ADM Homework 1

Xufeng Zhang, 2015060

zhangxufeng1998@gmail.com

Problem 1

Introduction (total 7)

In [ ]:

```
# 1 say hello world in python

print("Hello, World!")
```

In [ ]:

```
# 2 Python If-Else

import math
import os
import random
import re
import sys

if __name__ == '__main__':
    n = int(input().strip())
    if n%2 == 1:
        print("Weird")
    elif 6<= n <= 20:
        print("Weird")
    else:
        print("Not Weird")
```

In [ ]:

```
# 3 Arithmetic Operators

if __name__ == '__main__':
    a = int(input())
    b = int(input())
    print(a+b)
    print(a-b)
    print(a*b)
```

In [ ]:

```
# 4 Python: Division

if __name__ == '__main__':
    a = int(input())
    b = int(input())
    c=int(a/b)
    print(c)
    print(a/b)
```

In [ ]:

```
# 5 Loops

if __name__ == '__main__':
    n = int(input())
    if n < 0:
        print("error!")
    else:
        for i in range(0,n):
            a=i*i
            print(a)
```

In [ ]:

```
# 6 Write a Function

def is_leap(year):
    leap = False
    if year%400 == 0:
        leap = True
    elif year%100 == 0:
        leap = False
    elif year%4 == 0:
        leap = True
    return leap
```

In [ ]:

```
# 7 Print Function

if __name__ == '__main__':
    n = int(input())
    str1=""
    for i in range(1,n+1):
        stri = str(i)
        str1 = str1 + stri
    print(str1)
```

Data types (total:6)

In [ ]:

```
# 1 List Comprehensions

if __name__ == '__main__':
    x = int(input())
    y = int(input())
    z = int(input())
    n = int(input())
    print("[",end="")
    for i in range(0,x+1):
        for j in range(0,y+1):
            for k in range (0,z+1):
                if i+j+k != n:
                    print([i,j,k],end='')
                    if i+j+k != x+y+z:
                        print(", ",end='')
    print("]")
```

In [ ]:

```
# 2 Find the Runner-Up Score!

if __name__ == '__main__':
    n = int(input())
    arr = map(int, input().split())
    arr=list(arr)
    sortlist = sorted(arr)
    arrmax = sortlist[-1]
    smax = sortlist[0]
    if n >10:
        n = 10
    for i in range(0,n):
        if (arr[i] < arrmax) and (arr[i]>=smax):
            smax = arr[i]
    print(smax)
```

In [ ]:

```
# 3 Nested Lists

n = int(input())
mp = {}
L = set([])
st = []
for i in range(n):
    name = input()
    score = float(input())
    mp[name] = score
    L.add(score)
L2 = sorted(L)
u = L2[1]
for k, v in mp.items():
    if v == u:
        st.append(k)
st2 = sorted(st, key=str.lower)
for a in st2:
    print(a)
```

In [ ]:

```
# 4 Finding the percentage

if __name__ == '__main__':
    n = int(input())
    student_marks = {}
    for _ in range(n):
        name, *line = input().split()
        scores = list(map(float, line))
        student_marks[name] = scores
    query_name = input()

    query=student_marks[query_name]
    query=list(query)
    sum=0
    for i in range(0,len(query)):
        sum=sum+query[i]
    aver=sum/len(query)
    print(format(aver,'.2f'))
```

In [ ]:

```
# 5 Lists

if __name__ == '__main__':
    N = int(input())
    L = []
    for i in range(N):
        s = input().split()
        dem = s[0]
        args = s[1:]
        if dem != 'print':
            eval('L.' + dem + '(' + ', '.join(args) + ')')
        else:
            print(L)
```

In [ ]:

```
# 6 Tuples

input()
xtuple=map(int,input().strip().split(" "))
xtuple=tuple(xtuple)
a=hash(xtuple)
print(a)
```

Strings (total 14)

In [ ]:

```
# 1 aWAP cASE

def swap_case(s):
    answer = ""
    for i in s:
        if i == i.upper():
            answer = answer + i.lower()
        else:
            answer = answer + i.upper()
    return answer
    return

if __name__ == '__main__':
    s = input()
    result = swap_case(s)
    print(result)
```

In [ ]:

```
# 2 String Split and Join

def split_and_join(line):
    line = line.split(" ")
    line = "-".join(line)
    return line

if __name__ == '__main__':
    line = input()
    result = split_and_join(line)
    print(result)
```

