Assignment # 3

Medium & Hard Challenges of Python on HackerRank

- **Name:**
  Syed Daniyal Hasan
- **Reg No.:**
  402137
- **Subject:**
  Artificial Intelligence
- **Submitted To:**
  Dr. Yasar Ayaz
- **Date:**
  January 4, 2024

➢ Solutions of Challenges:-

1) Default Arguments:

```python
class EvenStream(object):…
def print_from_stream(n, stream=EvenStream()):
    stream.__init__()
    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())
```

2) Write a function:

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

    # Write your logic here
    if (year % 400 == 0):
        return True
    if (year % 100 == 0):
        return leap
    if (year % 4 == 0):
        return True
    else:
        return False

    return leap

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

3) Minion Game:

```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)
```

4) Time Delta:

```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()
```

5) Find Angle MBC:

```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='')
```

6) No Idea!

```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)
```

7) Word Order

```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())
```

8) Compress the String

```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=' ')
```

9) Piling Up!

```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))
```

10) Triangle Quest 2

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

11) Iterables & Iterators

```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))
```

12) Triangle Quest

```python
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)
```

13) Classes: Dealing with Complex Numbers

```python
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):
        r = (self.real*no.real)-(self.imaginary*no.imaginary)
        i = (self.real*no.imaginary+no.real*self.imaginary)
        return Complex(r, i)
    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)
```

```python
    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')
```

14) Athlete Sort

```python
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)
```

15) ginortS

```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='')
```

16) Validating Email Address With a Filter

```python
def fun(email):
    try:
        username, url = email.split('@')
        website, extension = url.split('.')
    except ValueError:
        return False

    if username.replace('-', '').replace('_', '').isalnum() is False:
        return False
    elif website.isalnum() is False:
        return False
    elif len(extension) > 3:
        return False
    else:
        return True

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)
```

17) Regex Substitution

```python
# Enter your code here. Read input from STDIN. Print output to STDOUT
import re
for _ in range(int(input())):
    print(re.sub(r'(?<= )(&&|\|\|)(?= )', lambda x: 'and' if x.group() == '&&' else 'or', input()))
```

18) Validating Credit Card Numbers

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

19) Matrix Script

```python
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)
```

20) Merge the Tools!

```python
def merge_the_tools(string, k):
    # your code goes here
    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__':
    string, k = input(), int(input())
    merge_the_tools(string, k)
```

