AIML_Python_M1_D3_PracticeLab

Test Summary

- No. of Sections: 1
- No. of Questions: 10
- Total Duration: 100 min

Section 1 - Coding

Section Summary

- No. of Questions: 10
- Duration: 100 min

Additional Instructions:

None

Q1. **Charity Dinner**

WeCanNGO Trust is organizing a charity dinner at St. John's College. Since older students are both wiser and richer, the members of the trust decide that the price of each ticket will be based on how many years the students have been in the school. A first-year student will buy a PINK ticket, a second-year student will buy a GREEN ticket, a third-year student will buy a RED ticket, and a fourth-year student will buy an ORANGE ticket.
Assume that all tickets are sold. Each color of the ticket is uniquely priced. Write a program to output all combinations of tickets that produce exactly the desired amount to be raised. The combinations may appear in a specific order. First ORANGE is considered, then RED, then GREEN, and finally PINK. Also, display the total number of combinations found and the smallest number of tickets to be printed to raise the desired amount so that printing cost is minimized.

Input Format

The First 4 lines of the input corresponding to the cost of a PINK, GREEN, RED, ORANGE ticket (in the exact order).
The last line of the input corresponds to the amount of money raised by selling tickets.

Output Format

Output all combinations of tickets that produce exactly the desired amount to be raised. The combinations may appear in any order. Output the total number of combinations found. Output the smallest number of tickets to print to raise the desired amount to minimize printing cost.
Refer sample input and output for formatting specifications.

Sample Input

```
1
2
3
4
```

Sample Output

```
# of PINK is 0 # of GREEN is 0 # of RED is 1 # of ORA
# of PINK is 1 # of GREEN is 1 # of RED is 0 # of ORA
# of PINK is 3 # of GREEN is 0 # of RED is 0 # of ORA
```

Sample Input

```
1
2
4
3
```

Sample Output

```
# of PINK is 0 # of GREEN is 0 # of RED is 1 # of ORA
# of PINK is 0 # of GREEN is 2 # of RED is 0 # of ORA
# of PINK is 1 # of GREEN is 0 # of RED is 0 # of ORA
```

Time Limit: 10 ms Memory Limit: 256 kb Code Size: 1024 kb

Q2. **Series 3**

The Event Organizing Company "Buzzcraft" focuses on event management to create a win-win situation for all involved stakeholders. Buzzcraft doesn't look at building one-time associations with clients, instead, aim at creating long-lasting collaborations that will span years to come. This goal of the company has helped them evolve and expand big with organizing many events to date.
The number of events that the company organizes every month is recorded sensibly and is deemed to have followed a specific series like 30, 35, 38, 41, 54, 53 ...
Write a program that takes an integer N as the input and will output the series till the Nth term.

Input Format

The first line of the input is an integer N.

Output Format

Output a single line of the series till Nth term, each separated by a space
Refer sample input and output for formatting specifications.

Sample Input

Sample Output

5	30 35 38 41 54

Sample Input **Sample Output**

10	30 35 38 41 54 53 78 71 110 95

Time Limit: 10 ms Memory Limit: 256 kb Code Size: 1024 kb

Q3. **Winter Challenge**

Swapna is a regular reader of Youth Digest magazine. The magazine has a whole host of fun and interesting facts from around for the youth especially that encourage interactivity and enhances their imagination.
"Winter Challenge" is an event announced in the December month edition of the magazine. Readers of the magazine who are between 10 to 15 years can participate in the special challenge. Those readers who participate and give the correct answer for the challenge will avail exciting gift coupons. According to the event, the challenge published was:
Given 0 < x < m, where x and m are integers, the modular inverse of x is the unique integer n, 0 < n < m, such that the remainder upon dividing x × n by m is 1. For example, 4 x 13 = 52 = 17 x 3 + 1, so the remainder when 52 is divided by 17 is 1, and thus 13 is the inverse of 4 modulo 17.
Swapna wants your help to find the correct answer for the problem based on the inputs given in the magazine. You are to write a program which accepts as input the two integers x and m, and outputs either the modular inverse n, or -1 if there is no such integer n. You may assume that m ≤ 100.
Function Specifications:
Use the function name, return type and the argument type as:
int findValue(int,int)
The Function should return the appropriate value or return -1 if no such value exists.

Input Format

First line of the input is an integer that corresponds to x.
Second line of the input is an integer that corresponds to m.

Output Format

Output should display the appropriate value or print -1 otherwise.
Refer sample input and output for formatting specifications.

Sample Input **Sample Output**

6 10	-1

Sample Input **Sample Output**

4 17	13

Time Limit: 10 ms Memory Limit: 256 kb Code Size: 1024 kb

Q4. Write a function to check whether a number is perfect or not. In number theory, a perfect number is a positive integer that is equal to the sum of its proper positive divisors, that is, the sum of its positive divisors excluding the number itself (also known as its aliquot sum). Equivalently, a perfect number is a number that is half the sum of all of its positive divisors (including itself).