In [ ]:

```
# 3 What's your name?

# Complete the 'print_full_name' function below.
#
# The function is expected to return a STRING.
# The function accepts following parameters:
#  1. STRING first
#  2. STRING last
#

def print_full_name(first, last):
    print("Hello ",end="")
    print(first,end="")
    print(" ",end="")
    print(last,end="")
    print("! You just delved into python.")
if __name__ == '__main__':
    first_name = input()
    last_name = input()
    print_full_name(first_name, last_name)
```

In [ ]:

```
# 4 Mutations

def mutate_string(s,i,c):
    string = s[:i] + c + s[i+1:]
    return string

if __name__ == '__main__':
    s = input()
    i, c = input().split()
    s_new = mutate_string(s, int(i), c)
    print(s_new)
```

In [ ]:

```
# 5 Find a String

def count_substring(string, sub_string):
    sublen = len(sub_string)
    sum=0
    for i in range(0,len(string)):
        if string[i] == sub_string[0]:
            if string[i:sublen+i] == sub_string:
                sum = sum+1
    return sum

if __name__ == '__main__':
    string = input().strip()
    sub_string = input().strip()

    count = count_substring(string, sub_string)
    print(count)
```

In [ ]:

```
# 6 String Validators

if __name__ == '__main__':
    s = input()
    line = list(s)
    a,b,c,d,e=False,False,False,False,False
    for i in line:
        if i.isalnum():
            a=True
        if i.isalpha():
            b=True
        if i.isdigit():
            c=True
        if i.islower():
            d=True
        if i.isupper():
            e=True

    print(a)
    print(b)
    print(c)
    print(d)
    print(e)
```

In [ ]:

```
# 7 Text Alignment

#Replace all ______ with rjust, ljust or center.

thickness = int(input()) #This must be an odd number
c = 'H'

#Top Cone
for i in range(thickness):
    print((c*i).rjust(thickness-1)+c+(c*i).ljust(thickness-1))

#Top Pillars
for i in range(thickness+1):
    print((c*thickness).center(thickness*2)+(c*thickness).center(thickness*6))

#Middle Belt
for i in range((thickness+1)//2):
    print((c*thickness*5).center(thickness*6))

#Bottom Pillars
for i in range(thickness+1):
    print((c*thickness).center(thickness*2)+(c*thickness).center(thickness*6))

#Bottom Cone
for i in range(thickness):
    print(((c*(thickness-i-1)).rjust(thickness)+c+(c*(thickness-i-1)).ljust(thickness)).rjust(th
ickness*6))
```

In [ ]:

```
# 8 Text Wrap

import textwrap
def wrap(string, max_width):
    last = len(string)%max_width
    nub = int(len(string)/max_width)
    cunt=0
    for i in range(1,nub+2):
        string = string[:i*max_width+cunt]+'\n'+string[i*max_width+cunt:]
        cunt=cunt+1
    return string

if __name__ == '__main__':
    string, max_width = input(), int(input())
    result = wrap(string, max_width)
    print(result)
```

In [ ]:

```
# 9 Designer Door Mat

# Enter your code here. Read input from STDIN. Print output to STDOUT
N, M = map(int, input().split(" "))
symb='.|.'
for i in range(1,N,2):
    print((i*symb).center(M,'-'))

print('WELCOME'.center(M,'-'))

for i in range(N-2,-1,-2):
    print((i*symb).center(M, '-'))
```

In [ ]:

```
# 10 String Formatting

def print_formatted(number):
    l=len(bin(number)[2:])
    for i in range(1, number + 1):
        floatVar = str(i)
        octVar = str(oct(i)[2:])
        hexVar = str(hex(i)[2:]).upper()
        binVar = str(bin(i)[2:])
        print(floatVar.rjust(l)+' '+octVar.rjust(l)+' '+hexVar.upper().rjust(l)+' '+binVar.rjust
(l))

if __name__ == '__main__':
    n = int(input())
    print_formatted(n)
```

In [ ]:

```
# 11 Alphabet Rangoli

# Learn it from discussion part
# main part created by roger_eppich
def print_rangoli(size):
    Str = 'abcdefghijklmnopqrstuvwxyz'[0:size]

    for i in range(size-1, -size, -1):
        x = abs(i)
        if x >= 0:
            line = Str[size:x:-1]+Str[x:size]
            print ("--"*x+ '-'.join(line)+"--"*x)

if __name__ == '__main__':
    n = int(input())
    print_rangoli(n)
```