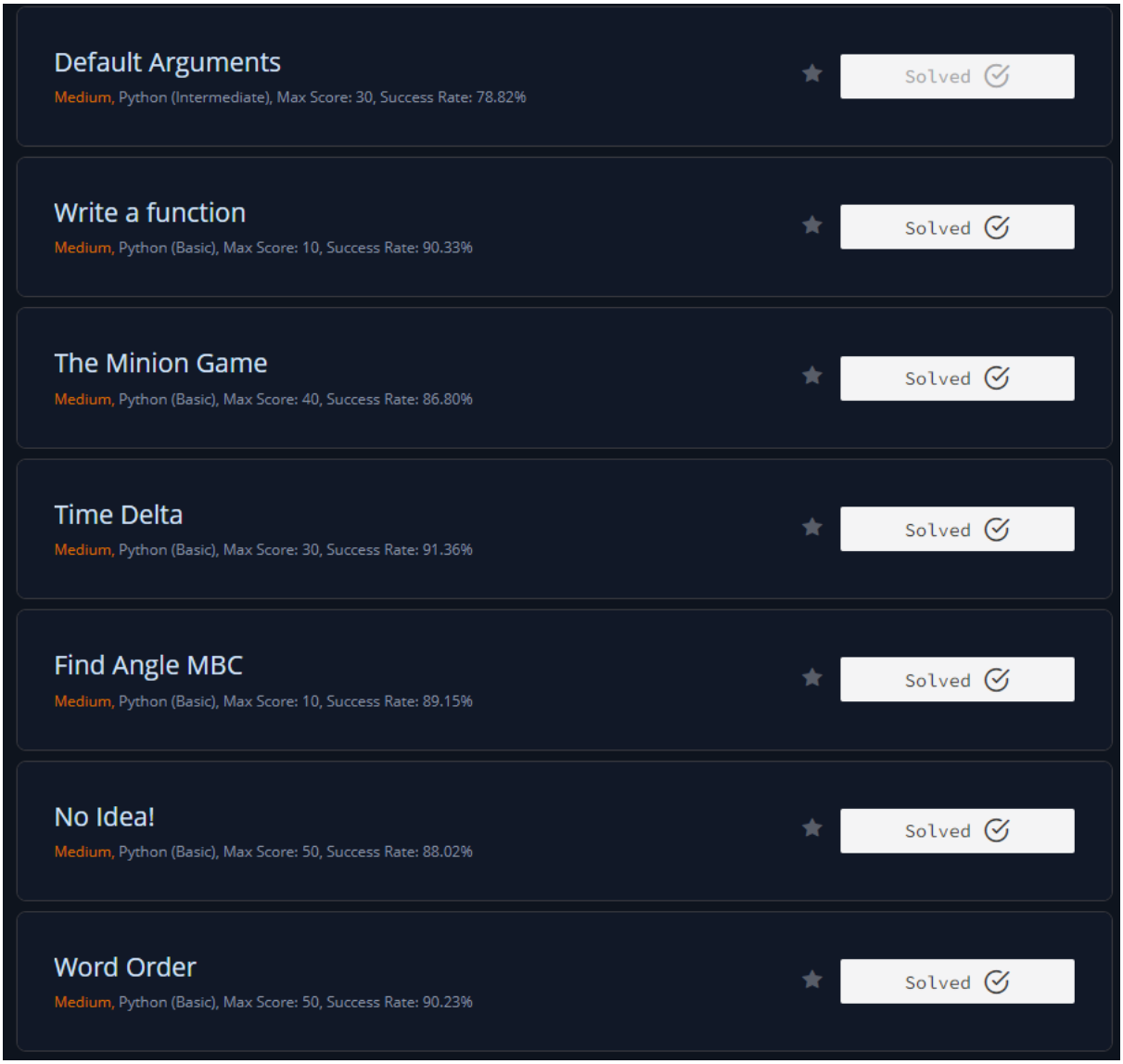
21) Company Logo

```python
from collections import Counter

S = input()
S = sorted(S)

FREQUENCY = Counter(list(S))

for k, v in FREQUENCY.most_common(3):
    print(k, v)

```

22) Maximize It!

```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))

```

Screenshots:



Compress the String!
Medium, Python (Basic), Max Score: 20, Success Rate: 97.15%
Solved

Piling Up!
Medium, Python (Basic), Max Score: 50, Success Rate: 90.64%
Solved

Triangle Quest 2
Medium, Python (Basic), Max Score: 20, Success Rate: 95.38%
Solved

Iterables and Iterators
Medium, Python (Basic), Max Score: 40, Success Rate: 96.60%
Solved

Triangle Quest
Medium, Python (Basic), Max Score: 20, Success Rate: 93.84%
Solved

Classes: Dealing with Complex Numbers
Medium, Python (Basic), Max Score: 20, Success Rate: 90.92%
Solved

Athlete Sort
Medium, Python (Basic), Max Score: 30, Success Rate: 95.53%
Solved

ginortS

Medium, Python (Basic), Max Score: 40, Success Rate: 97.63%

Solved

Validating Email Addresses With a Filter

Medium, Python (Basic), Max Score: 20, Success Rate: 90.82%

Solved

Regex Substitution

Medium, Python (Basic), Max Score: 20, Success Rate: 94.12%

Solved

Validating Credit Card Numbers

Medium, Python (Basic), Max Score: 40, Success Rate: 95.47%

Solved

Matrix Script

Hard, Problem Solving (Advanced), Max Score: 100, Success Rate: 89.97%

Solved

Merge the Tools!

Medium, Problem Solving (Basic), Max Score: 40, Success Rate: 93.75%

Solved

Company Logo

Medium, Problem Solving (Basic), Max Score: 30, Success Rate: 89.84%

Solved

Maximize It!

Hard, Problem Solving (Basic), Max Score: 50, Success Rate: 81.25%

Solved

All Required Challenges have been Solved: