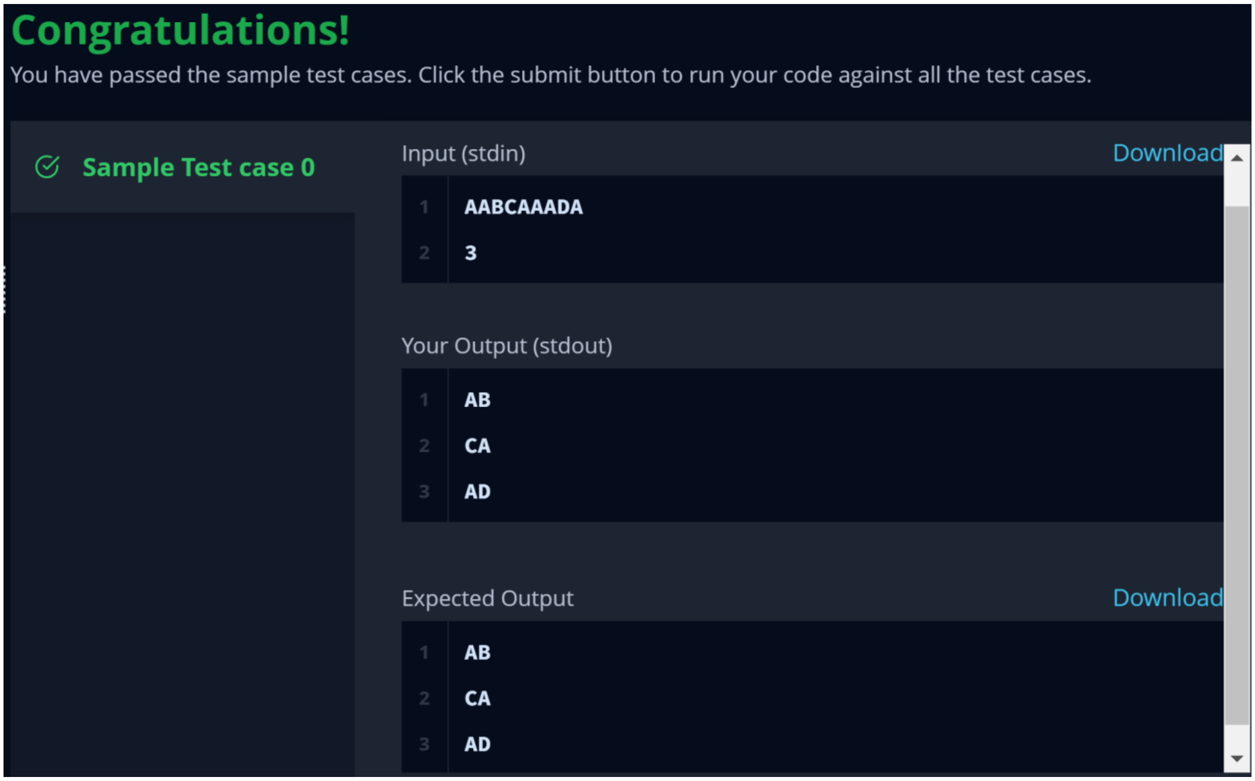
```python
def merge_the_tools(string, k):
    l = len(string)//k
    for i in range(l):
        print(''.join(dict.fromkeys(string[i*k:(i*k)+k])))

if __name__ == '__main__':
```

```
string, k = input(), int(input())
merge_the_tools(string, k)
```

b. Screenshots:



```python
def merge_the_tools(string, k):
    l = len(string)//k
    for i in range(l):
        print(''.join(dict.fromkeys(string[i*k:(i*k)+k])))

if __name__ == '__main__':
    string, k = input(), int(input())
    merge_the_tools(string, k)
```



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Sample Test case 0

Input (stdin)

```
AABCAAADA
3
```

Your Output (stdout)

```
AB
CA
AD
```

Expected Output

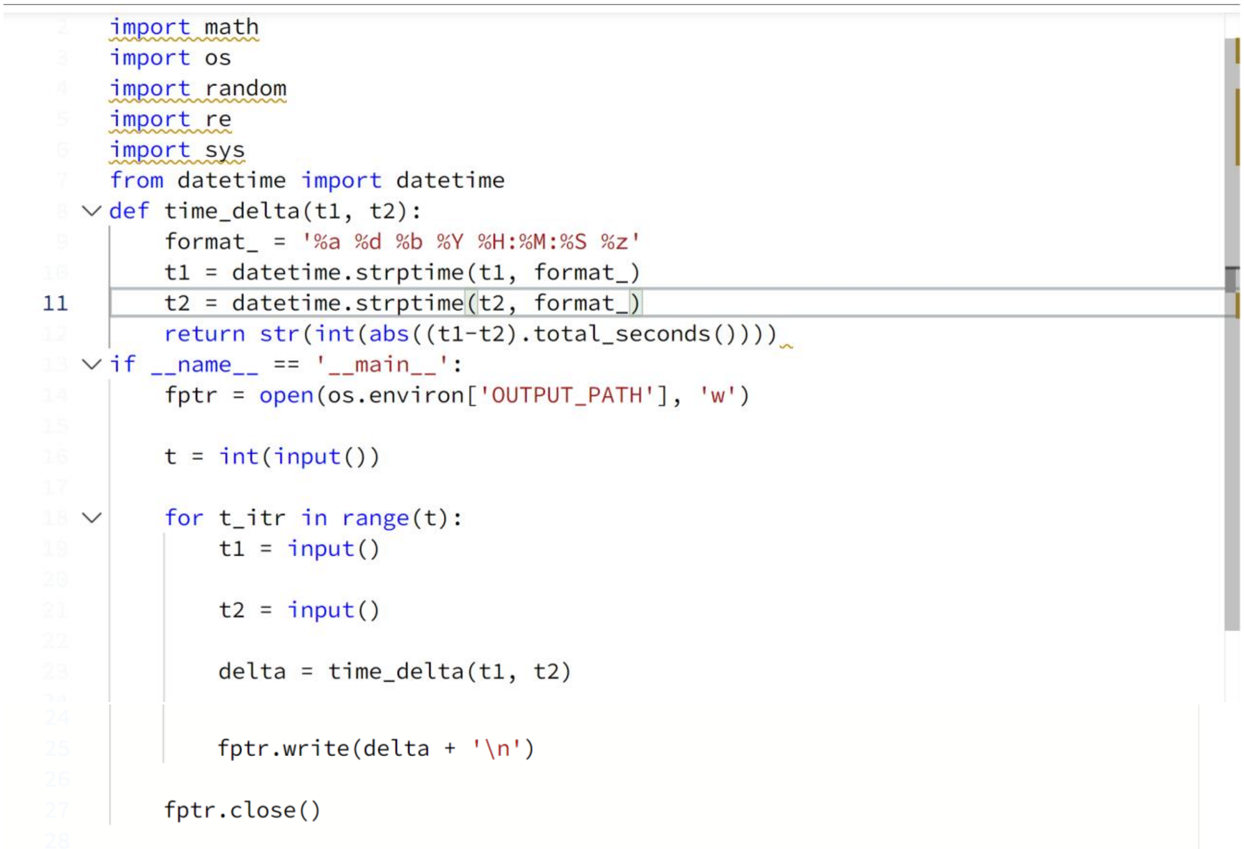
```
AB
CA
AD
```

1.4 Time Delta

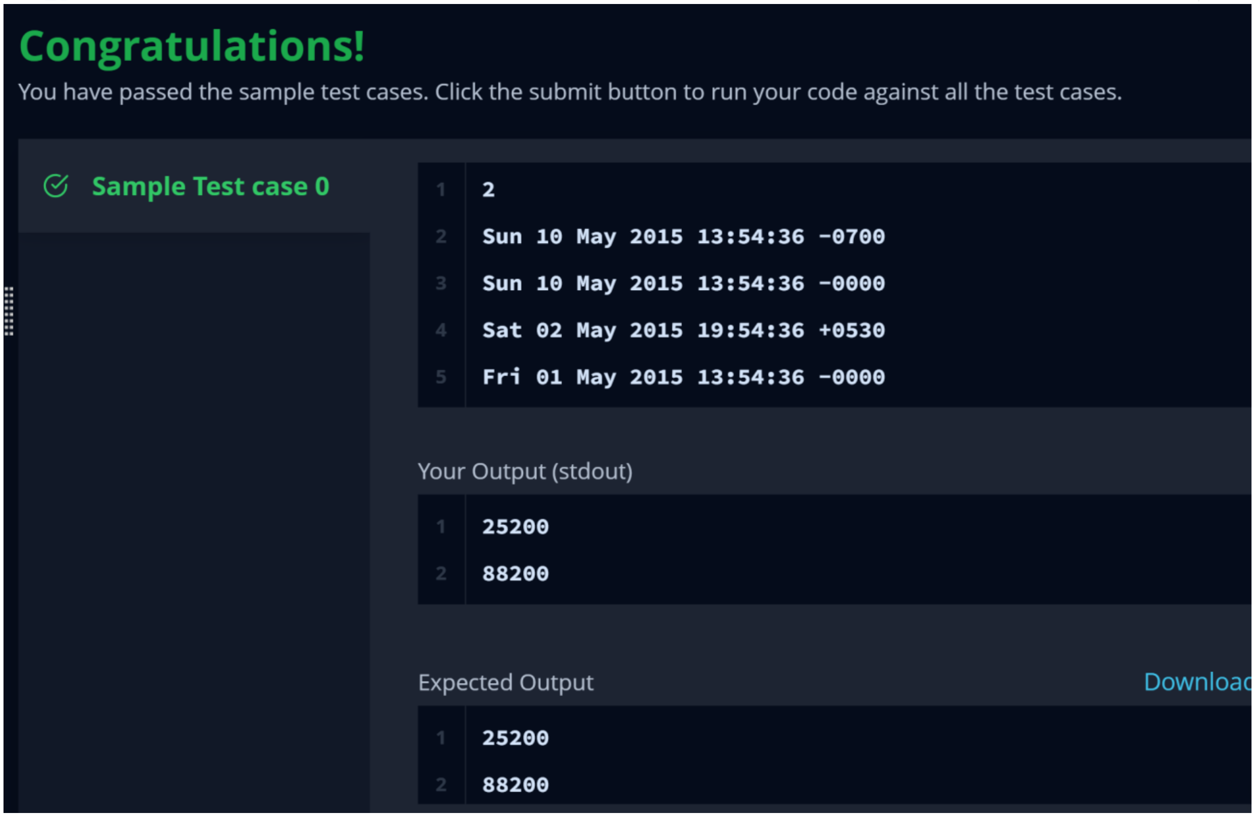
a. Code:

```
import math
import os
import random
import re
import sys
from datetime import datetime
def time_delta(t1, t2):
    format_ = '%a %d %b %Y %H:%M:%S %z'
    t1 = datetime.strptime(t1, format_)
    t2 = datetime.strptime(t2, 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()
```

b. Screenshots:



```python
import math
import os
import random
import re
import sys
from datetime import datetime
def time_delta(t1, t2):
    format_ = '%a %d %b %Y %H:%M:%S %z'
    t1 = datetime.strptime(t1, format_)
    t2 = datetime.strptime(t2, 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()
```



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Sample Test case 0

```
2
Sun 10 May 2015 13:54:36 -0700
Sun 10 May 2015 13:54:36 -0000
Sat 02 May 2015 19:54:36 +0530
Fri 01 May 2015 13:54:36 -0000
```

Your Output (stdout)

```
25200
88200
```

Expected Output

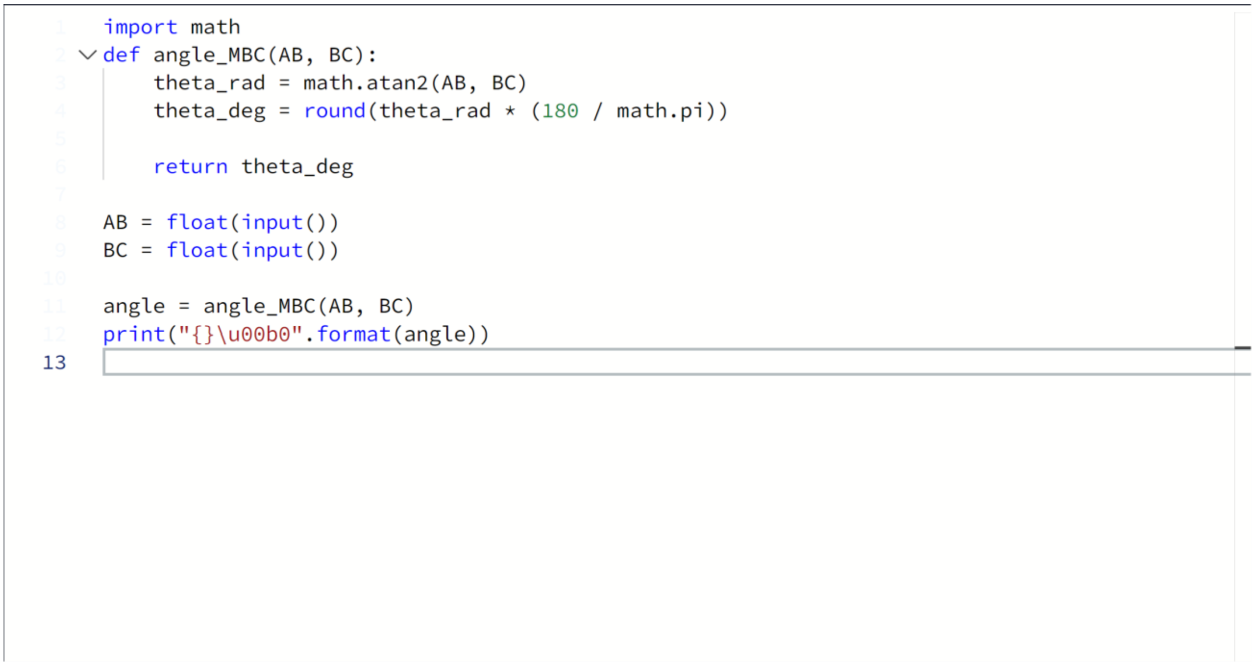
```
25200
88200
```

1.5 Find Angle MBC

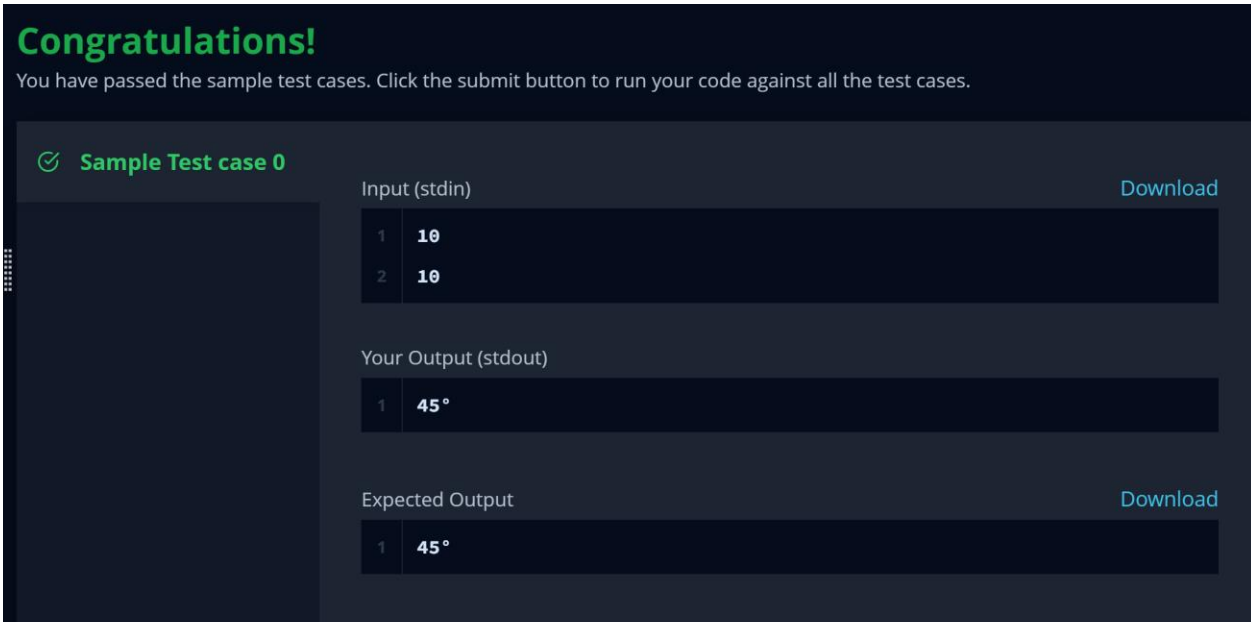
a. Code:

```
import math
def angle_MBC(AB, BC):
    theta_rad = math.atan2(AB, BC)
    theta_deg = round(theta_rad * (180 / math.pi))

    return theta_deg

AB = float(input())
BC = float(input())

angle = angle_MBC(AB, BC)
print("{}\u00b0".format(angle))
```

b. Screenshots:



```
import math
def angle_MBC(AB, BC):
    theta_rad = math.atan2(AB, BC)
    theta_deg = round(theta_rad * (180 / math.pi))

    return theta_deg

AB = float(input())
BC = float(input())

angle = angle_MBC(AB, BC)
print("{}\u00b0".format(angle))