In [ ]:

```
# 12 Capitalize

#!/bin/python3

import math
import os
import random
import re
import sys

def solve(s):
    s = s.split(" ")
    for i in range(0,len(s)):
        a = s[i].capitalize()
        s[i] = a
    answer = (" ".join(s))
    return answer

if __name__ == '__main__':
    fptr = open(os.environ['OUTPUT_PATH'], 'w')

    s = input()

    result = solve(s)

    fptr.write(result + '\n')

    fptr.close()
```

In [ ]:

```
# 13 The Minion Game

#learn it from discussion part
def minion_game(string):

    vowels = 'AEIOU'

    kevsc = 0
    stusc = 0
    for i in range(len(string)):
        if s[i] in vowels:
            kevsc += (len(string)-i)
        else:
            stusc += (len(string)-i)

    if kevsc > stusc:
        print("Kevin", kevsc)
    elif kevsc < stusc:
        print("Stuart", stusc)
    else:
        print("Draw")

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

In [ ]:

```
# 14 Merge the Tools!

def merge_the_tools(string, k):
    for i in range(0, len(string), k):
        s = ""
        for j in string[i:i+k]:
            if j in s:
                continue
            else:
                s += j
        print(s)

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

Sets (total 13)

In [ ]:

```
# 1 Introduction to set

# learn it from discussion part
def average(array):
    # your code goes here
    return sum(set(array))/len(set(array))

if __name__ == '__main__':
    n = int(input())
    arr = list(map(int, input().split()))
    result = average(arr)
    print(result)
```

In [ ]:

```
# 2 No idea!

a=input().split(' ')
l=input().split(' ')
firstset=set([int(i) for i in input().split(' ')])
secondset=set([int(i) for i in input().split(' ')])
for j in range(len(l)):
    l[j]=int(l[j])

score=0
for i in l:
    if i in firstset:
        score+=1
    if i in secondset:
        score-=1
print(score)
```

In [ ]:

```
# 3 Symmetric Difference

# Enter your code here. Read input from STDIN. Print output to STDOUT
a,b=(int(input()),input().split())
c,d=(int(input()),input().split())
x=set(b)
y=set(d)
answer = x^y
answer = list(answer)
for i in range(len(answer)):
    answer[i]=int(answer[i])
answer = sorted(answer)
for j in range(len(answer)):
    print(answer[j])
```

In [ ]:

```
# 4 Set.add()

# Enter your code here. Read input from STDIN. Print output to STDOUT
num = int(input())
dist = set()
for i in range(int(num)):
    dist.add(input())

print(len(dist))
```

In [ ]:

```
# 5 Set .discard(), .remove() & .pop()

n = int(input())
s = set(map(int, input().split()))
l = int(input())

for i in range(l) :
    cmd=input().split()
    if cmd[0]=="pop" :
        s.pop()
    elif cmd[0]=="remove" :
        s.remove(int(cmd[1]))
    elif cmd[0]=="discard" :
        s.discard(int(cmd[1]))

print (sum(s))
```

In [ ]:

```
# 6 Set .union() Operation

n = input()
set1 = set(map(int, input().split()))
b = input()
set2 = set(map(int, input().split()))

set3 = set1|set2
print(len(set3))
```

In [ ]:

```
# 7 Set .intersection() Operation

n = input()
set1 = set(map(int, input().split()))
b = input()
set2 = set(map(int, input().split()))

set3 = set1 & set2
print(len(set3))
```

In [ ]:

```
# 8 Set .difference() Operation

n = input()
set1 = set(map(int, input().split()))
b = input()
set2 = set(map(int, input().split()))

set3 = set1 - set2
print(len(set3))
```

In [ ]:

```
# 9 Set .symmetric_difference() Operation

n = input()
set1 = set(map(int, input().split()))
b = input()
set2 = set(map(int, input().split()))

set3 = set1^set2
print(len(set3))
```

