Assignment 3 Muhammad Tayyab

Reg# 400628

1 Write a function

2

Change Theme Language Pypy 3

```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))
```

Line: 15 Col: 1

2 The minion game

Change Theme Language Pypy 3

```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)
```

Line: 13 Col: 1

3 Merge the Tool

Change Theme Language Pypy 3

```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)
```

Line: 14 Col: 38

4 Time Delta

Change Theme Language: Python 3

```python
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()
```

Line: 21 Col: 17

5 Find angle MBC

Change Theme Language Pypy 3

```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='')
```

Line: 13 Col: 55

6 No idea

Change Theme Language Pypy 3

```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)
```

Line: 15 Col: 21

7 Word order

Change Theme Language Pypy 3

```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())
```

Line: 13 Col: 23

8 Compress the string

Change Theme Language Pypy 3

```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 Company logo

Change Theme Language Pypy 3

```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)
```

10 Piling up

Change Theme Language Pypy 3

```python
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))
```

Line: 17 Col: 1

11 Triangular quest 2

Change Theme Language Pypy 3

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

Line: 2 Col: 27

12 Iterables & Iterators

```
# Enter your code here. Read input from STDIN. Print output to STDO
UT
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))
```

13 Triangular quest

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

14 Classes: dealing with complex number

```
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.imagi
nary)

    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/de
nom.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
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')
```

15 Athelete sort

```
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)
```

16. Ginortx

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

17. Validating Email address with a filter

```
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)
```

18. Reduce function

```
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)
```

19. Regrex substitution

```
# Enter your code here. Read input from STDIN. Print output to STDO
UT

import re
for _ in range(int(input())):
    print(re.sub(r'(?<= )(&&|\|\|)(?= )', lambda x: 'and' if x.grou
p() == '&&' else 'or', input()))
```

20. Validating Credit card number

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

21. Word score

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

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))
```

22. Default argument

```
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=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())
```

23. Maximize it

```
# Enter your code here. Read input from STDIN. Print output to STDO
UT
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))
```

24 Validating postal codes

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

25 Matrix script

```
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)
```

Write a function
Medium, Python (Basic), Max Score: 10, Success Rate: 90.33%
Solved

The Minion Game
Medium, Python (Basic), Max Score: 40, Success Rate: 86.80%
Solved

Merge the Tools!
Medium, Problem Solving (Basic), Max Score: 40, Success Rate: 93.76%
Solved

Time Delta
Medium, Python (Basic), Max Score: 30, Success Rate: 91.36%
Solved

Find Angle MBC
Medium, Python (Basic), Max Score: 10, Success Rate: 89.15%
Solved

No Idea!
Medium, Python (Basic), Max Score: 50, Success Rate: 88.03%
Solved

Word Order
Medium, Python (Basic), Max Score: 50, Success Rate: 90.24%
Solved

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

Company Logo
Medium, Problem Solving (Basic), Max Score: 30, Success Rate: 89.84%
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: 90.83%
Solved

Reduce Function
Medium, Max Score: 30, Success Rate: 98.37%
Solved

Regex Substitution
Medium, Python (Basic), Max Score: 20, Success Rate: 94.11%
Solved

Validating Credit Card Numbers
Medium, Python (Basic), Max Score: 40, Success Rate: 95.47%
Solved

Words Score
Medium, Max Score: 10, Success Rate: 94.94%
Solved

Default Arguments
Medium, Python (Intermediate), Max Score: 30, Success Rate: 78.83%
Solved

Maximize It!
Hard, Problem Solving (Basic), Max Score: 50, Success Rate: 81.27%
Solved

Validating Postal Codes
Hard, Max Score: 80, Success Rate: 87.40%
Solved

Matrix Script
Hard, Problem Solving (Advanced), Max Score: 100, Success Rate: 89.98%
Solved