```



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Sample Test case 0

Input (stdin)

```
10
10
```

Your Output (stdout)

```
45°
```

Expected Output

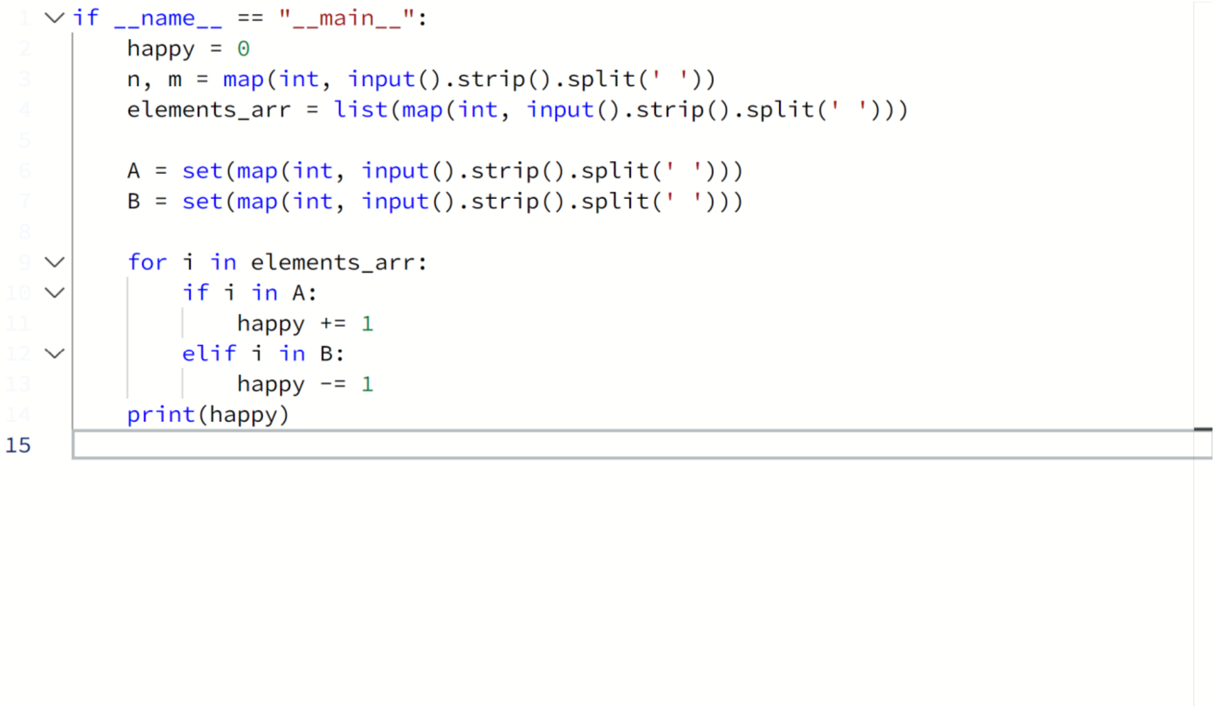
```
45°
```

1.6 No Idea!

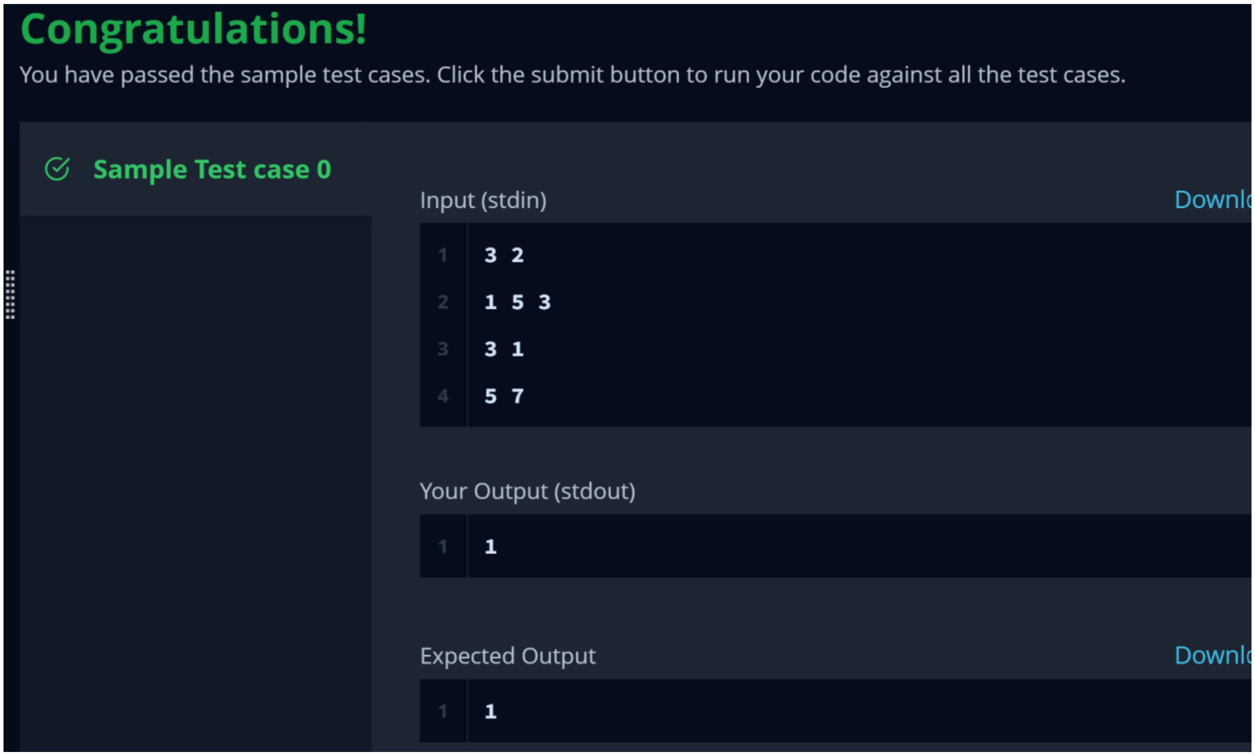
a. Code:

```
if __name__ == "__main__":
    happy = 0
    n, m = map(int, input().strip().split(' '))
    elements_arr = list(map(int, input().strip().split(' ')))

    A = set(map(int, input().strip().split(' ')))
    B = set(map(int, input().strip().split(' ')))

    for i in elements_arr:
        if i in A:
            happy += 1
        elif i in B:
            happy -= 1
    print(happy)
```

b. Screenshots:



```python
if __name__ == "__main__":
    happy = 0
    n, m = map(int, input().strip().split(' '))
    elements_arr = list(map(int, input().strip().split(' ')))

    A = set(map(int, input().strip().split(' ')))
    B = set(map(int, input().strip().split(' ')))

    for i in elements_arr:
        if i in A:
            happy += 1
        elif i in B:
            happy -= 1
    print(happy)
```



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Sample Test case 0

Input (stdin)

```
3 2
1 5 3
3 1
5 7
```

Your Output (stdout)

```
1
```

Expected Output

```
1
```

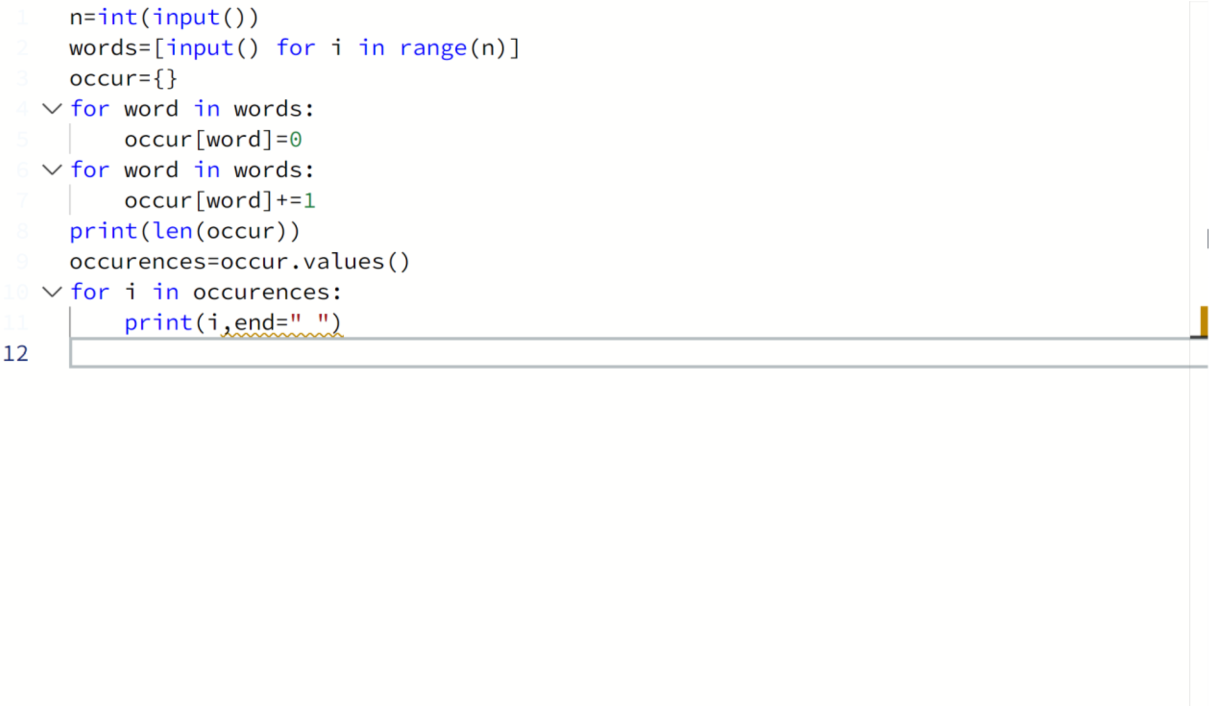
1.7 Word Order

a. Code:

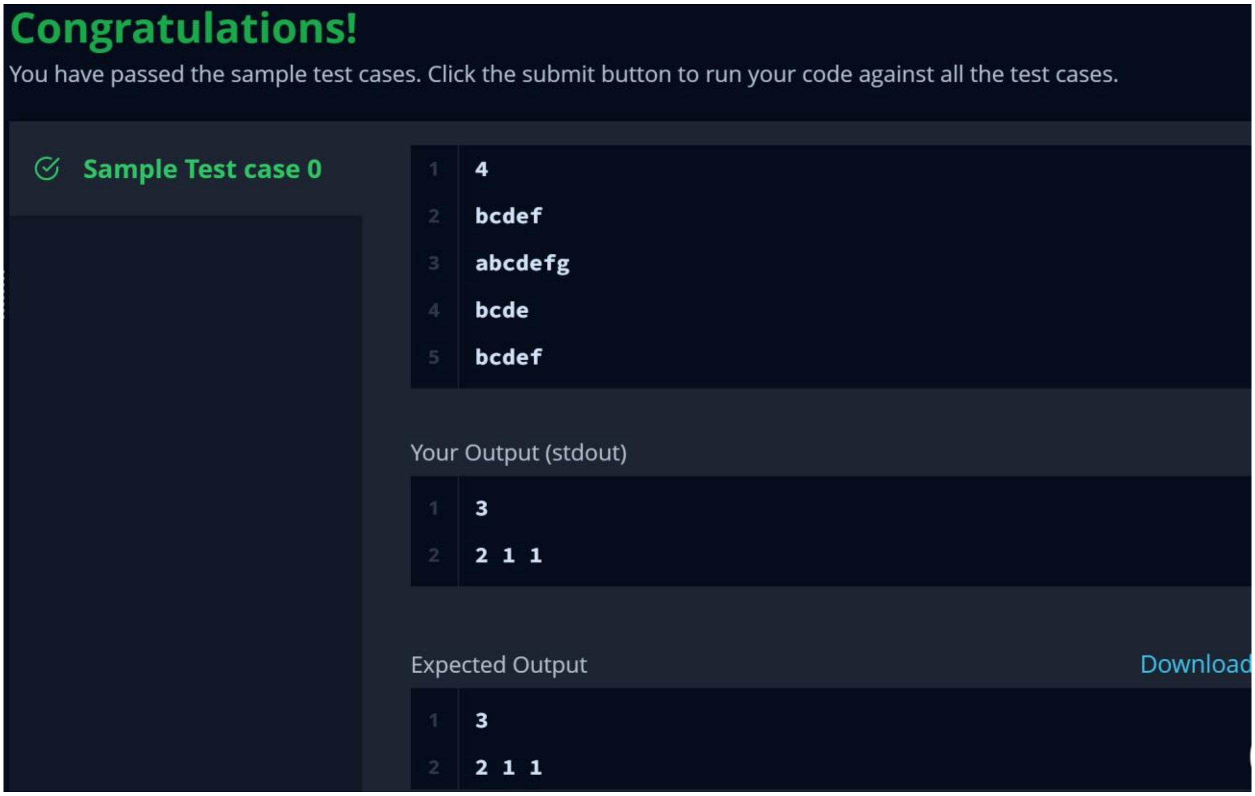
```
n=int(input())
words=[input() for i in range(n)]
occur={}
```

```
for word in words:
    occur[word]=0
for word in words:
    occur[word]+=1
print(len(occur))
occurences=occur.values()
for i in occurences:
    print(i,end=" ")
```

b. Screenshots:



```
n=int(input())
words=[input() for i in range(n)]
occur={}
for word in words:
    occur[word]=0
for word in words:
    occur[word]+=1
print(len(occur))
occurences=occur.values()
for i in occurences:
    print(i,end=" ")

```



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

```
4
bcdef
abcdefg
bcde
bcdef
```

Your Output (stdout)

```
3
2 1 1
```

Expected Output

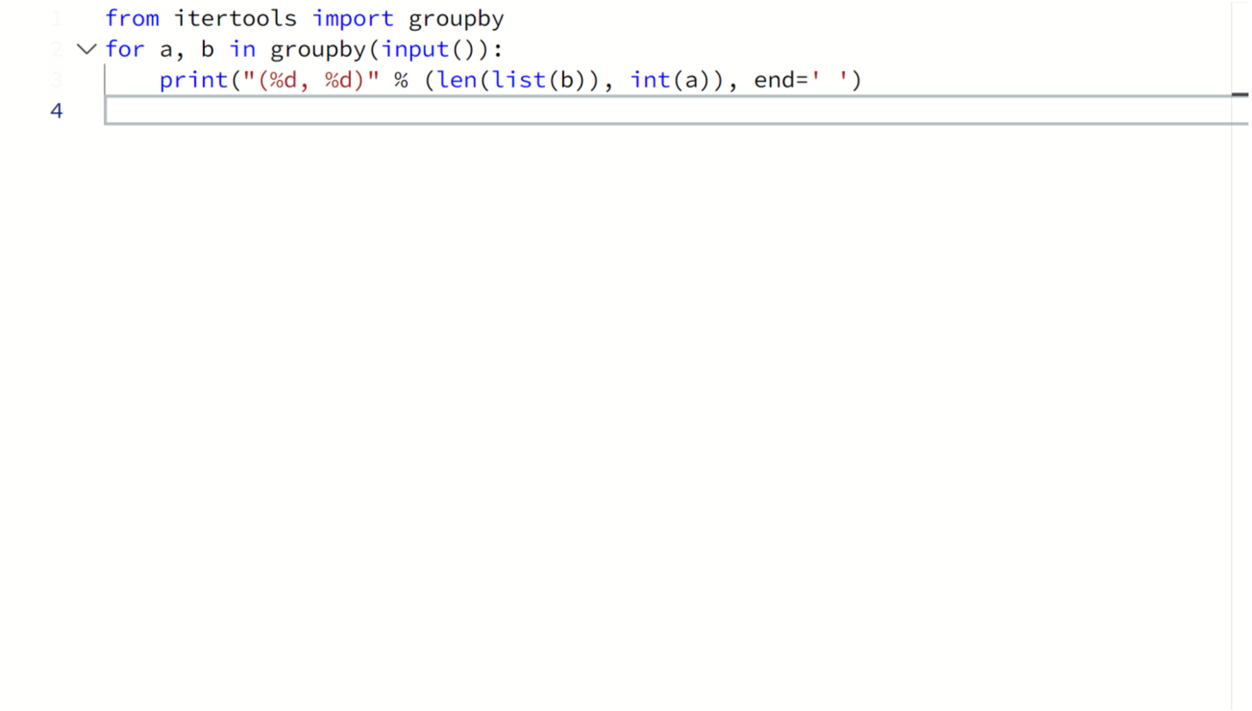
```
3
2 1 1
```

1.8 Compress the String!

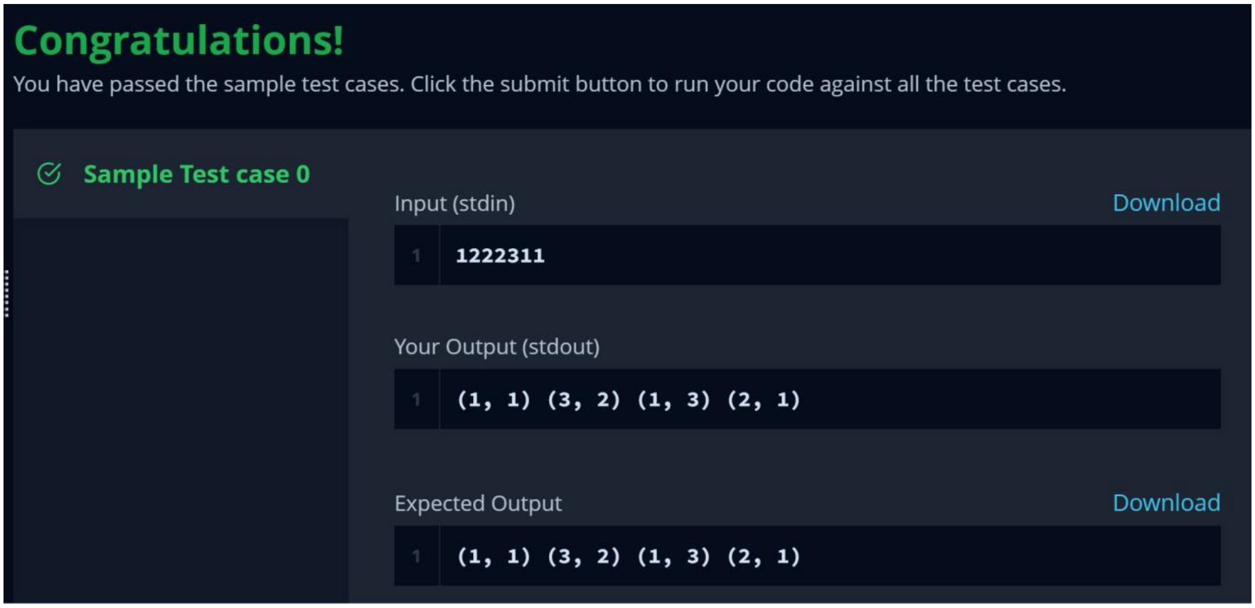
a. Code:

```
from itertools import groupby
for a, b in groupby(input()):
    print("(%d, %d)" % (len(list(b)), int(a)), end=' ')
```

b. Screenshots:



```
from itertools import groupby
for a, b in groupby(input()):
    print("(%d, %d)" % (len(list(b)), int(a)), end=' ')