In [ ]:

```
# 10 Set Mutations

n = int(input())
s1 = set(map(int, input().split()))
N = int(input())

for _ in range(N):
    cmd = input().split()
    s2 = set(map(int, input().split()))
    if(cmd[0] == "intersection_update"):
        s1.intersection_update(s2)
    elif(cmd[0] == "update"):
        s1.update(s2)
    elif(cmd[0] == "symmetric_difference_update"):
        s1.symmetric_difference_update(s2)
    elif(cmd[0] == "difference_update"):
        s1.difference_update(s2)

print(sum(s1))
```

In [ ]:

```
# 11 The Captain's Room

k,arr = int(input()),list(map(int, input().split()))

sum1=sum(arr)
set1=set(arr)
sum2=sum(set1)*k

nroom=sum1-sum2
room=nroom/(len(arr)-len(set1)*k)
print(int(room))
```

In [ ]:

```
# 12 Check Subset

n = int(input())
for i in range(n):
    x=input()
    a=set(input().split())
    y=input()
    b=set(input().split())
    print(a.issubset(b))
```

In [ ]:

```
# 13 Check Strict Superset

A = list(map(int, input().split()))
n = int(input())
L = [list(map(int, input().split())) for _ in range(n)]

output = False

for i in L:

    if set(i).issubset(set(A)) and len(i) < len(A):
        output = True
    else:
        output = False
        break

print(output)
```

Collections (total 8)

In [ ]:

```
# 1 collections.Counter()

# Enter your code here. Read input from STDIN. Print output to STDOUT
import collections

n = int(input())
shoes = collections.Counter(map(int,input().split()))
nc = int(input())

total = 0

for i in range(nc):
    size, price = map(int,input().split())
    if shoes[size]:
        total += price
        shoes[size] -= 1

print(total)
```

In [ ]:

```
# 2 DefaultDict Tutorial

# Enter your code here. Read input from STDIN. Print output to STDOUT
from collections import defaultdict
d = defaultdict(list)

n, m = map(int,input().split())

for i in range(1,n+1):
    d[input()].append(str(i))

for i in range(m):
    print (' '.join(d[input()]) or -1)
```

In [ ]:

```
# 3 Collections.namedtuple()

# Enter your code here. Read input from STDIN. Print output to STDOUT
from collections import namedtuple

n = int(input())
a = input()

total = 0

Student = namedtuple('Student', a)
for i in range(n):
    student = Student(*input().split())
    total += int(student.MARKS)

aver = total/n
print(format(aver,'.2f'))
```

In [ ]:

```
# 4 Collections.OrderedDict()

from collections import OrderedDict

d = OrderedDict()

for i in range(int(input())):
    item, space, quantity = input().rpartition(' ')
    d[item] = d.get(item, 0) + int(quantity)

for item, quantity in d.items():
    print(item, quantity)
```

In [ ]:

```
# 5 Word Order

# Enter your code here. Read input from STDIN. Print output to STDOUT
from collections import Counter, OrderedDict

words = OrderedDict()

for i in range(int(input())):
    word = input()
    words.setdefault(word, 0)
    words[word] += 1

print(len(words))
print(*words.values()) # for format of the output
```

In [ ]:

```
# 6 Collections.deque()

from collections import deque

d = deque()

n=int(input())

for i in range(n):
    x=input().split()
    if x[0] == 'pop': d.pop()
    if x[0] == 'popleft': d.popleft()
    if x[0] == 'append': d.append(int(x[1]))
    if x[0] == 'appendleft': d.appendleft(int(x[1]))


print(*d)
```

In [ ]:

```
# 7 Company Logo


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


if __name__ == '__main__':
    s = input()
    d = Counter(sorted(s))
    for i in d.most_common(3):
        print(*i)
```

In [ ]:

```
# 8 Piling Up!

# learned it from discussion part
for t in range(int(input())):
    l = int(input())
    lst = list(map(int,input().split()))
    i = 0
    while i < l - 1 and lst[i] >= lst[i+1]:
        i += 1
    while i < l - 1 and lst[i] <= lst[i+1]:
        i += 1

    if i == l-1:
        print("Yes")
    else:
        print("No")
```

Data and Time (total 2)

In [ ]:

```
# 1 Calendar Module

import calendar
m, d, y = map(int, input().split())
ans = calendar.day_name[calendar.weekday(y, m, d)].upper()
print(ans)
```

In [ ]:

```
# 2 Time Data

# learned it from discussion part
from datetime import datetime as dt
fmt = '%a %d %b %Y %H:%M:%S %z'
for i in range(int(input())):
    print(int(abs((dt.strptime(input(), fmt) -
                   dt.strptime(input(), fmt)).total_seconds())))
```

Exception (only 1)

In [ ]:

```
for i in range(int(input())):
    try:
        a,b=map(int,input().split())
        print(a//b)
    except Exception as e:
        print("Error Code:",e)
```

Bulit-ins (total 3)

In [ ]:

```
# 1 zipped

co,line = input().split()
line=int(line)
co=int(co)
arr=[]
for i in range(line):
    arr.append(list(map(float,input().split())))

ans=[]

for k in range(co):
    s=0
    for j in range(line):
        subarr=arr[j]
        s = s + subarr[k]

    aver = s/line
    ans.append(aver)


for _ in range(len(ans)):
    print(ans[_])
```

In [ ]:

```
# 2 Athlete Sort

# I learned if from discussion part
n_m = list(map(int, input().split()))

table = list()

for i in range(n_m[0]):
    table.append(list(map(int, input().split())))

k = int(input())
sorted_table = sorted(table, key=lambda record: record[k])
for item in sorted_table:
    print(*item)
```

In [ ]:

```
# 3 ginortS

import re

patterns = [ r'[a-z]', r'[A-Z]' , r'[13579]'  ,r'[02468]']
s = input()
ss = [  r for pattern in patterns for r in sorted(re.findall( pattern, s ))  ]
print("".join(ss))
```

Python Functionals (only 1)

In [ ]:

```
cube = lambda x:pow(x,3) # complete the lambda function

def fibonacci(n):
    lis = [0,1]
    for i in range(2,n):
         lis.append(lis[i-2] + lis[i-1])
    return(lis[0:n])

if __name__ == '__main__':
    n = int(input())
    print(list(map(cube, fibonacci(n))))
```

Regex and Parsing challenges (total 17)

In [ ]:

```
# 1 Detect Floating Point Number

import re
m = int(input())
for i in range(0,m):
    temp =input()
    print(bool(re.match(r"^[-+]?[0-9]*\.[0-9]+$",temp)))
```

In [ ]:

```
# 2 Re.split()

regex_pattern = r"[.,]" # Do not delete 'r'.

import re
print("\n".join(re.split(regex_pattern, input())))
```

In [ ]:

```
# 3 Group(), Groups() & Groupdict()

import re
m = re.search(r"([a-z0-9])\1+", input())
print(m.group(1) if m else -1)
```

In [ ]:

```
# 4 Re.findall() & Re.finditer()

# learn it form disscussion part
import re
v = "aeiou"
c = "qwrtypsdfghjklzxcvbnm"
print('\n'.join(re.findall(r'(?<=[%s])([%s]{2,})[%s]'%(c,v,c),input(),re.I) or ['-1']))
```

In [ ]:

```
# 5 Re.start() & Re.end()

# Enter your code here. Read input from STDIN. Print output to STDOUT
import re
s = input()
k = input()
matches = list(re.finditer(r'(?={})'.format(k), s)) # It is more convient for us to use re.findi
ter to solve this problem
if matches:
    print('\n'.join(str((match.start(),match.start() + len(k) - 1)) for match in matches))
else:
    print('(-1, -1)')
```

In [ ]:

```
# 6 Regex Substitution

# learn it from discussion part
for _ in range(int(input())):
    line = input()

    while ' && ' in line or ' || ' in line:
        line = line.replace(" && ", " and ").replace(" || ", " or ")

    print(line)
```

In [ ]:

```
# 7 Validating Roman Numerals

regex_pattern = r"M{,3}(CM|CD|D?C{,3})(XC|XL|L?X{,3})(IX|IV|V?I{,3})$"  # Do not delete 'r'.

import re
print(str(bool(re.match(regex_pattern, input()))))
```

In [ ]:

```
# 8 Validating phone numbers

n = int(input())
for i in range(n):
    line = input()
    if len(line)==10 :
        if  line.isnumeric():
            if (int(line[0])==7 or int(line[0])==8 or int(line[0])==9):
                print('YES')
            else:
                print('NO')
        else:
            print('NO')
    else:
        print('NO')
```

In [ ]:

```
# 9 Validating and Parsing Email Addresses

import re

pattern = re.compile(r'<[A-Za-z](\w|-|\.|_)+@[A-Za-z]+\.[A-Za-z]{1,3}>')
n = int(input())
for i in range(n):
    x,y = input().split(' ')
    m = re.match(pattern,y)
    if m:
        print(x,y)
```

In [ ]:

```
# 10 Hex Color Code

import re
for i in range(int(input())):
    matches = re.findall(r':?.(#[0-9a-fA-F]{6}|#[0-9a-fA-F]{3})', input())
    if matches:
        print(*matches, sep='\n') #* for the output format
```

In [ ]:

```
# 11 HTML Parser - Part 1

# this question is hard, I learned it from discussion part.
from html.parser import HTMLParser
class MyHTMLParser(HTMLParser):
    def handle_starttag(self, tag, attrs):
        print("Start :", tag)
        for item in attrs:
            print('->',item[0],'>', item[1])
    def handle_endtag(self, tag):
        print("End   :", tag)
    def handle_startendtag(self, tag, attrs):
        print("Empty :", tag)
        for ele in attrs:
            print('->',ele[0],'>', ele[1])
n = int(input())
answer = MyHTMLParser()
for i in range(n):
    line = input()
    answer.feed(line)
```

In [ ]:

```
# 12 HTML Parser - Part 2

from html.parser import HTMLParser
# learned it from disscussion part

class MyHTMLParser(HTMLParser):
  def handle_comment(self, data):
    if '\n' in data:
        print('>>> Multi-line Comment')
        print(data)
    else:
        print('>>> Single-line Comment')
        print(data)

  def handle_data(self, data):
    if '\n' not in data:
        print(">>> Data")
        print(data)


html = ''
for i in range(int(input())):
    html += input().rstrip()
    html += '\n'

parser = MyHTMLParser()
parser.feed(html)
parser.close()
```

In [ ]:

```
# 13 Detect HTML Tags, Attributes and Attribute Values

from html.parser import HTMLParser

class MyHTMLParser(HTMLParser):
    def handle_starttag(self, tag, attrs):
        print(tag)
        [print('-> {} > {}'.format(*attr)) for attr in attrs]

html = '\n'.join([input() for _ in range(int(input()))])

parser = MyHTMLParser()
parser.feed(html)
parser.close()
```

In [ ]:

```
# 14 Validating UID

import re

a = r"(?!.*(.).*\1)"
b = r"(?=(?:.*[A-Z]){2,})"
c = r"(?=(?:.*\d){3,})"
d = r"[a-zA-Z0-9]{10}"
filter = a,b,c,d
for uid in [input() for i in range(int(input()))]:
    if all([re.match(f, uid) for f in filter]):
        print("Valid")
    else:
        print("Invalid")
```

In [ ]:

```
# 15 Validating Credit Card Numbers

import re

for i in range(int(input())):
    s = input()
    if re.match(r"^[456]([\d]{15}|[\d]{3}(-[\d]{4}){3})$", s) and not re.search(r"([\d])\1\1
\1", s.replace("-", "")):

        print("Valid")
    else:
        print("Invalid")
```

In [ ]:

```
# 16 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 not delete 'r'.te '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)
```

In [ ]:

```
# 17 Matrix Script

import re

n, m = map(int, input().split())
matrix, s = [], ''
for _ in range(n):
    matrix_item = input()
    matrix.append(matrix_item)

for i in range(m):
    for j in range(n):
        s += matrix[j][i]

s=re.sub(r'(\w)\W{1,}(\w)', r'\1 \2', s)
print(s)
```

XML (total 2)

In [ ]:

```
# 1 Find the Score

import sys
import xml.etree.ElementTree as etree

# learn it from discussion
def get_attr_number(node):
    return sum([len(elem.items()) for elem in tree.iter()])

if __name__ == '__main__':
    sys.stdin.readline()
    xml = sys.stdin.read()
    tree = etree.ElementTree(etree.fromstring(xml))
    root = tree.getroot()
    print(get_attr_number(root))
```

In [ ]:

```
# 2 Find the Maximum Depth

import xml.etree.ElementTree as etree

maxdepth = 0
def depth(elem, level):
    global maxdepth
    if (level == maxdepth):
        maxdepth = maxdepth + 1

    for item in elem:
        depth(item, level + 1)
    # your code goes here

if __name__ == '__main__':
    n = int(input())
    xml = ""
    for i in range(n):
        xml =  xml + input() + "\n"
    tree = etree.ElementTree(etree.fromstring(xml))
    depth(tree.getroot(), -1)
    print(maxdepth)
```