```



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Sample Test case 0

Input (stdin)

```
1222311
```

Your Output (stdout)

```
(1, 1) (3, 2) (1, 3) (2, 1)
```

Expected Output

```
(1, 1) (3, 2) (1, 3) (2, 1)
```

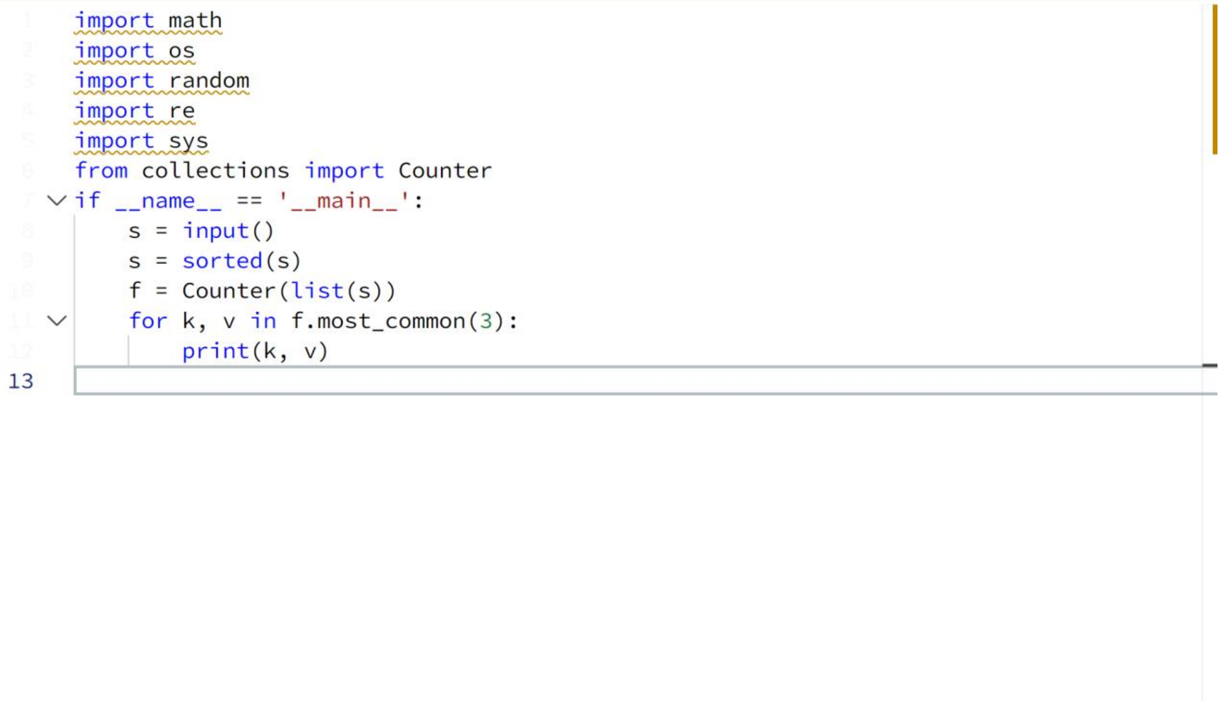
1.9 Company Logo

a. Code:

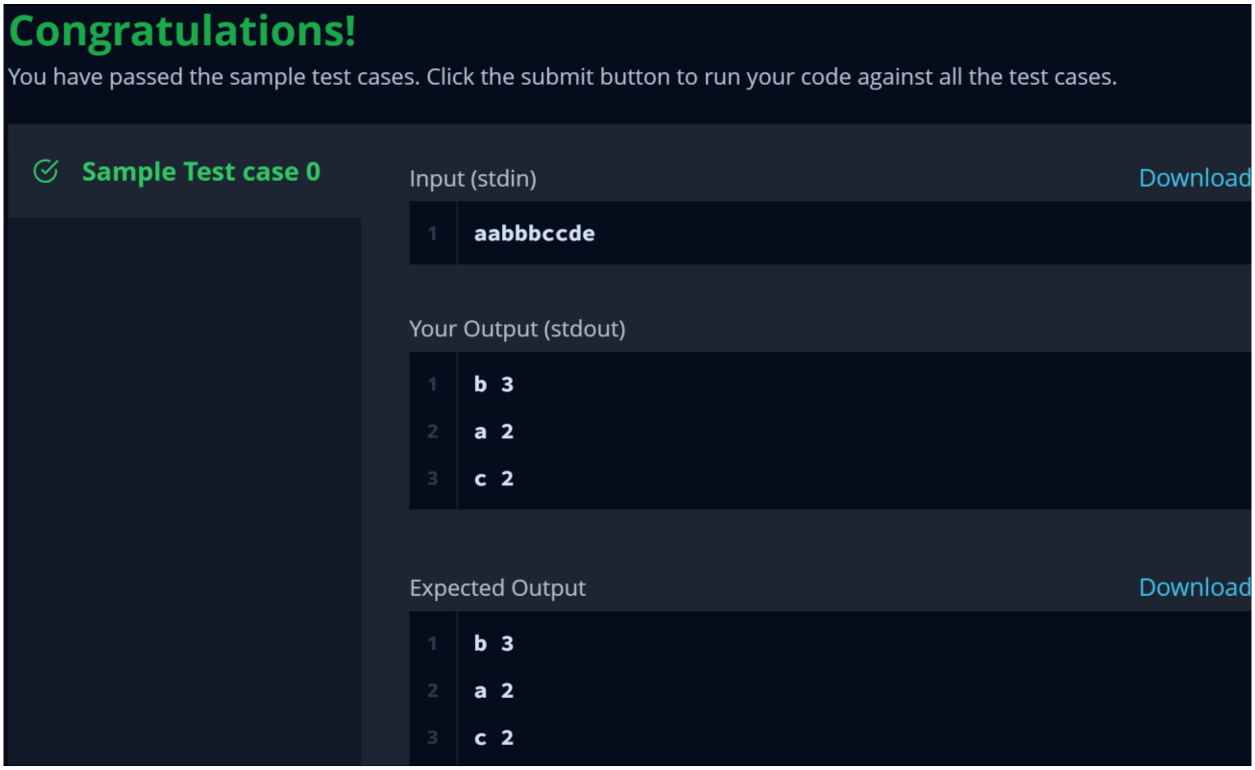
```
import math
import os
import random
import re
import sys
from collections import Counter
```

```
if __name__ == '__main__':
    s = input()
    s = sorted(s)
    f = Counter(list(s))
    for k, v in f.most_common(3):
        print(k, v)
```

b. Screenshots:



```
import math
import os
import random
import re
import sys
from collections import Counter
if __name__ == '__main__':
    s = input()
    s = sorted(s)
    f = Counter(list(s))
    for k, v in f.most_common(3):
        print(k, v)

```



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

```
aabbbccde
```

```
b 3
a 2
c 2
```

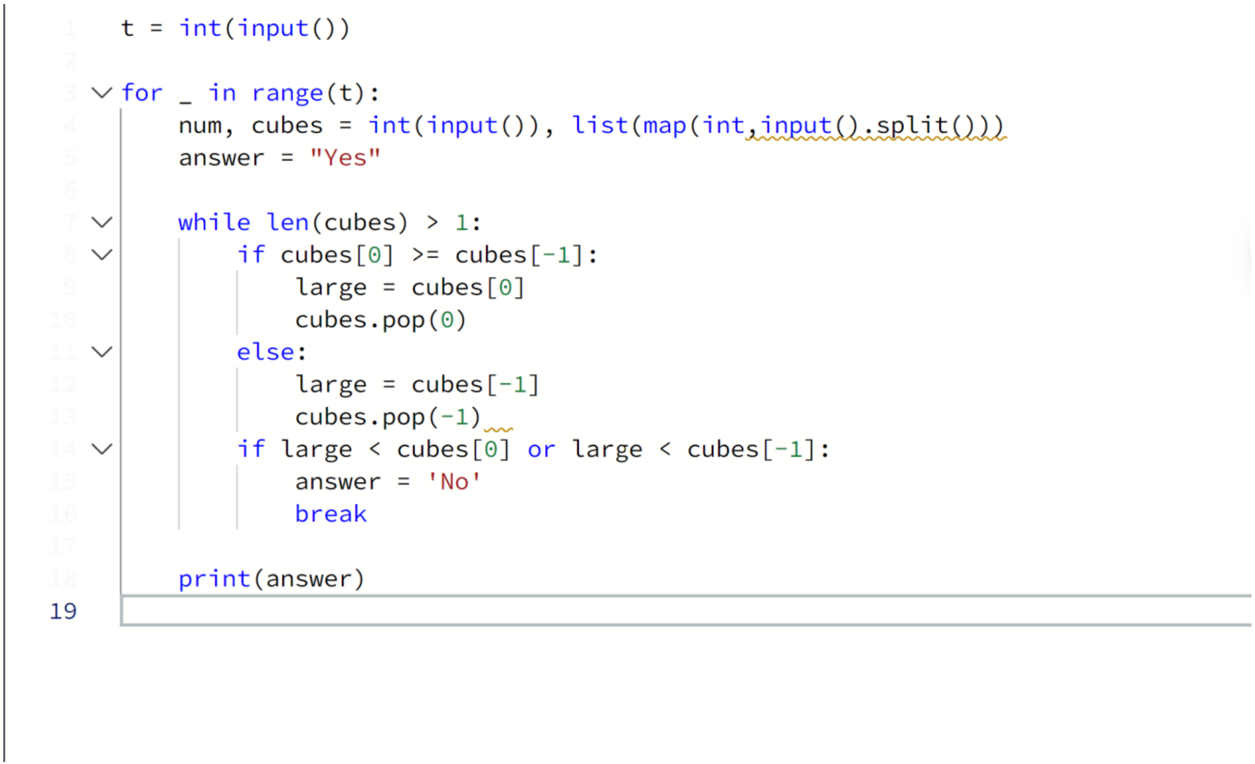
```
b 3
a 2
c 2
```

1.10 Piling Up!

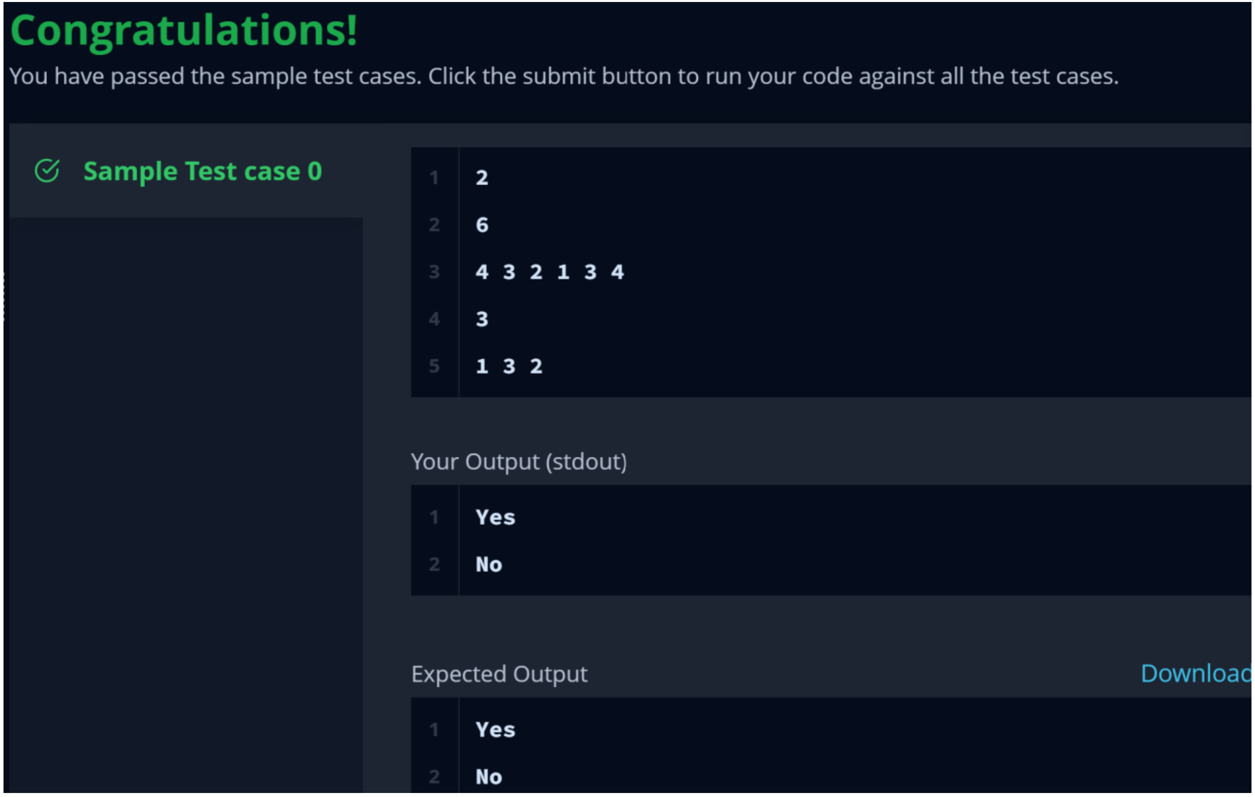
a. Code:

```
t = int(input())

for _ in range(t):
    num, cubes = int(input()), list(map(int,input().split()))
    answer = "Yes"

    while len(cubes) > 1:
        if cubes[0] >= cubes[-1]:
            large = cubes[0]
            cubes.pop(0)
        else:
            large = cubes[-1]
            cubes.pop(-1)
        if large < cubes[0] or large < cubes[-1]:
            answer = 'No'
            break

    print(answer)
```

b. Screenshots:



```
t = int(input())

for _ in range(t):
    num, cubes = int(input()), list(map(int,input().split()))
    answer = "Yes"

    while len(cubes) > 1:
        if cubes[0] >= cubes[-1]:
            large = cubes[0]
            cubes.pop(0)
        else:
            large = cubes[-1]
            cubes.pop(-1)
        if large < cubes[0] or large < cubes[-1]:
            answer = 'No'
            break

    print(answer)
```



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

```
2
6
4 3 2 1 3 4
3
1 3 2
```

Your Output (stdout)

```
Yes
No
```

Expected Output

```
Yes
No
```

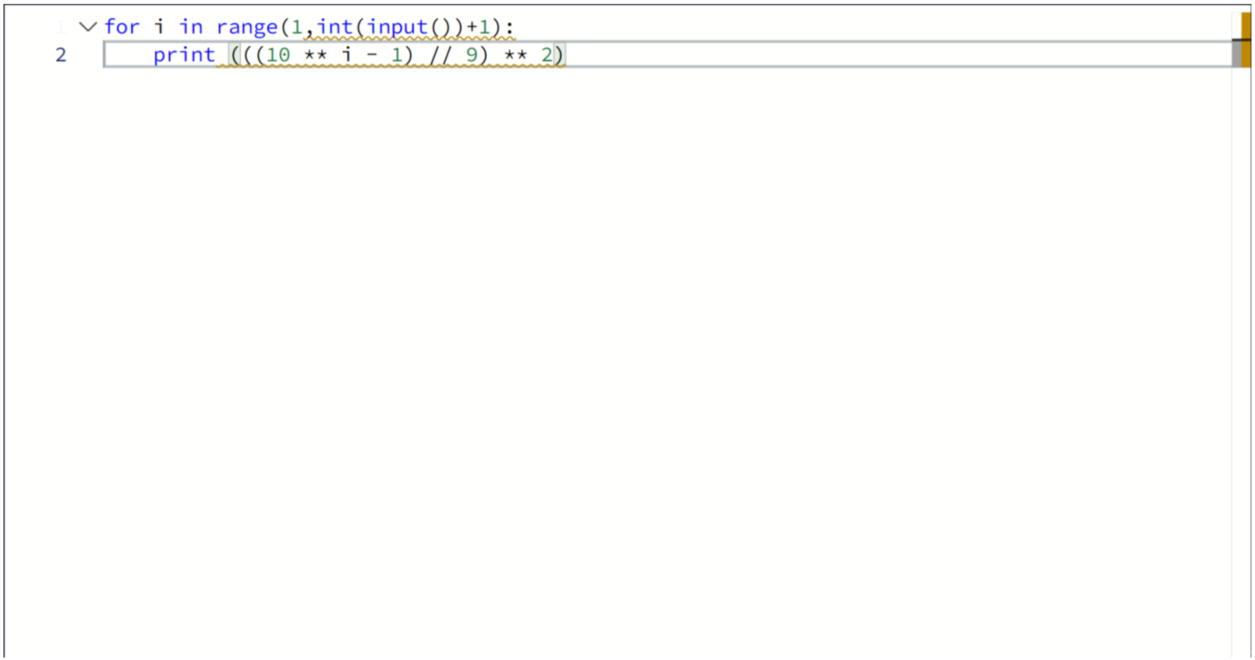
1.11 Triangle Quest 2

a. Code:

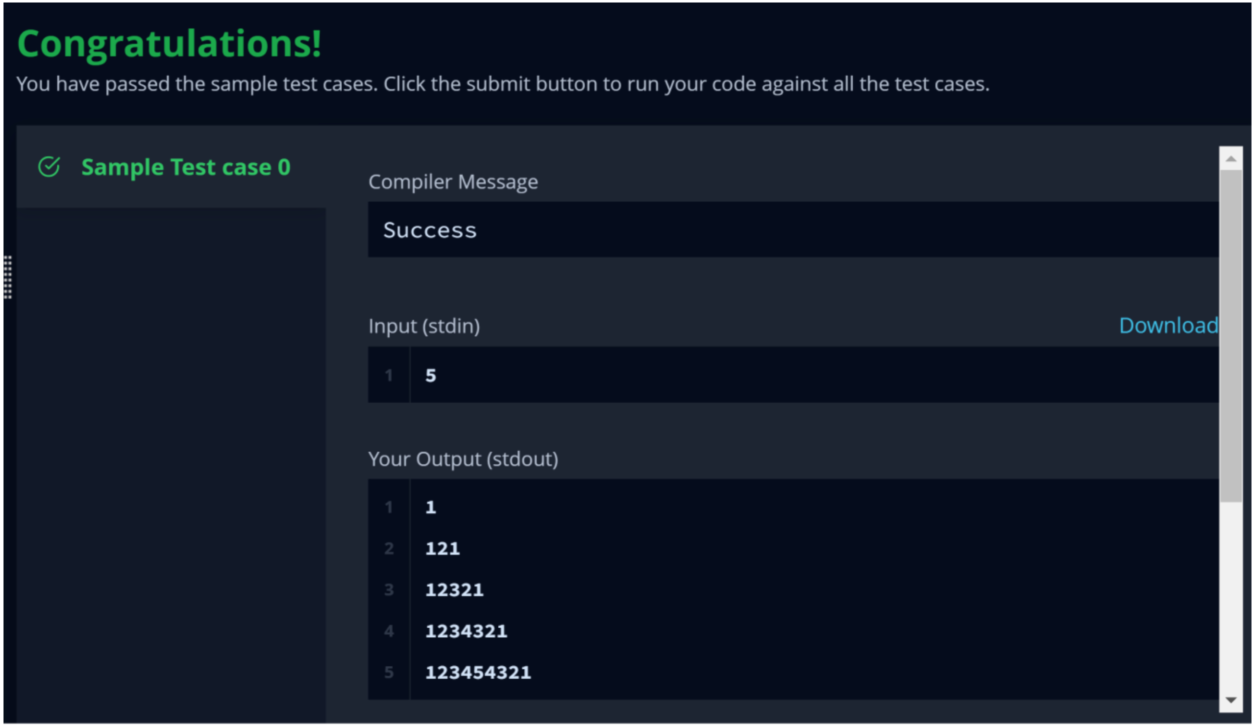
for i in range(1,int(input())+1):

```
print (((10 ** i - 1) // 9) ** 2)
```

b. Screenshots:



```
for i in range(1,int(input())+1):
    print (((10 ** i - 1) // 9) ** 2)
```



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Compiler Message

```
Success
```

Input (stdin)

```
5
```

Your Output (stdout)

```
1
121
12321
1234321
123454321
```

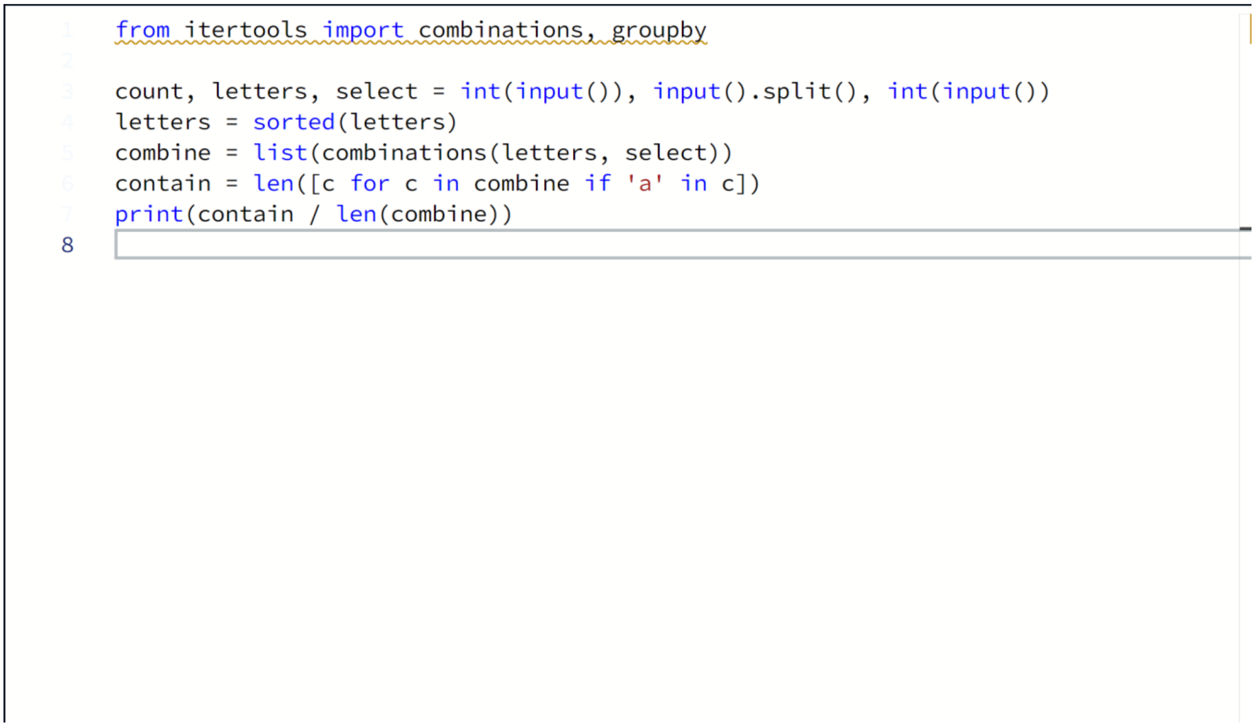
1.12 Iterables and Iterators

a. Code:

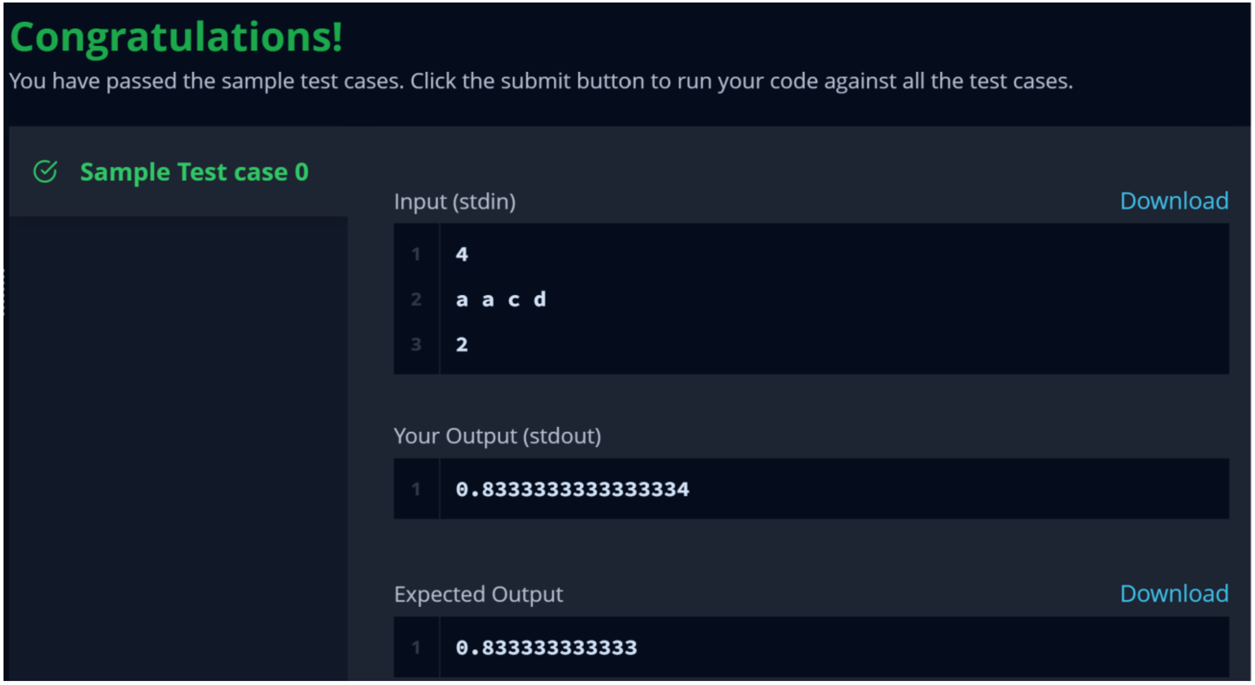
```
from itertools import combinations, groupby
```

```
count, letters, select = int(input()), input().split(), int(input())
letters = sorted(letters)
combine = list(combinations(letters, select))
contain = len([c for c in combine if 'a' in c])
print(contain / len(combine))
```

b. Screenshots:



```
from itertools import combinations, groupby

count, letters, select = int(input()), input().split(), int(input())
letters = sorted(letters)
combine = list(combinations(letters, select))
contain = len([c for c in combine if 'a' in c])
print(contain / len(combine))

```



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Input (stdin)

```
4
a a c d
2
```

Your Output (stdout)

```
0.8333333333333334
```

Expected Output

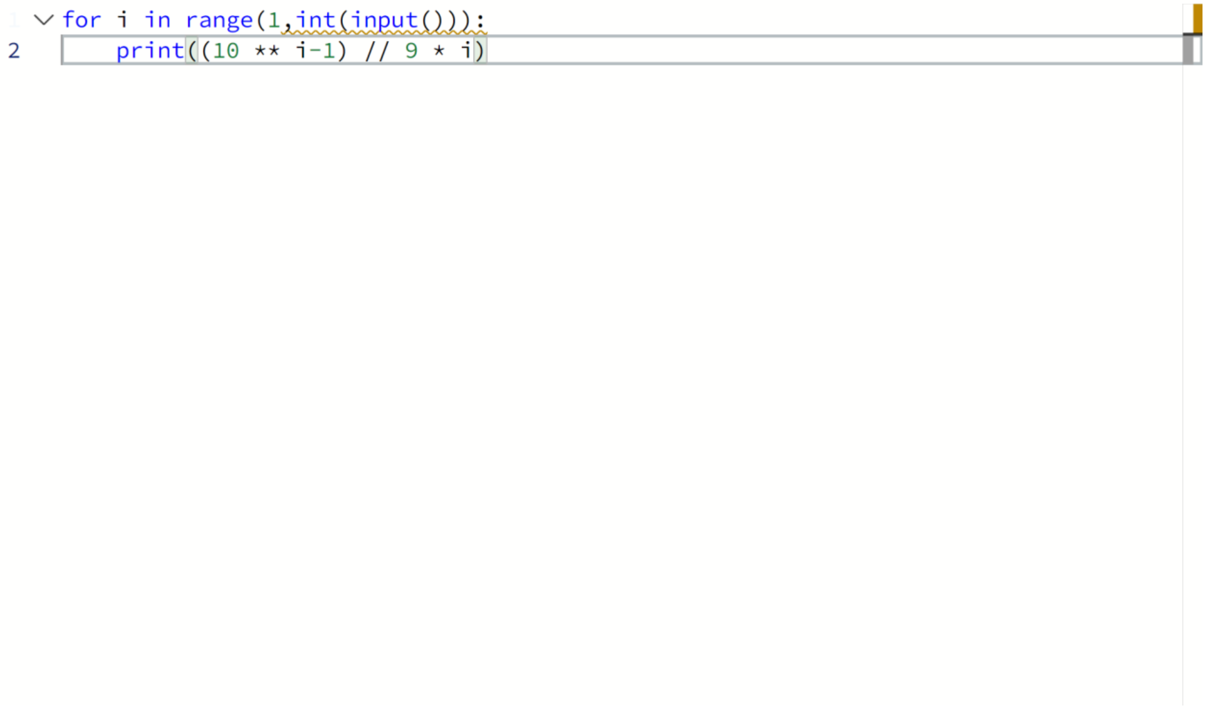
```
0.833333333333
```

1.13 Triangle Quest

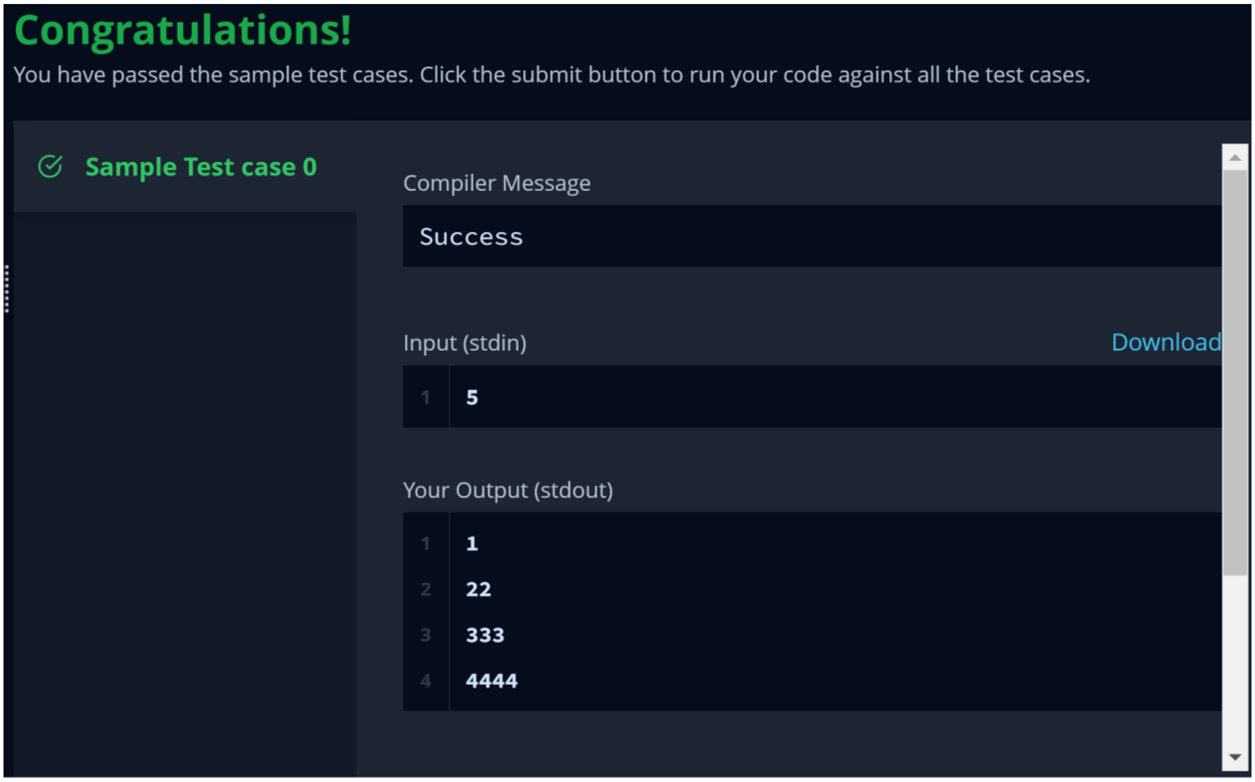
a. Code:

```
for i in range(1,int(input())):
   print((10 ** i-1) // 9 * i)
```

b. Screenshots:



```
for i in range(1,int(input())):
    print((10 ** i-1) // 9 * i)
```



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Sample Test case 0

Compiler Message

```
Success
```

Input (stdin)

```
5
```

Your Output (stdout)

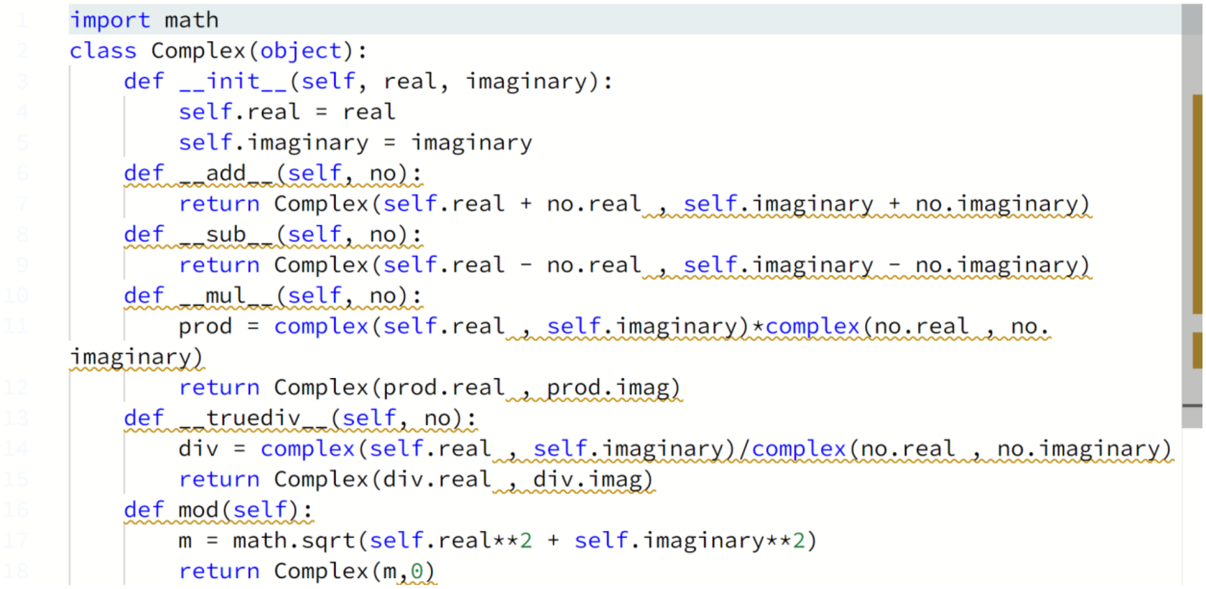
```
1
22
333
4444
```

1.14 Classes: Dealing with Complex Numbers

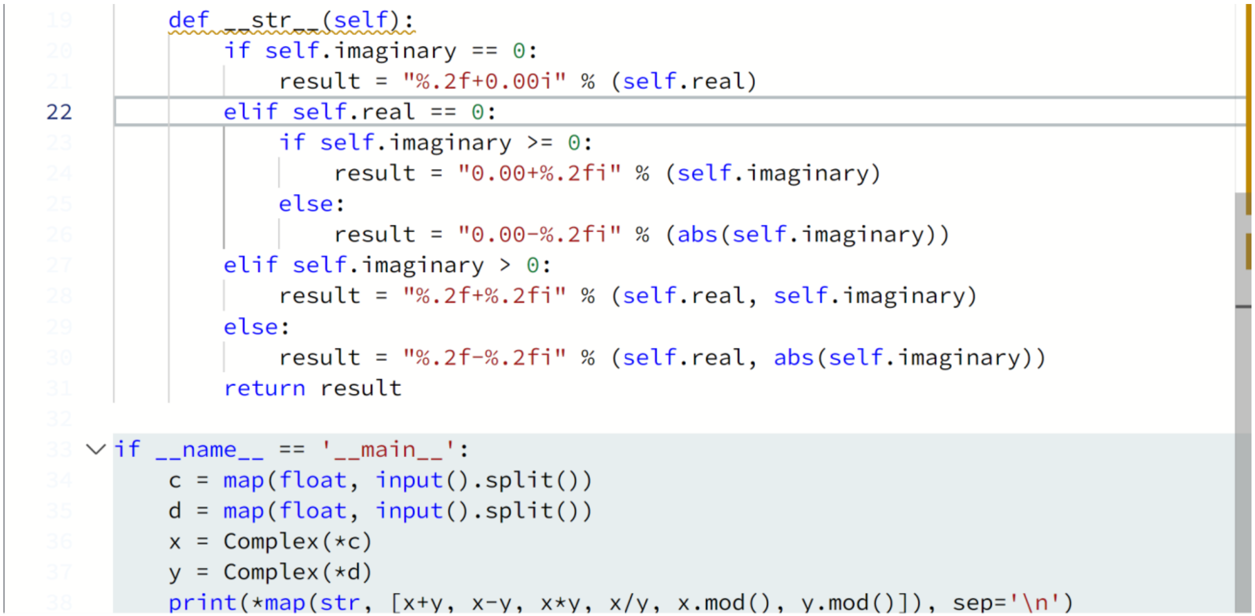
a. Code:

```
import math
class Complex(object):
    def __init__(self, real, imaginary):
        self.real = real
        self.imaginary = imaginary
    def __add__(self, no):
        return Complex(self.real + no.real , self.imaginary + no.imaginary)
    def __sub__(self, no):
        return Complex(self.real - no.real , self.imaginary - no.imaginary)
    def __mul__(self, no):
        prod = complex(self.real , self.imaginary)*complex(no.real , no.imaginary)
        return Complex(prod.real , prod.imag)
    def __truediv__(self, no):
        div = complex(self.real , self.imaginary)/complex(no.real , no.imaginary)
        return Complex(div.real , div.imag)
    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

if __name__ == '__main__':
    c = map(float, input().split())
    d = map(float, input().split())
    x = Complex(*c)
    y = Complex(*d)
    print(*map(str, [x+y, x-y, x*y, x/y, x.mod(), y.mod()]), sep='\n')
```

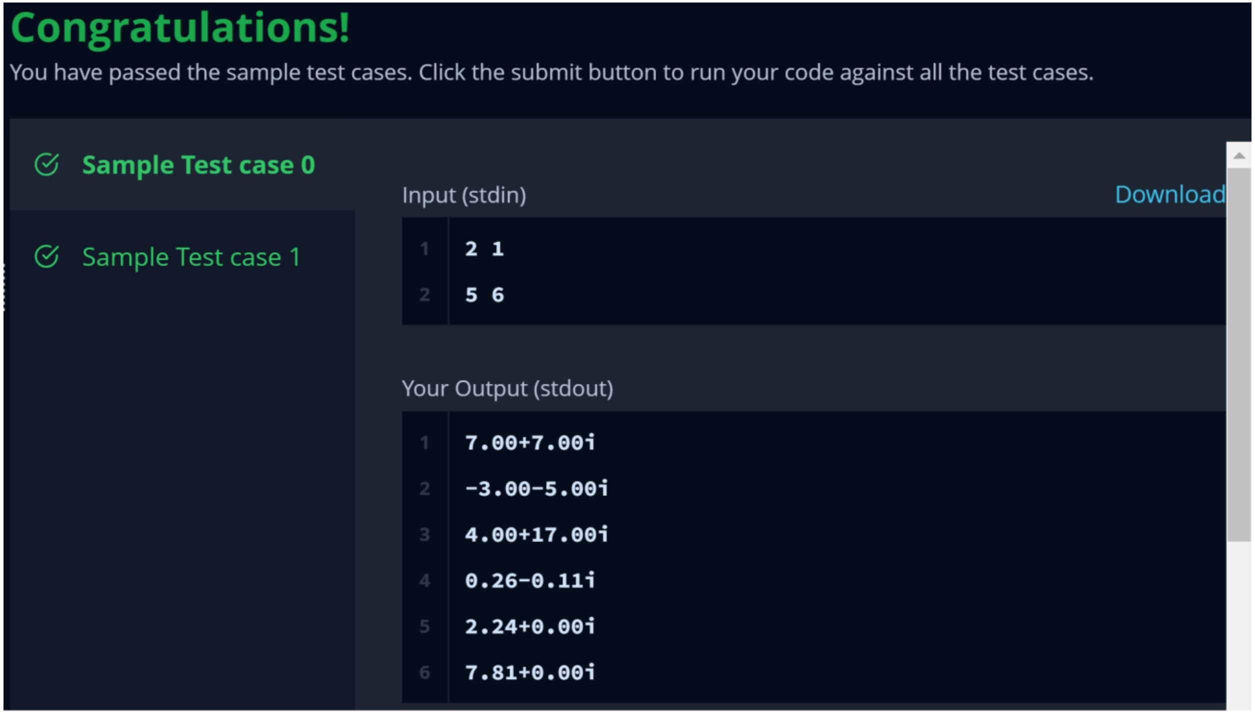
b. Screenshots:



```
import math
class Complex(object):
    def __init__(self, real, imaginary):
        self.real = real
        self.imaginary = imaginary
    def __add__(self, no):
        return Complex(self.real + no.real , self.imaginary + no.imaginary)
    def __sub__(self, no):
        return Complex(self.real - no.real , self.imaginary - no.imaginary)
    def __mul__(self, no):
        prod = complex(self.real , self.imaginary)*complex(no.real , no.
imaginary)
        return Complex(prod.real , prod.imag)
    def __truediv__(self, no):
        div = complex(self.real , self.imaginary)/complex(no.real , no.imaginary)
        return Complex(div.real , div.imag)
    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

if __name__ == '__main__':
    c = map(float, input().split())
    d = map(float, input().split())
    x = Complex(*c)
    y = Complex(*d)
    print(*map(str, [x+y, x-y, x*y, x/y, x.mod(), y.mod()]), sep='\n')
```



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Sample Test case 0

Sample Test case 1

Input (stdin)

```
2 1
5 6
```

Your Output (stdout)

```
7.00+7.00i
-3.00-5.00i
4.00+17.00i
0.26-0.11i
2.24+0.00i
7.81+0.00i
```

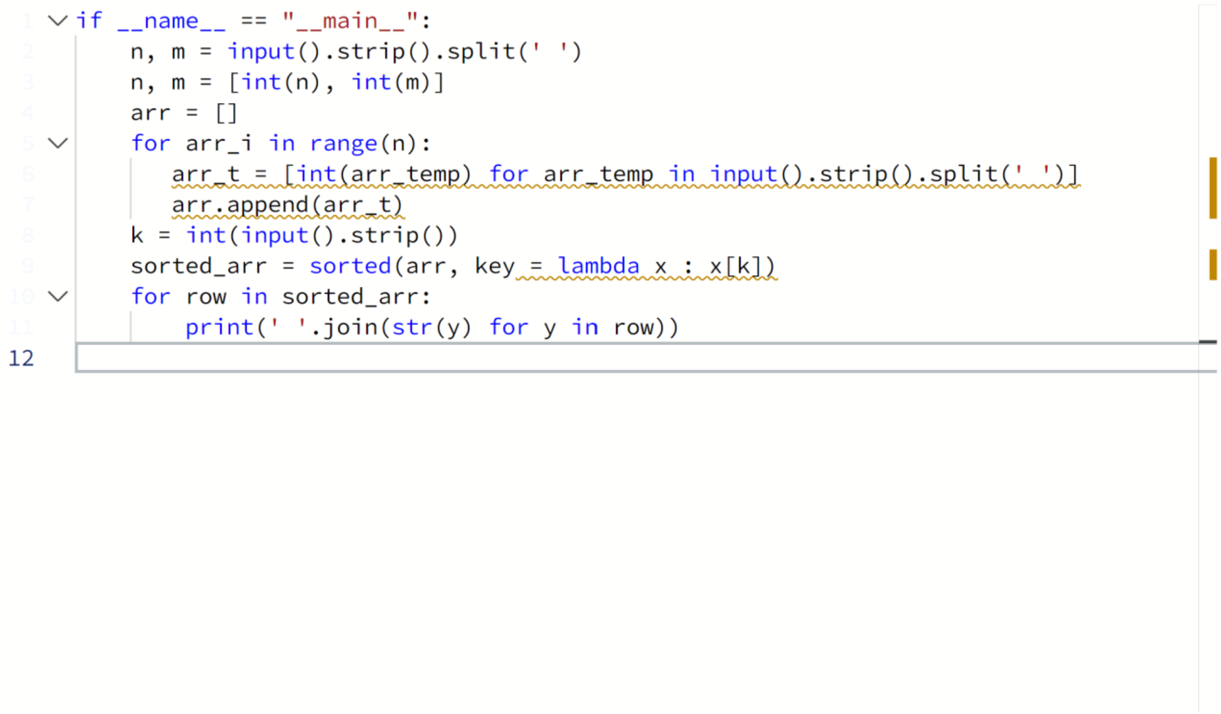
1.15 Athlete Sort

a. Code:

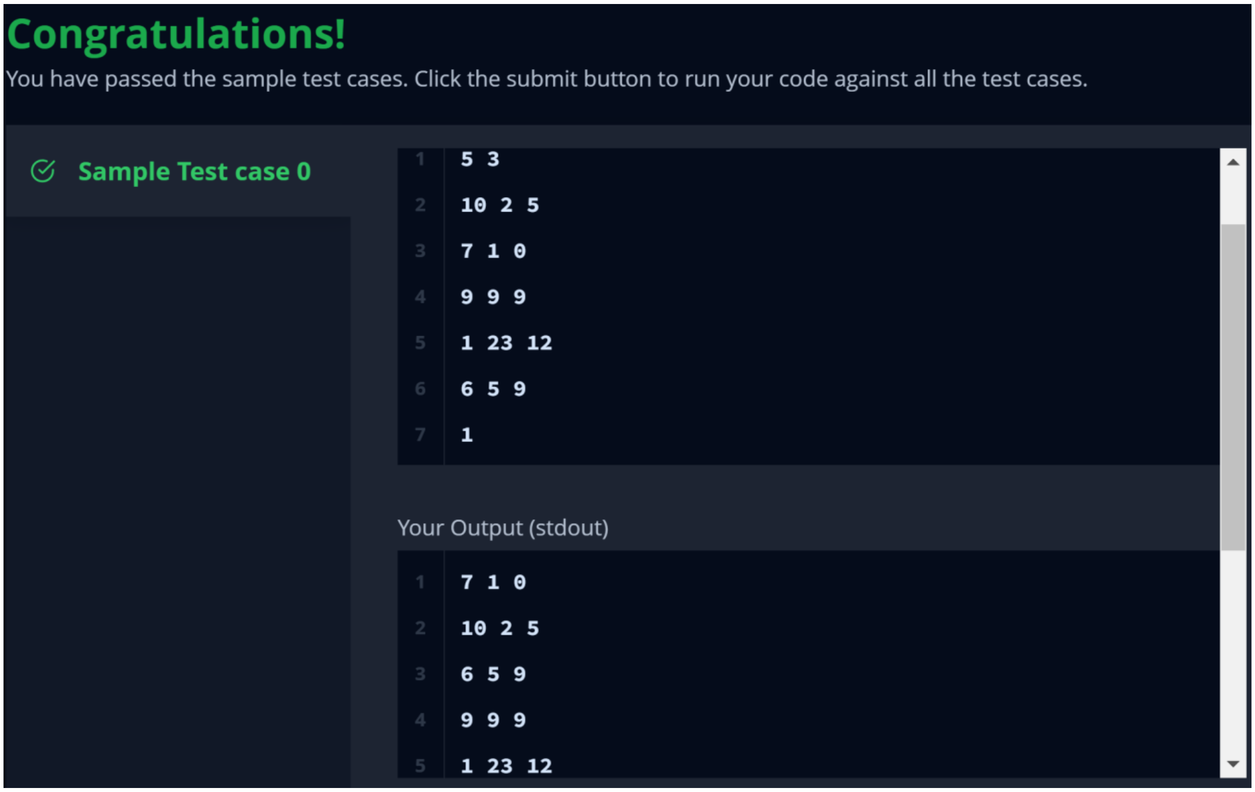
```
if __name__ == "__main__":
    n, m = input().strip().split(' ')
    n, m = [int(n), int(m)]
    arr = []
    for arr_i in range(n):
        arr_t = [int(arr_temp) for arr_temp in input().strip().split(' ')]
```

```
  arr.append(arr_t)
k = int(input().strip())
sorted_arr = sorted(arr, key = lambda x : x[k])
for row in sorted_arr:
   print(' '.join(str(y) for y in row))
```

b. Screenshots:



```
if __name__ == "__main__":
    n, m = input().strip().split(' ')
    n, m = [int(n), int(m)]
    arr = []
    for arr_i in range(n):
        arr_t = [int(arr_temp) for arr_temp in input().strip().split(' ')]
        arr.append(arr_t)
    k = int(input().strip())
    sorted_arr = sorted(arr, key = lambda x : x[k])
    for row in sorted_arr:
        print(' '.join(str(y) for y in row))
```



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

```
5 3
10 2 5
7 1 0
9 9 9
1 23 12
6 5 9
1
```

Your Output (stdout)

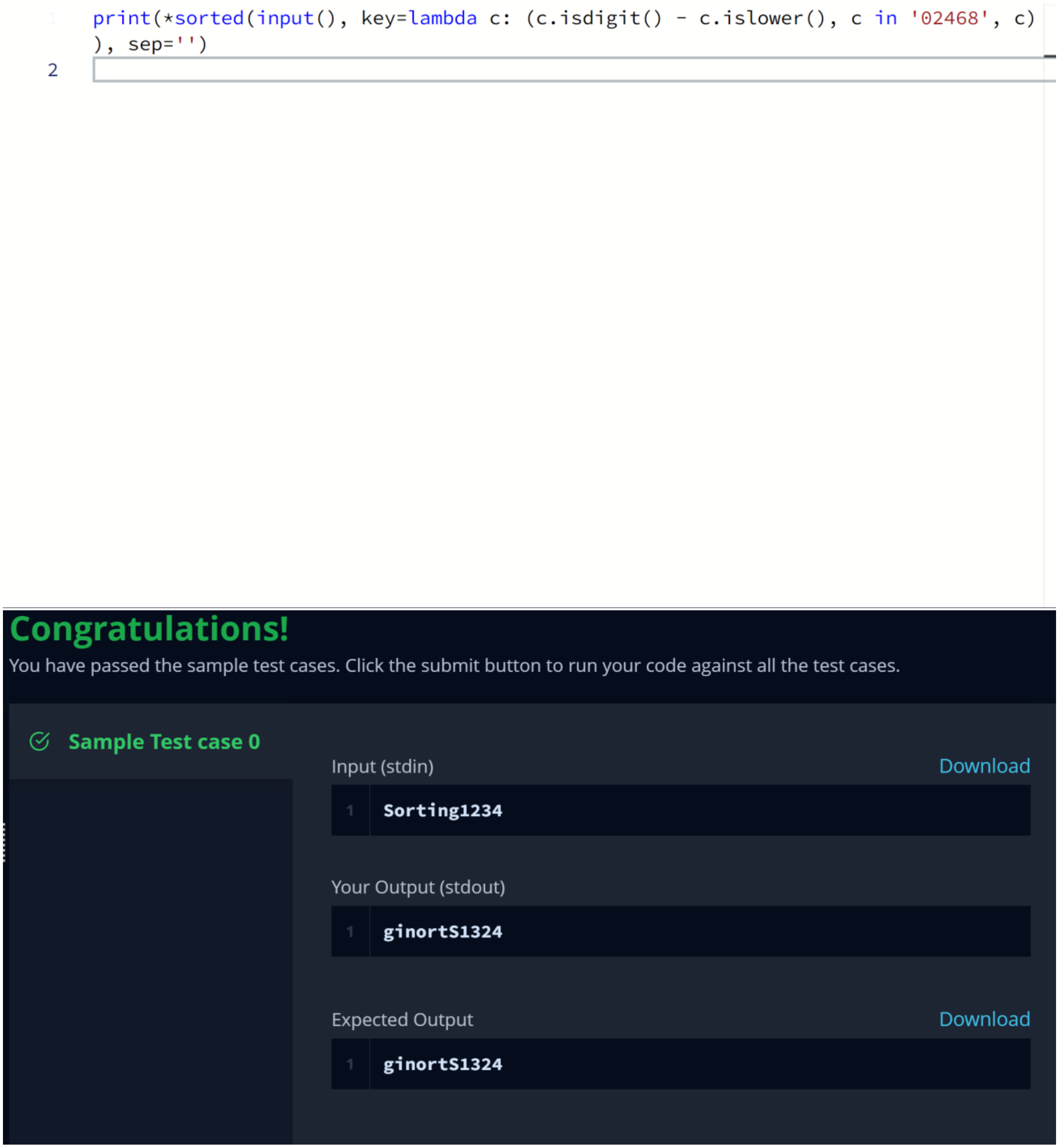
```
7 1 0
10 2 5
6 5 9
9 9 9
1 23 12
```

1.16 ginortS

a. Code:

```
print(*sorted(input(), key=lambda c: (c.isdigit() - c.islower(), c in '02468', c)), sep='')
```

b. Screenshots:



```
print(*sorted(input(), key=lambda c: (c.isdigit() - c.islower(), c in '02468', c)
), sep='')
```

Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Input (stdin)

```
Sorting1234
```

Your Output (stdout)

```
ginortS1324
```

Expected Output

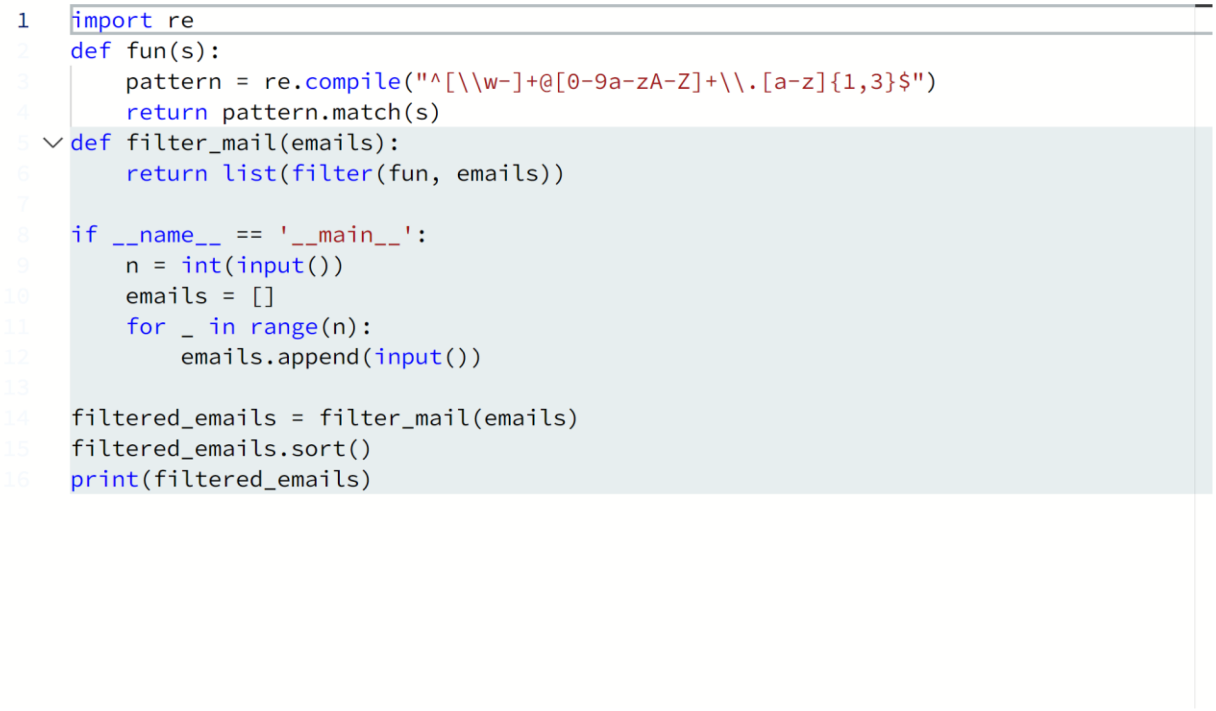
```
ginortS1324
```

1.17 Validating Email Addresses With a Filter

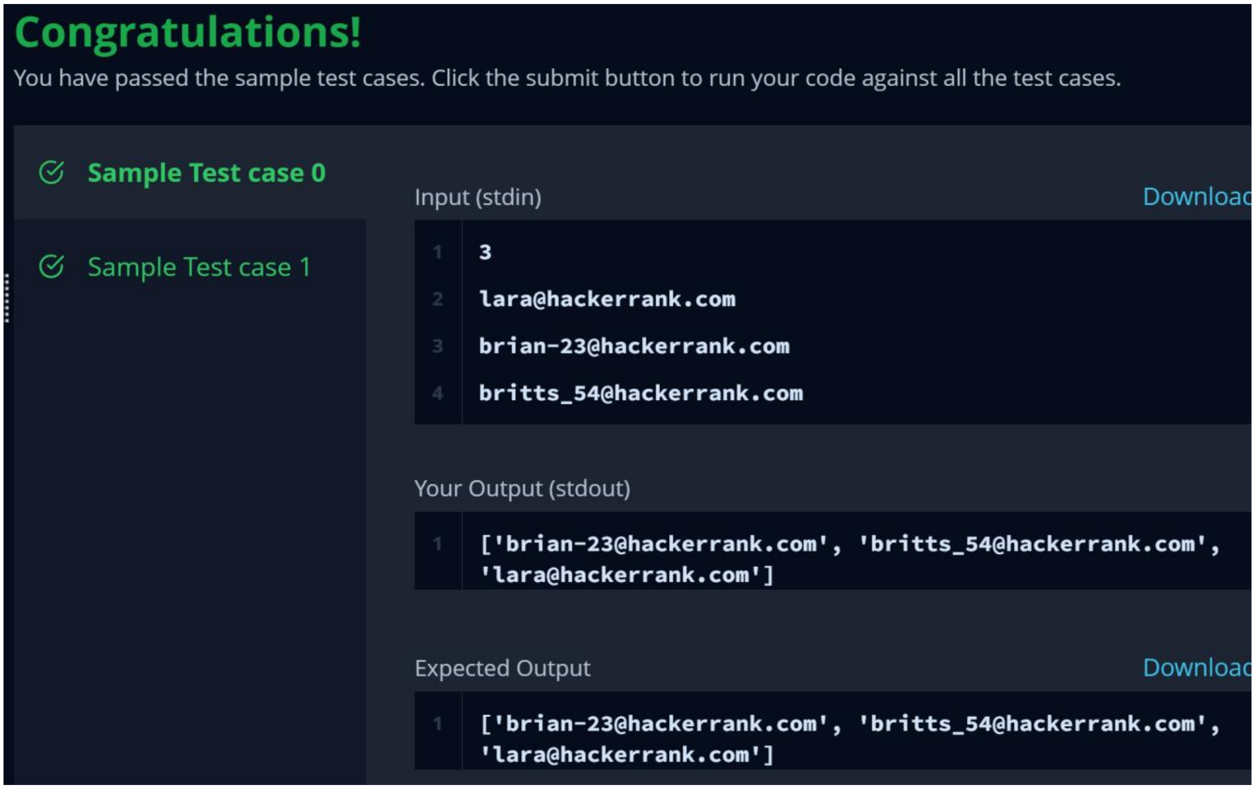
a. Code:

```
import re
def fun(s):
    pattern = re.compile("^[\\w-]+@[0-9a-zA-Z]+\\.[a-z]{1,3}$")
    return pattern.match(s)
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)
```

b. Screenshots:



```
import re
def fun(s):
    pattern = re.compile("^[\\w-]+@[0-9a-zA-Z]+\\.[a-z]{1,3}$")
    return pattern.match(s)
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)
```



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

```
3
lara@hackerrank.com
brian-23@hackerrank.com
britts_54@hackerrank.com
```

```
['brian-23@hackerrank.com', 'britts_54@hackerrank.com',
'lara@hackerrank.com']
```

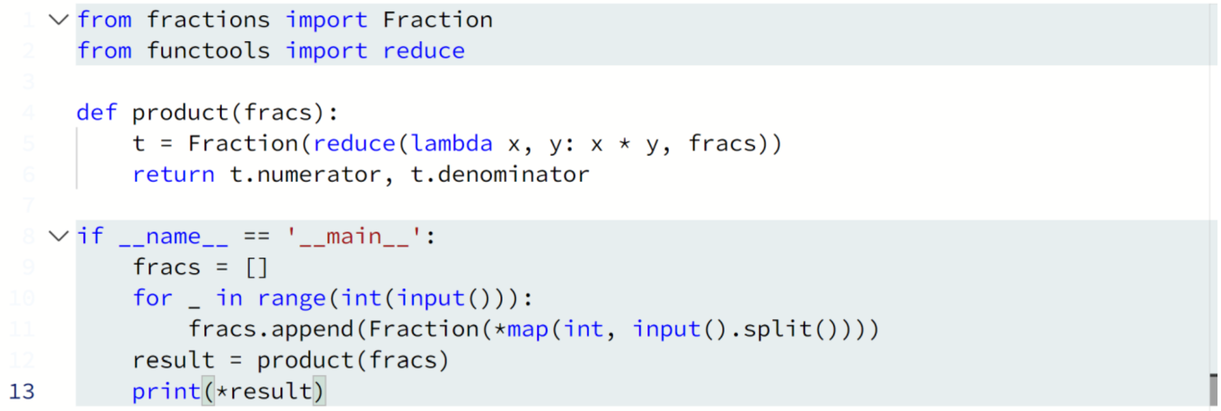
```
['brian-23@hackerrank.com', 'britts_54@hackerrank.com',
'lara@hackerrank.com']
```

1.18 Reduce Function

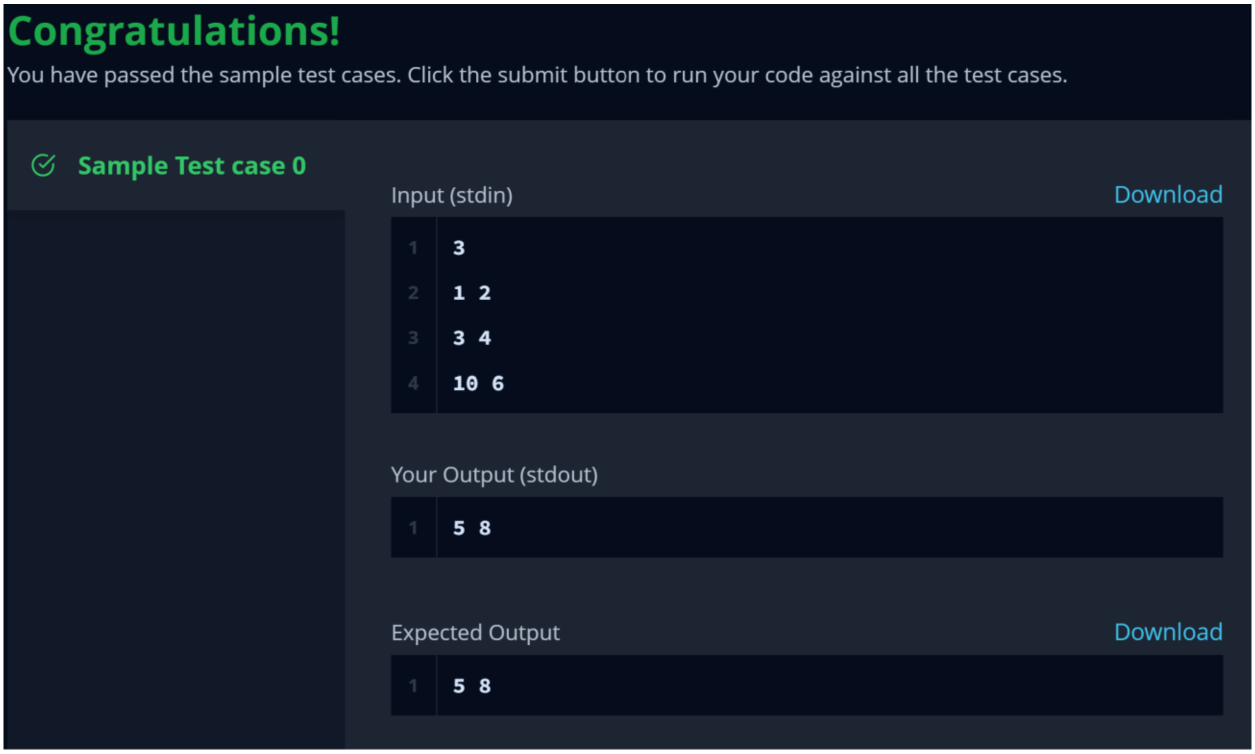
a. Code:

```
from fractions import Fraction
from functools import reduce

def product(fracs):
   t = Fraction(reduce(lambda x, y: x * y, fracs))
   return t.numerator, t.denominator

if __name__ == '__main__':
   fracs = []
   for _ in range(int(input())):
      fracs.append(Fraction(*map(int, input().split())))
   result = product(fracs)
   print(*result)
```

b. Screenshots:



```python
from fractions import Fraction
from functools import reduce

def product(fracs):
    t = Fraction(reduce(lambda x, y: x * y, fracs))
    return t.numerator, t.denominator

if __name__ == '__main__':
    fracs = []
    for _ in range(int(input())):
        fracs.append(Fraction(*map(int, input().split())))
    result = product(fracs)
    print(*result)
```



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Input (stdin)

```
3
1 2
3 4
10 6
```

Your Output (stdout)

```
5 8
```

Expected Output

```
5 8
```

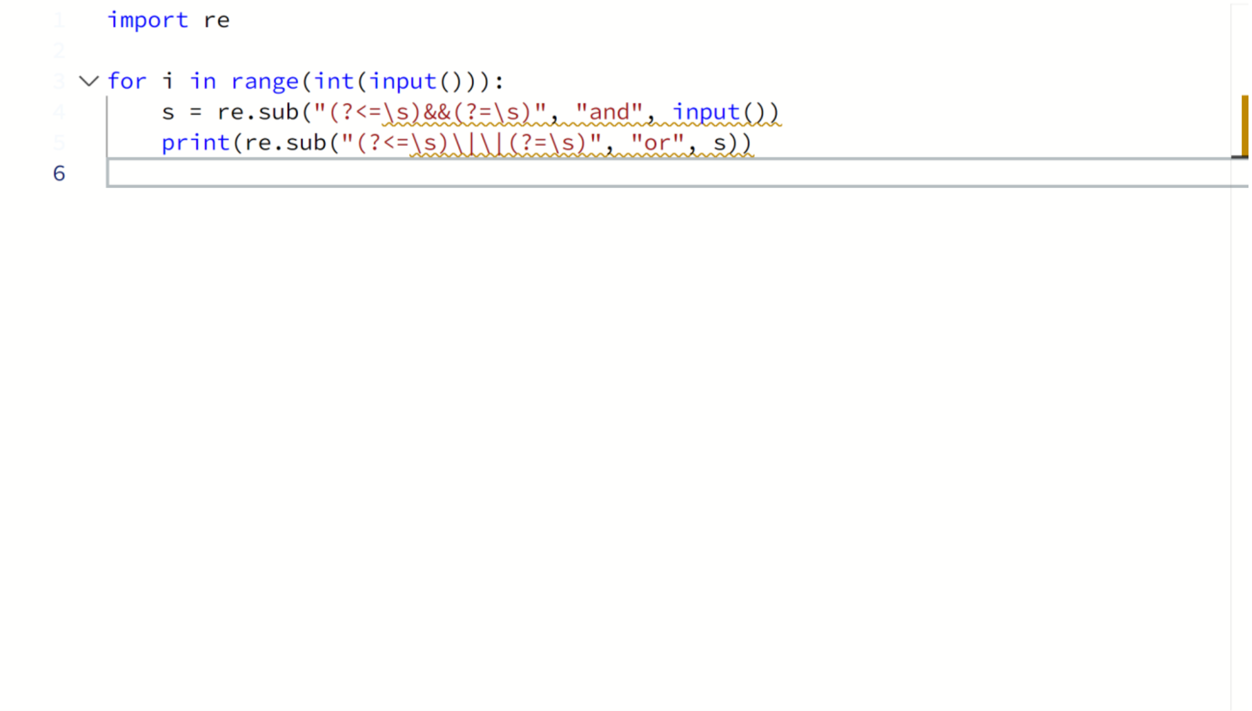
1.19 Regex Substitution

a. Code:

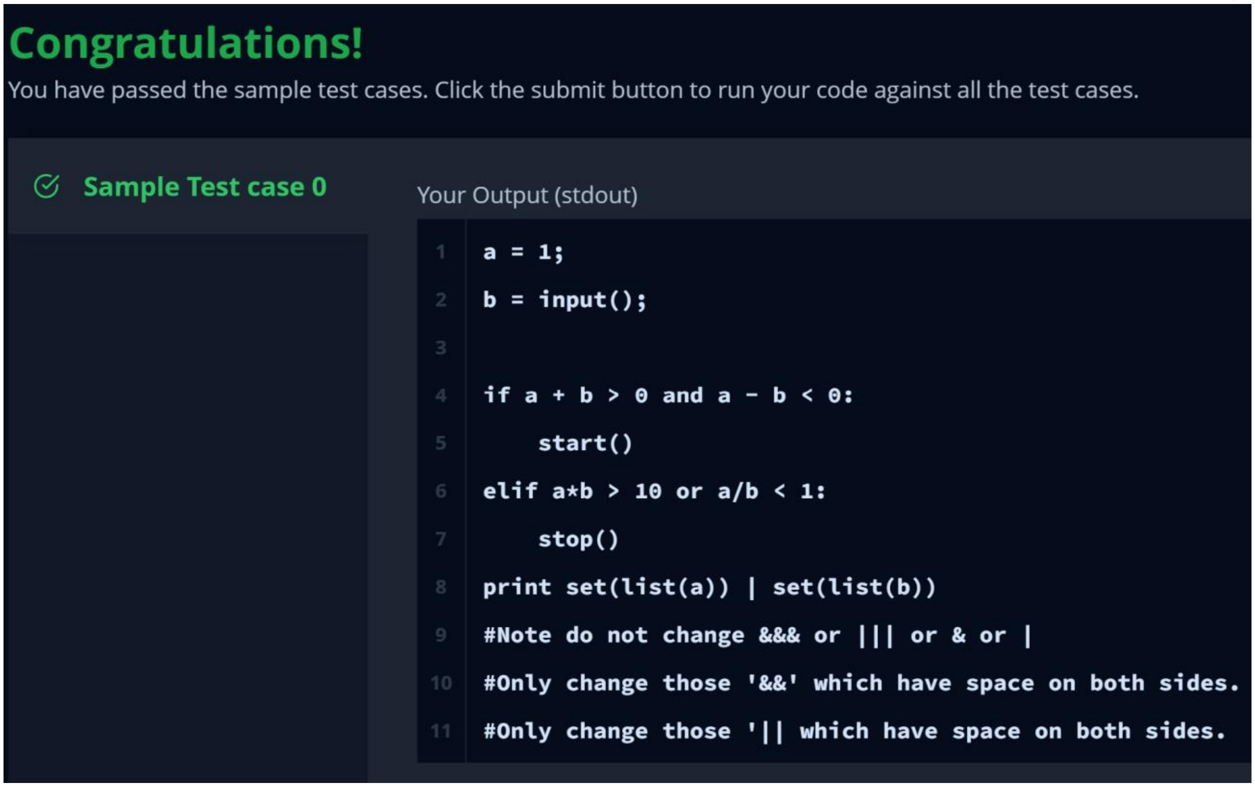
```
import re

for i in range(int(input())):
```

```
s = re.sub("(?<=\s)&&(?=\s)", "and", input())
print(re.sub("(?<=\s)\|\|(?=\s)", "or", s))
```

b. Screenshots:



```
import re

for i in range(int(input())):
    s = re.sub("(?<=\s)&&(?=\s)", "and", input())
    print(re.sub("(?<=\s)\|\|(?=\s)", "or", s))

```



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Sample Test case 0

Your Output (stdout)

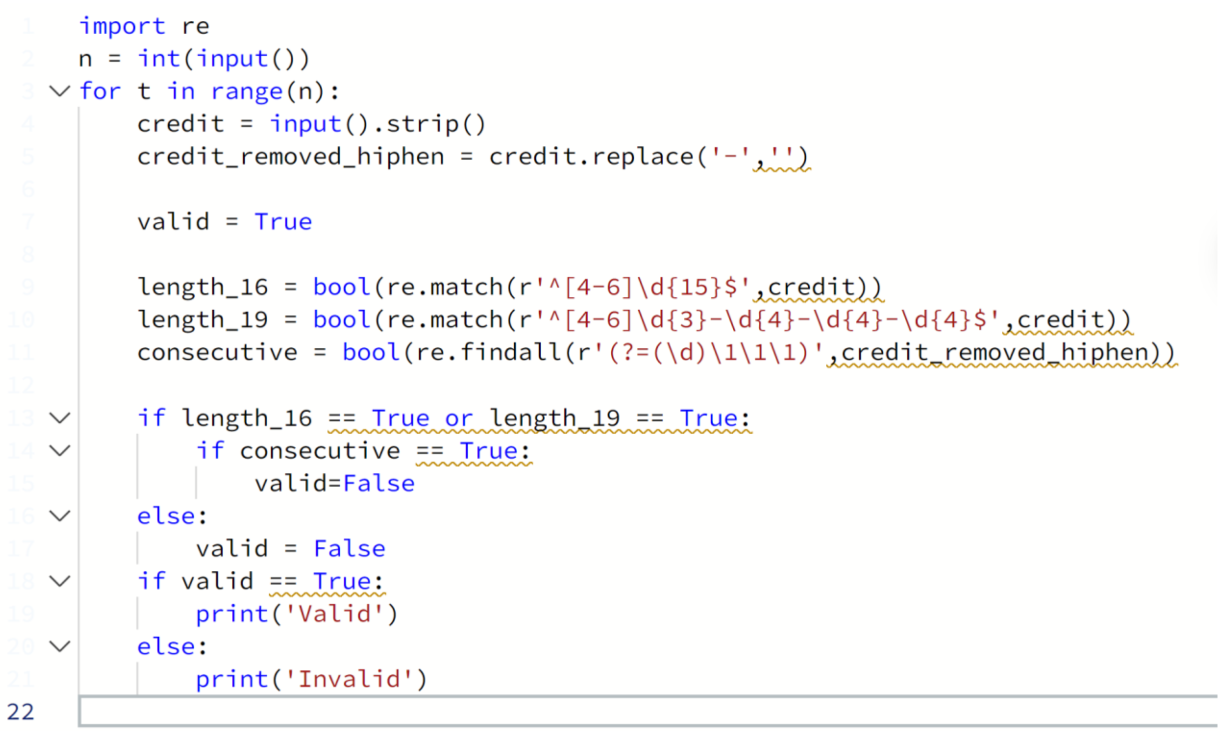
```
a = 1;
b = input();

if a + b > 0 and a - b < 0:
    start()
elif a*b > 10 or a/b < 1:
    stop()
print set(list(a)) | set(list(b))
#Note do not change &&& or ||| or & or |
#Only change those '&&' which have space on both sides.
#Only change those '|| which have space on both sides.
```

1.20 Validating Credit Card Numbers

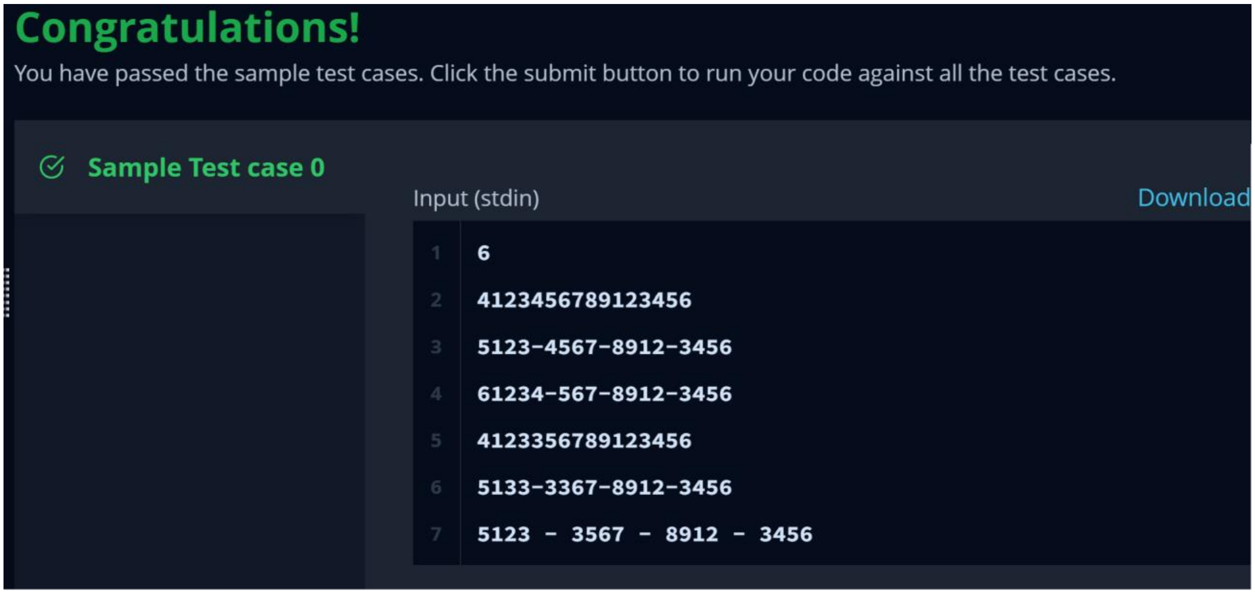
a. Code:

```
import re
n = int(input())
for t 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 == True:
        print('Valid')
    else:
        print('Invalid')
```

b. Screenshots:



```python
import re
n = int(input())
for t 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 == True:
        print('Valid')
    else:
        print('Invalid')
```