Closures and Decorations (total 2)

In [ ]:

```
# 1 Standardize Mobile Number Using Decorators

def wrapper(f):
    def fun(l):
        fun = f(["+91 "+c[-10:-5]+" "+c[-5:] for c in l])
    return fun

@wrapper
def sort_phone(l):
    print(*sorted(l), sep='\n')

if __name__ == '__main__':
    l = [input() for _ in range(int(input()))]
    sort_phone(l)
```

In [ ]:

```
# 2 Decorators 2 - Name Directory

def person_lister(f):
    def inner(people):
        for i in sorted(people, key=lambda x: int(x[2])):
            yield f(i)
    return inner

@person_lister
def name_format(person):
    return ("Mr. " if person[3] == "M" else "Ms. ") + person[0] + " " + person[1]

if __name__ == '__main__':
    people = [input().split() for i in range(int(input()))]
    print(*name_format(people), sep='\n')
```

Numpy (total 15)

In [ ]:

```
# 1 Arrays

import numpy

def arrays(arr):
    arr=numpy.array(arr,float)
    return arr[::-1]

arr = input().strip().split(' ')
result = arrays(arr)
print(result)
```

In [ ]:

```
# 2 Shape and Reshape

import numpy as np
a=np.array(list(map(int,input().split())))
a.shape=(3,3)
print(a)
```

In [ ]:

```
# 3 Transpose and Flatten

import numpy as np

n , m = input().split()
a = []
for i in range(int(n)):
    my_array = input().split()
    a.append(my_array)
arr = np.array(a,int)
print(np.transpose(arr))
print(arr.flatten())
```

In [ ]:

```
# 4 Concatenate

import numpy as np
N, M, P = map(int,input().split())
a=[]
for i in range(N+M):
    list1 = list(map(int,input().split()))
    a.append(list1)

a = np.array(a)
print(a)
```

In [ ]:

```
# 5 Zeros and Ones

import numpy

d = list(map(int,(input().split())))

arr1 = numpy.zeros((d), dtype = numpy.int)
print(arr1)

arr2 = numpy.ones((d), dtype = numpy.int)
print(arr2)
```

In [ ]:

```
# 6 Eyes and Identity

# learn it from wi34rd in discussion
# what I wrote make some mistakes in sapce.
import numpy

numpy.set_printoptions(sign=' ')
print(numpy.eye(*map(int, input().split())))
```

In [ ]:

```
# 7 Array Mathematics

import numpy as np

n, m = map(int, input().split())
a, b = (np.array([input().split() for _ in range(n)], dtype=int) for _ in range(2))

print(a+b)
print(a-b)
print(a*b)
print(a//b)
print(a%b)
print(a**b)
```

In [ ]:

```
# 8 Floor, Ceil and Rint

import numpy
numpy.set_printoptions(sign=' ')
a = numpy.array(input().split(),float)

print(numpy.floor(a))
print(numpy.ceil(a))
print(numpy.rint(a))
```

In [ ]:

```
# 9 Sum and Pord

import numpy

n,m = map(int, input().split())
A = numpy.array([input().split() for _ in range(n)],int)
print(numpy.prod(numpy.sum(A, axis=0), axis=0))
```

In [ ]:

```
# 10 Min and Max

import numpy
n,m = map(int, input().split())
A = numpy.array([input().split() for _ in range(n)],int)
minA = numpy.min(A, axis=1)
maxv = numpy.max(minA)
print(maxv)
```

In [ ]:

```
# 11 Mean Var and Std

import numpy
array = []
n, m = map(int, input().split())
for i in range(n):
    array.append(list(map(int, input().split())))
array = numpy.array(array)
print(numpy.mean(array, axis=1))
print(numpy.var(array, axis=0))
print(round(numpy.std(array), 11))
```

In [ ]:

```
# 12 Dot and Cross

import numpy as np
n = int(input())
a = np.array([input().split() for i in range(n)], int)
b = np.array([input().split() for i in range(n)], int)
print(np.dot(a, b))
```

In [ ]:

```
# 13 inner and Outer

import numpy as np
a = np.array(input().split(), int)
b = np.array(input().split(), int)
print(np.inner(a, b))
print(np.outer(a, b))
```