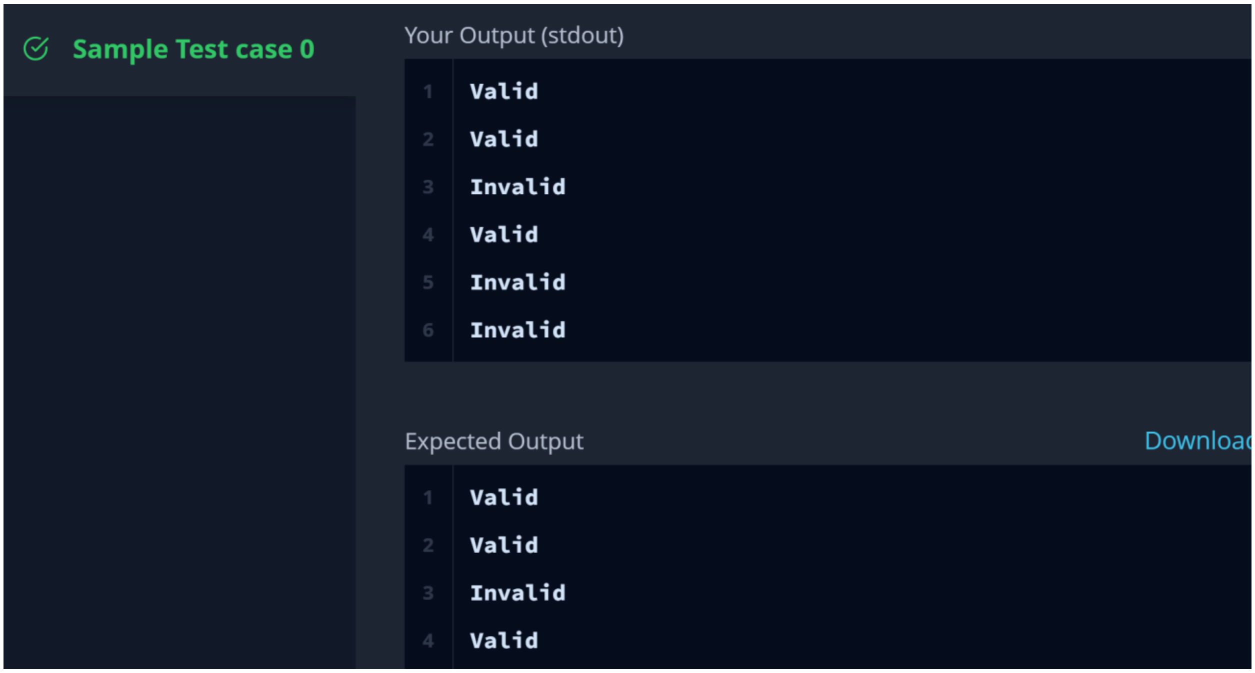


Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Input (stdin)

```
6
4123456789123456
5123-4567-8912-3456
61234-567-8912-3456
4123356789123456
5133-3367-8912-3456
5123 - 3567 - 8912 - 3456
```



```
Valid
Valid
Invalid
Valid
Invalid
Invalid
```

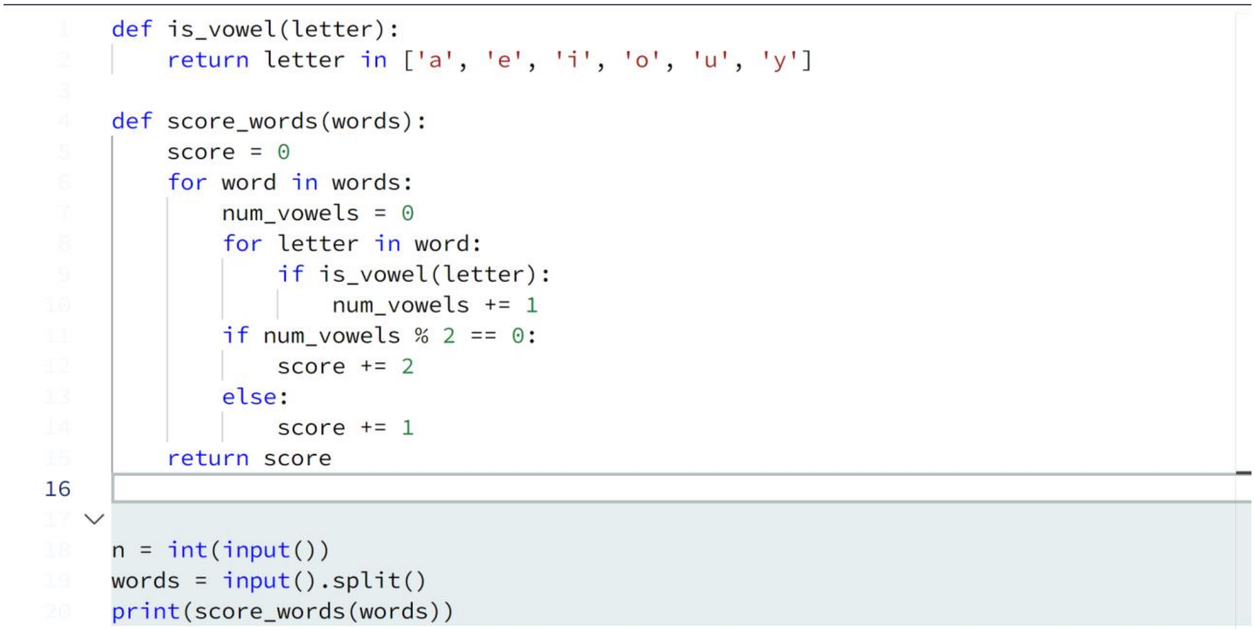
```
Valid
Valid
Invalid
Valid
```

1.21 Words Score

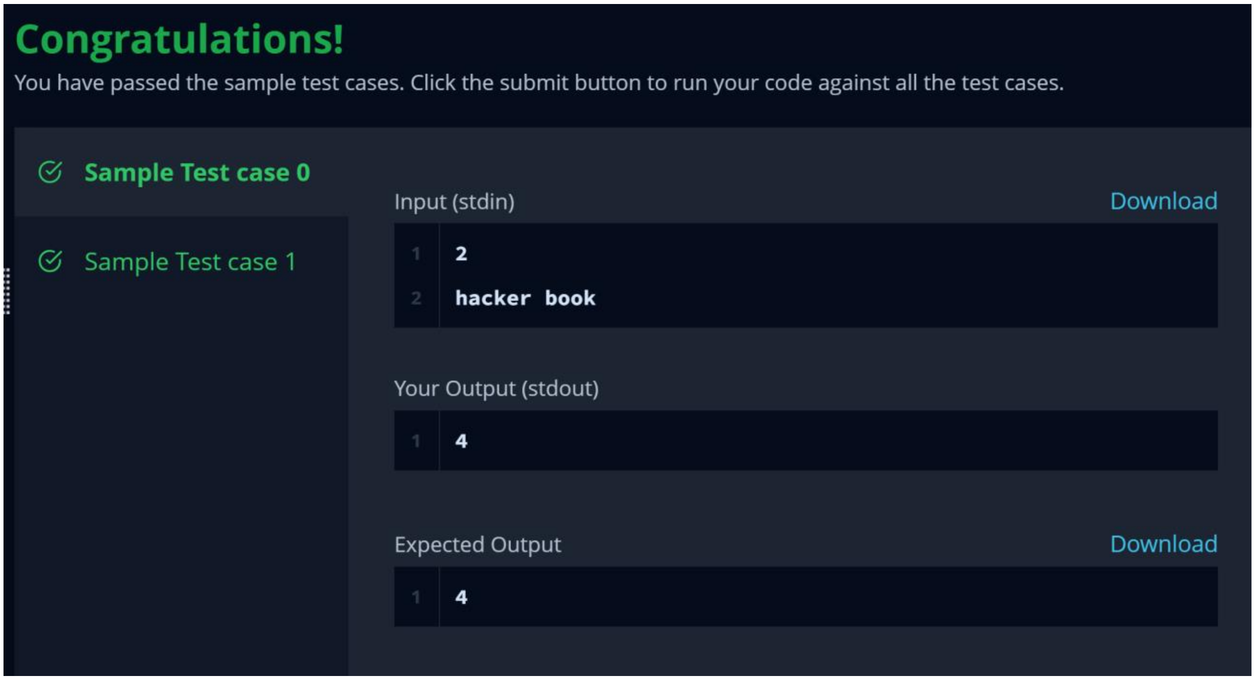
a. Code:

```
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 += 1
    return score


n = int(input())
words = input().split()
print(score_words(words))
```

b. Screenshots:



```python
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 += 1
    return score


n = int(input())
words = input().split()
print(score_words(words))
```



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Input (stdin)

```
2
hacker book
```

Your Output (stdout)

```
4
```

Expected Output

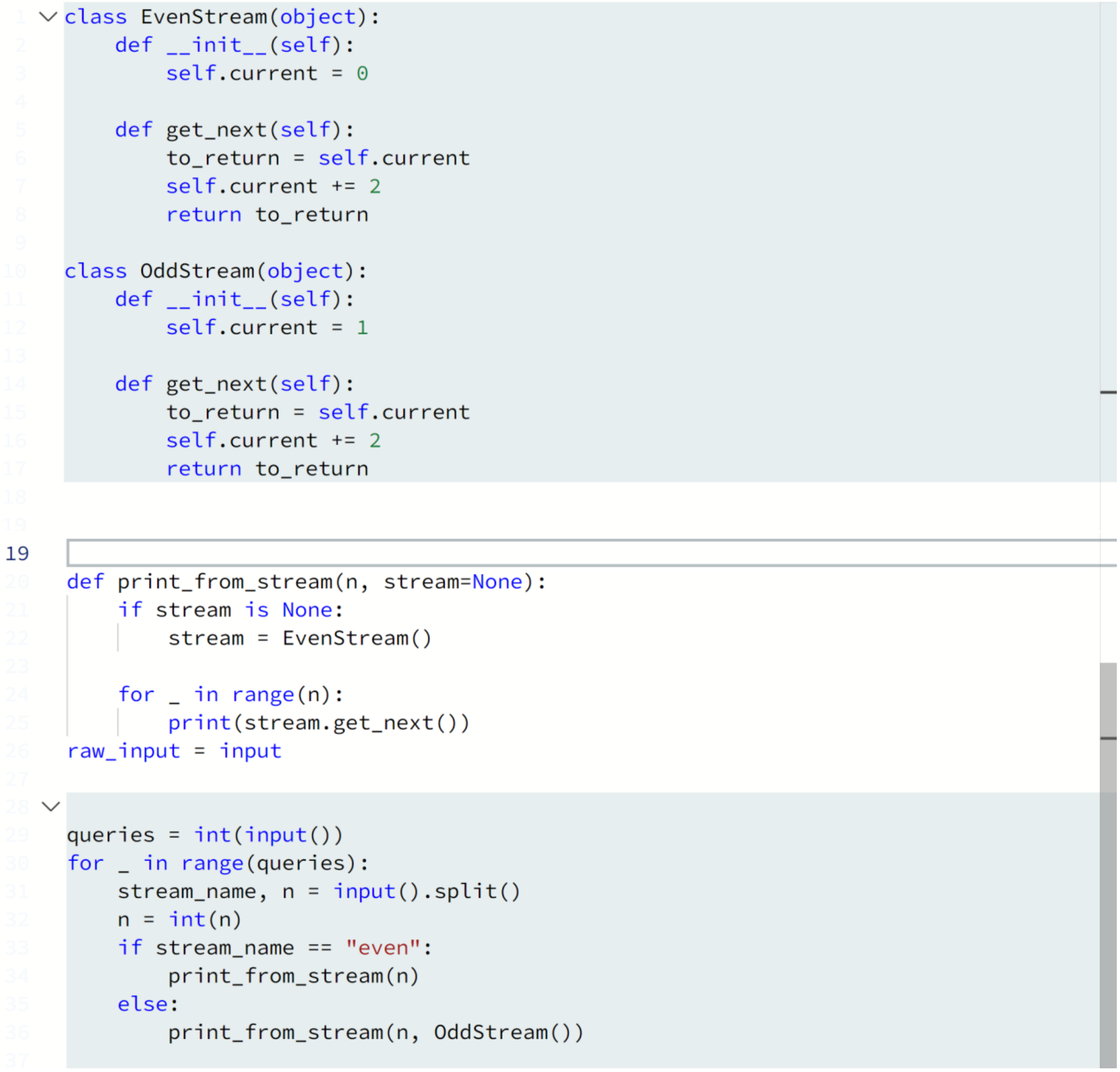
```
4
```

1.22 Default Arguments

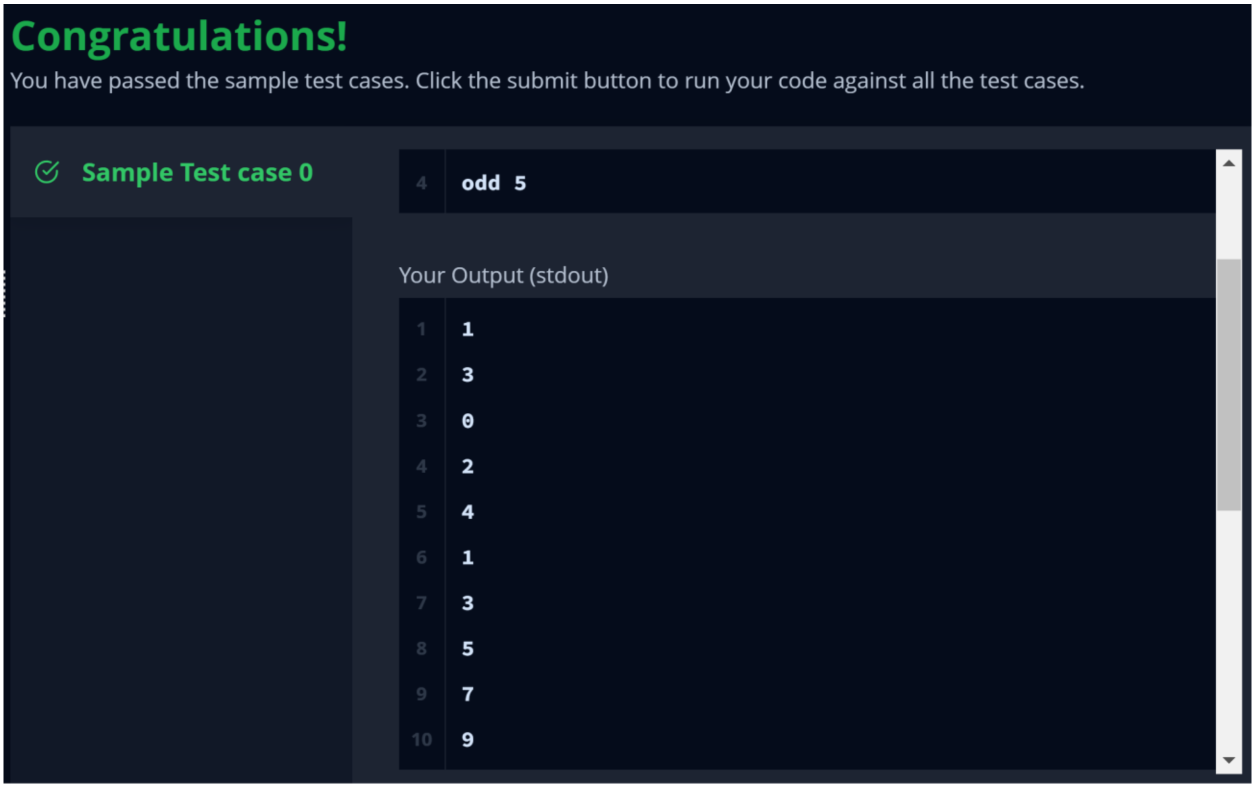
a. Code:

```python
class EvenStream(object):
    def __init__(self):
        self.current = 0

    def get_next(self):
        to_return = self.current
        self.current += 2
        return to_return

class OddStream(object):
    def __init__(self):
        self.current = 1

    def get_next(self):
        to_return = self.current
        self.current += 2
        return to_return


def print_from_stream(n, stream=None):
    if stream is None:
        stream = EvenStream()

    for _ in range(n):
        print(stream.get_next())
raw_input = input


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())
```

b. Screenshots:



```python
class EvenStream(object):
    def __init__(self):
        self.current = 0

    def get_next(self):
        to_return = self.current
        self.current += 2
        return to_return

class OddStream(object):
    def __init__(self):
        self.current = 1

    def get_next(self):
        to_return = self.current
        self.current += 2
        return to_return


def print_from_stream(n, stream=None):
    if stream is None:
        stream = EvenStream()

    for _ in range(n):
        print(stream.get_next())
raw_input = input


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())
```



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

```
odd 5
```

Your Output (stdout)

```
1
3
0
2
4
1
3
5
7
9
```

2 Hard Challenges

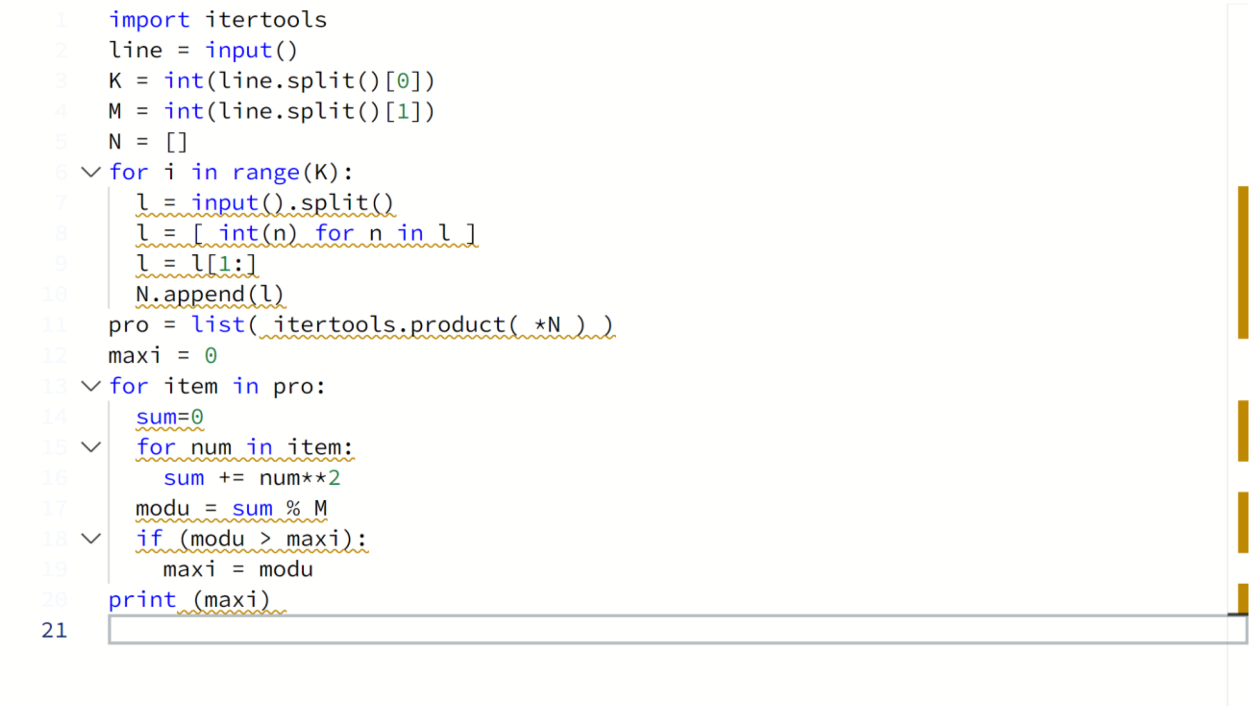
2.1 Maximize It!

a. Code:

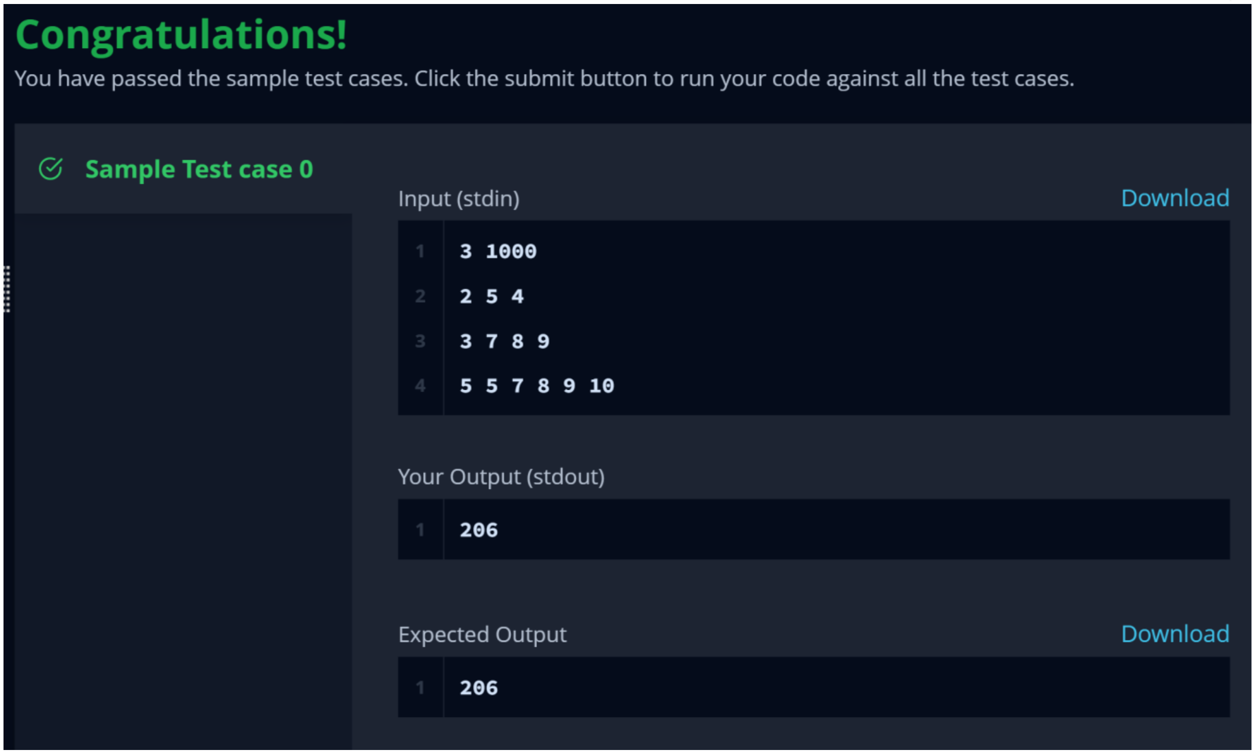
```
import itertools
line = input()
K = int(line.split()[0])
M = int(line.split()[1])
N = []
for i in range(K):
  l = input().split()
  l = [ int(n) for n in l ]
  l = l[1:]
  N.append(l)
pro = list( itertools.product( *N ) )
maxi = 0
for item in pro:
  sum=0
  for num in item:
    sum += num**2
  modu = sum % M
```

```
    if (modu > maxi):
      maxi = modu
print (maxi)
```

b. Screenshots:



```
import itertools
line = input()
K = int(line.split()[0])
M = int(line.split()[1])
N = []
for i in range(K):
    l = input().split()
    l = [ int(n) for n in l ]
    l = l[1:]
    N.append(l)
pro = list( itertools.product( *N ) )
maxi = 0
for item in pro:
    sum=0
    for num in item:
        sum += num**2
    modu = sum % M
    if (modu > maxi):
        maxi = modu
print (maxi)
```



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Input (stdin)

```
3 1000
2 5 4
3 7 8 9
5 5 7 8 9 10
```

Your Output (stdout)

```
206
```

Expected Output

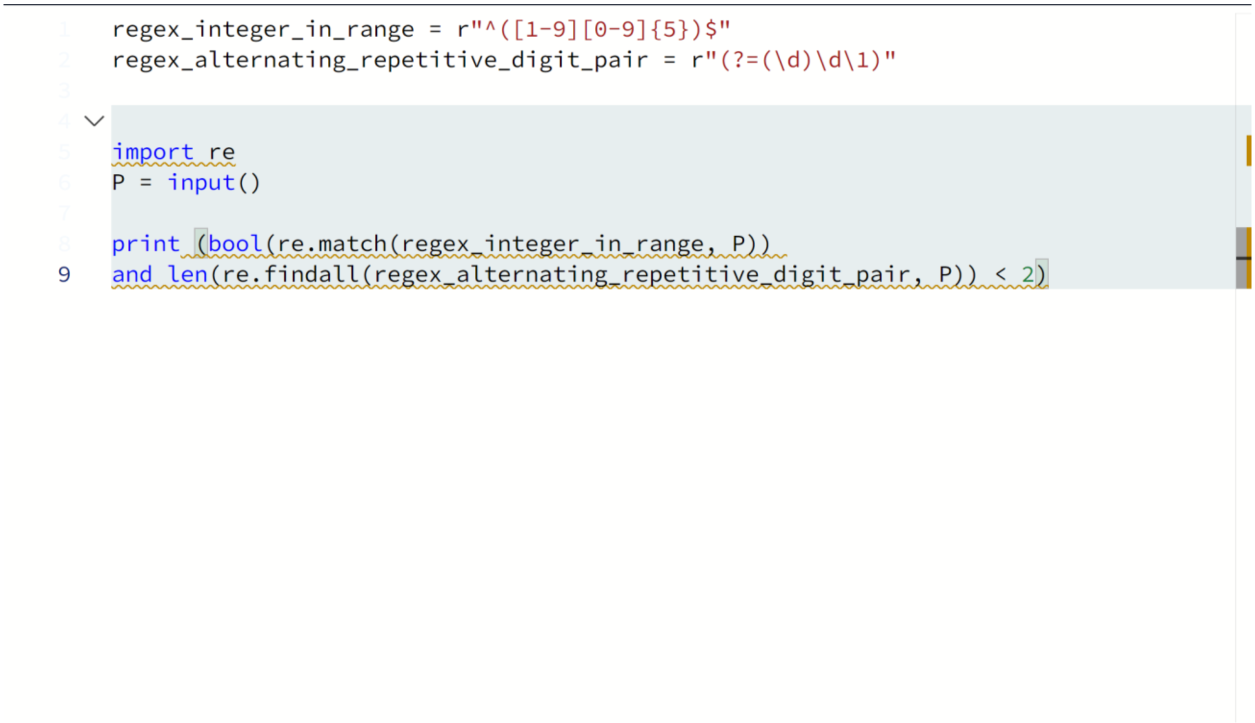
```
206
```

2.2 Validating Postal Codes

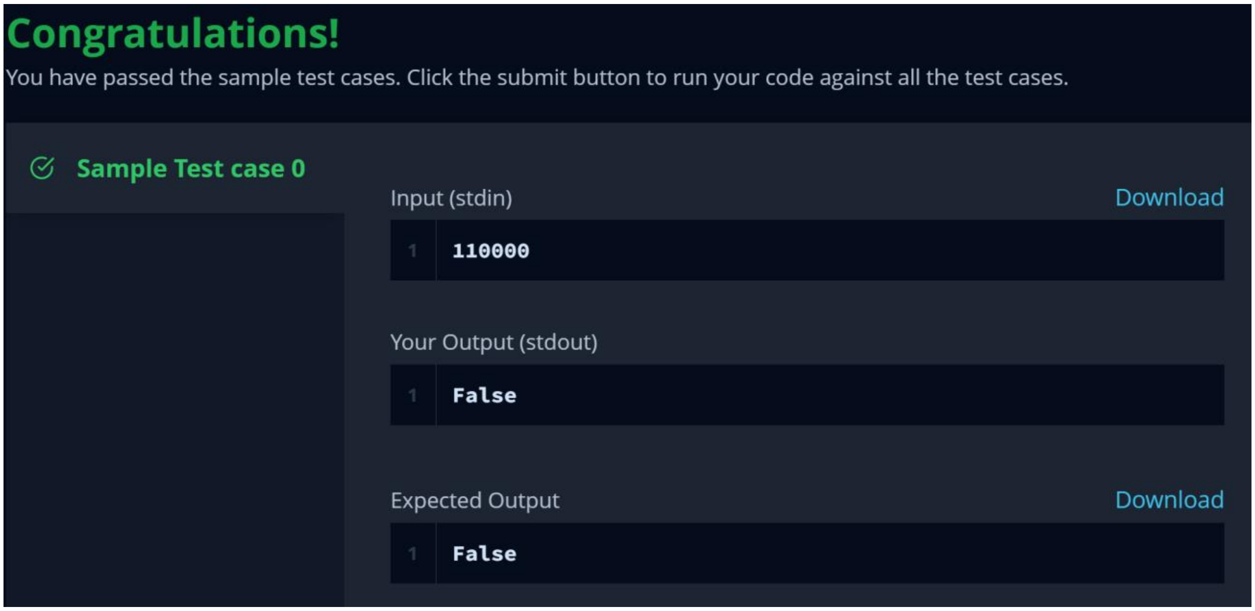
a. Code:

```
regex_integer_in_range = r"^([1-9][0-9]{5})$"
regex_alternating_repetitive_digit_pair = r"(?=(\d)\d\1)"

import re
P = input()

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

b. Screenshots:



```
regex_integer_in_range = r"^([1-9][0-9]{5})$"
regex_alternating_repetitive_digit_pair = r"(?=(\d)\d\1)"


import re
P = input()

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



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Input (stdin)

```
110000
```

Your Output (stdout)

```
False
```

Expected Output

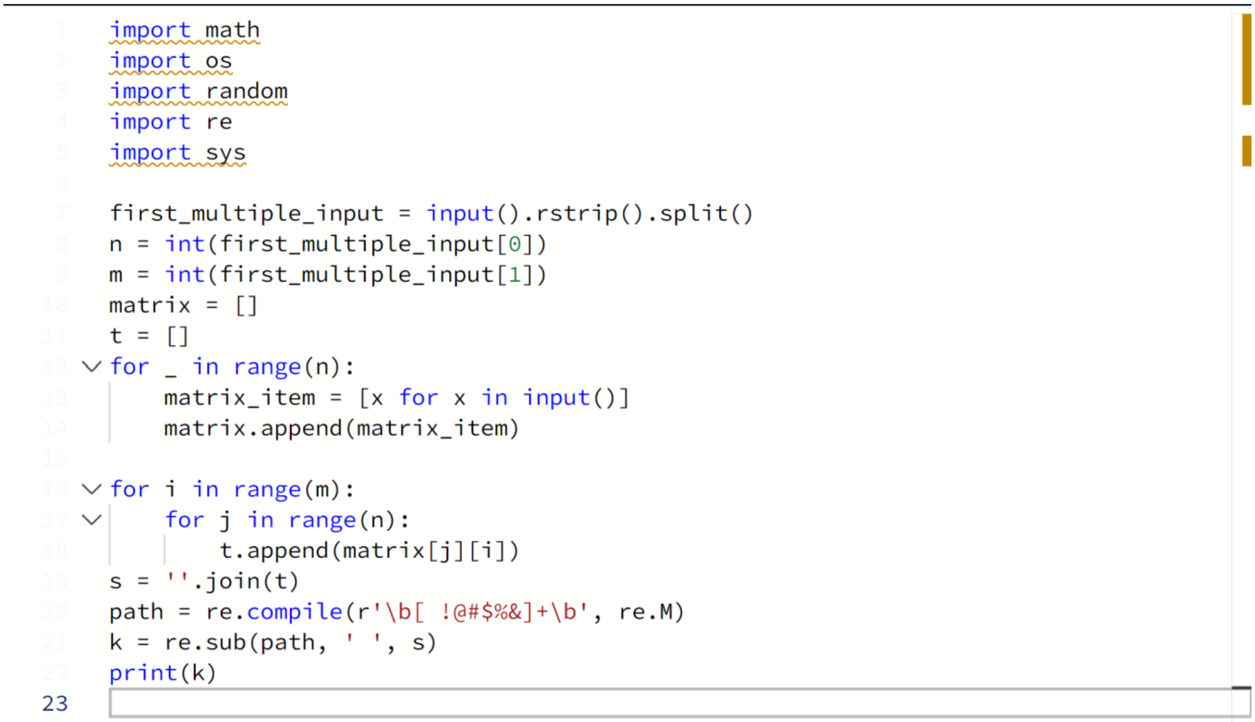
```
False
```

2.3 Matrix Script

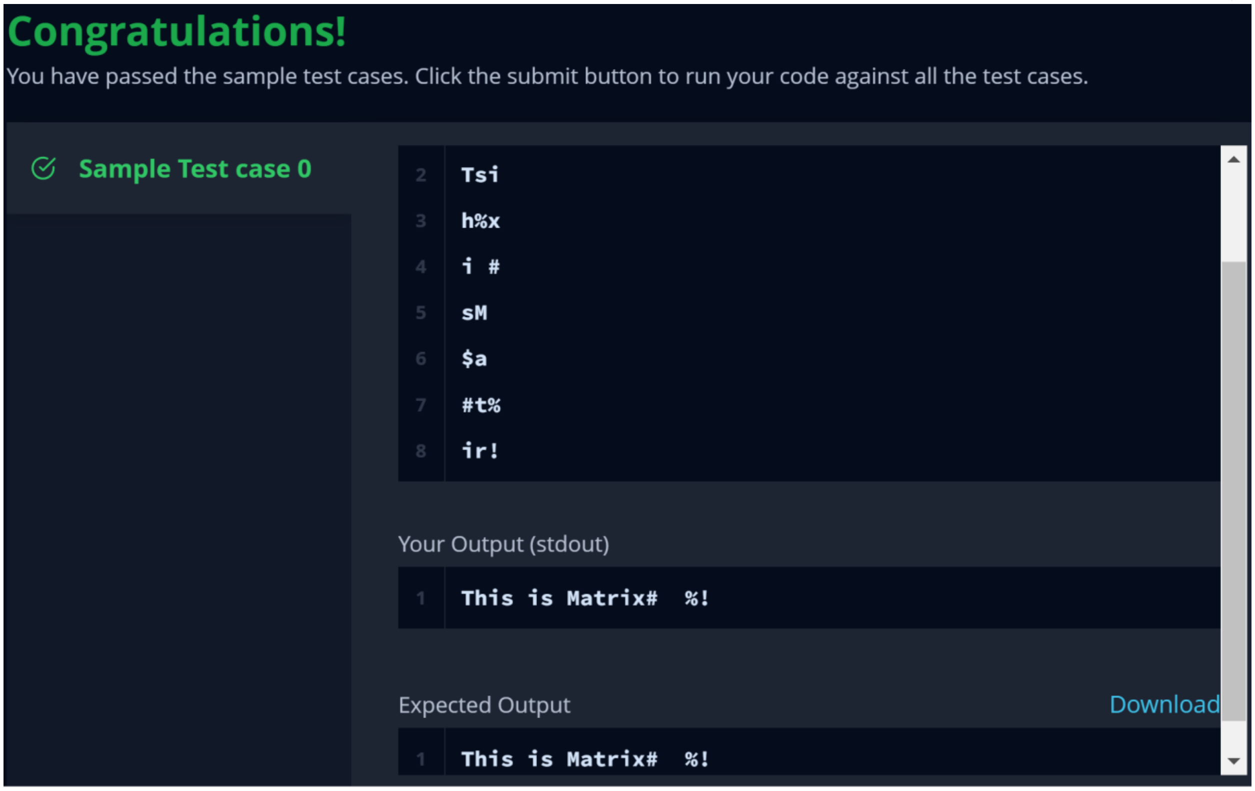
a. Code:

```
import math
import os
import random
import re
import sys

first_multiple_input = input().rstrip().split()
n = int(first_multiple_input[0])
m = int(first_multiple_input[1])
matrix = []
t = []
for _ in range(n):
    matrix_item = [x for x in input()]
    matrix.append(matrix_item)

for i in range(m):
    for j in range(n):
        t.append(matrix[j][i])
s = ''.join(t)
path = re.compile(r'\b[ !@#$%&]+\b', re.M)
k = re.sub(path, ' ', s)
print(k)
```

b. Screenshots:



```python
import math
import os
import random
import re
import sys

first_multiple_input = input().rstrip().split()
n = int(first_multiple_input[0])
m = int(first_multiple_input[1])
matrix = []
t = []
for _ in range(n):
    matrix_item = [x for x in input()]
    matrix.append(matrix_item)

for i in range(m):
    for j in range(n):
        t.append(matrix[j][i])
s = ''.join(t)
path = re.compile(r'\b[ !@#$%&]+\b', re.M)
k = re.sub(path, ' ', s)
print(k)

```



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

```
Tsi
h%x
i #
sM
$a
#t%
ir!
```

Your Output (stdout)

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

Expected Output

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

GitHub Repository Link: https://github.com/AinaZed/AinaZulfiqar_450192_AI_Assignment3/