In [ ]:

```
# 14 Polynomials

import numpy
n = list(map(float,input().split()))
m = input()
print(format(numpy.polyval(n,int(m)),'.1f'))
```

In [ ]:

```
# 15 Linear Algebra

import numpy
n=int(input())
a=numpy.array([input().split() for i in range(n)],float)
numpy.set_printoptions(legacy='1.13') #set up the format of output
print(numpy.linalg.det(a))
```

Problem 2

Birthday Cake Candles

In [ ]:

```
#!/bin/python3

import math
import os
import random
import re
import sys

#
# Complete the 'birthdayCakeCandles' function below.
#
# The function is expected to return an INTEGER.
# The function accepts INTEGER_ARRAY candles as parameter.
#

def birthdayCakeCandles(candles):
    return candles.count(max(candles))

if __name__ == '__main__':
    fptr = open(os.environ['OUTPUT_PATH'], 'w')

    candles_count = int(input().strip())

    candles = list(map(int, input().rstrip().split()))

    result = birthdayCakeCandles(candles)

    fptr.write(str(result) + '\n')

    fptr.close()
```

Number Line Jumps

In [ ]:

```
#!/bin/python3

import math
import os
import random
import re
import sys

#
# Complete the 'kangaroo' function below.
#
# The function is expected to return a STRING.
# The function accepts following parameters:
#  1. INTEGER x1
#  2. INTEGER v1
#  3. INTEGER x2
#  4. INTEGER v2
#

def kangaroo(x1, v1, x2, v2):
    # Write your code here
    return ("YES" if (x2 - x1) * (v2 - v1) < 0 and (x2 - x1) % (v2 - v1) == 0 else "NO")

if __name__ == '__main__':
    fptr = open(os.environ['OUTPUT_PATH'], 'w')

    first_multiple_input = input().rstrip().split()

    x1 = int(first_multiple_input[0])

    v1 = int(first_multiple_input[1])

    x2 = int(first_multiple_input[2])

    v2 = int(first_multiple_input[3])

    result = kangaroo(x1, v1, x2, v2)

    fptr.write(result + '\n')

    fptr.close()
```

Viral Advertising

In [ ]:

```
#!/bin/python3

import math
import os
import random
import re
import sys

#
# Complete the 'viralAdvertising' function below.
#
# The function is expected to return an INTEGER.
# The function accepts INTEGER n as parameter.
#

def viralAdvertising(n):
    m = [2]
    for i in range(n-1):
        m.append(int(3*m[i]/2))
    return sum(m)

if __name__ == '__main__':
    fptr = open(os.environ['OUTPUT_PATH'], 'w')

    n = int(input().strip())

    result = viralAdvertising(n)

    fptr.write(str(result) + '\n')

    fptr.close()
```

Recursive Digit Sum

In [ ]:

```
#!/bin/python3

import math
import os
import random
import re
import sys

#
# Complete the 'superDigit' function below.
#
# The function is expected to return an INTEGER.
# The function accepts following parameters:
#  1. STRING n
#  2. INTEGER k
#

def superDigit(n, k):
    x = n * k % 9
    if x :
        x = x
    else:
        x = 9

    return x

if __name__ == '__main__':
    fptr = open(os.environ['OUTPUT_PATH'], 'w')

    n,k = map(int,input().split())

    result = superDigit(n, k)

    fptr.write(str(result) + '\n')

    fptr.close()
```

Insertion Sort - Part 1

In [ ]:

```
#!/bin/python3

import math
import os
import random
import re
import sys

#
# Complete the 'insertionSort1' function below.
#
# The function accepts following parameters:
#  1. INTEGER n
#  2. INTEGER_ARRAY arr
#

def insertionSort1(n, arr):
    i = n-1
    item = arr[i]
    while(i>0 and item<arr[i-1]):
        arr[i] = arr[i-1]
        print(*arr)
        i-=1
    arr[i] = item
    print(*arr)

if __name__ == '__main__':
    n = int(input().strip())

    arr = list(map(int, input().rstrip().split()))

    insertionSort1(n, arr)
```

Insertion Sort - Part 2

In [ ]:

```
n=int(input())
arr=list(map(int,input().split()))

for i in range(1,n):
    while i>0:
        if arr[i] < arr[i-1]:
            arr[i-1],arr[i]=arr[i],arr[i-1]
        i=i-1
    print(*arr,sep=" ")
```