Example : The first perfect number is 6, because 1, 2, and 3 are its proper positive divisors, and 1 + 2 + 3 = 6. Equivalently, the number 6 is equal to half the sum of all its positive divisors: (1 + 2 + 3 + 6) / 2 = 6. The next perfect number is 28 = 1 + 2 + 4 + 7 + 14. This is followed by the perfect numbers 496 and 8128.

Input Format

The input consists of a number.

Output Format

The output prints whether the number is perfect or not.
Refer sample input and output for formatting specifications.

Sample Input **Sample Output**

6	6 is a perfect number

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q5. Write a python program to find intersection of two given lists using lambda

Arrange the output in ascending order.

Input Format

First line of input consists of space separated integers represents list1
Second line of input consists of space separated integers represents list2

Output Format

Output consists of list of common elements present in both list1 and list2.

Sample Input

```
1 2 3 4 5 6 7 8
2 3 1
```

Sample Output

```
[1, 2, 3]
```

Time Limit: 10 ms Memory Limit: 256 kb Code Size: 1024 kb

Q6. Write a program to generate a list of the first N Fibonacci numbers, 0 being the first number. Then, apply the map function and a lambda expression to square each Fibonacci number and print the list.

Input Format

One line of input: an integer N.

Output Format

A list on a single line containing the square of the first N Fibonacci numbers.

Sample Input

```
5
```

Sample Output

```
[0, 1, 1, 4, 9]
```

Time Limit: 10 ms Memory Limit: 256 kb Code Size: 1024 kb

Q7. Create a class name **Employee** with the instance attribute firstname and lastname.
Form the email by joining the first and last name together with a "." in between, and follow it with '@company.com' at the end. Make sure everything is in lowercase.
Examples:
emp_1 = Employee("John", "Smith")
emp_2.email → "mary.sue@company.com"

Input Format

The first line of the input contains the first name
The second line of the input contains the last name.

Output Format

Output prints the email.

Sample Input

```
ANBU
ARUT
```

Sample Output

```
anbu.arut@company.com
```

Sample Input

```
PLACEMENT
SEASON
```

Sample Output

```
placement.season@company.com
```

Time Limit: 100 ms Memory Limit: 256 kb Code Size: 1024 kb

Q8. Create a class called Sum. Write a Python program to find the three elements that sum to zero from a set (array) of n real numbers.

Input Format

Input consists of a list of integers.

Output Format

Output prints find the three elements that sum to zero from a set (array) of n real numbers.

Sample Input

```
-10 6 7 -13 8 2
```

Sample Output

```
[[-13, 6, 7], [-10, 2, 8]]
```

Sample Input

```
-21 7 3 8 3 -9 6 4
```

Sample Output

```
[[-9, 3, 6]]
```

Time Limit: 100 ms Memory Limit: 256 kb Code Size: 1024 kb

Q9. Seetha wants to understand the process of exception handling. She wants to print the variable x which was not assigned any value. But she needs to print the error message 'An exception occurred' instead of getting an error 'variable x not declared'.

Help her in this process.

Output Format

Displays the error message 'An exception occurred'

Sample Input

```

```

Sample Output

```
An exception occurred
```

Time Limit: 100 ms Memory Limit: 256 kb Code Size: 1024 kb

Q10. Write a program to Obtain the start time and end time for the stage event show. If the user enters a different format other than specified, an exception occurs and the program is interrupted. To avoid that, handle the exception and prompt the user to enter the right format as specified.

Start time and end time should be in the format 'YYYY-MM-DD HH:MM: SS'
If the input is in the above format, print the start time and end time.
If the input does not follow the above format, print "**Event time should be in the format 'YYYY-MM-DD HH:MM: SS'**"

Refer sample Input/Output format specification

Input Format

The first line of input consists of the start time of the event.
The second line of the input consists of the end time of the event.

Output Format

If the input is in the given format, print the start time and end time.
If the input does not follow the given format, print "**Event time should be in the format 'YYYY-MM-DD HH:MM: SS'**"

Sample Input

```
2020-01-10 09:0:0
2020-01-11 17:0:0
```

Sample Output

```
2020-01-10 09:0:0
2020-01-11 17:0:0
```

Sample Input

```
2020-01-10
2020-01-11 04:00:00
```

Sample Output

```
Event time should be in the format 'YYYY-MM-DD HH:MM:S
```

Sample Input

```
2020-01-10 09
11-01-2020 04:00:00
```

Sample Output

```
Event time should be in the format 'YYYY-MM-DD HH:MM:S
```

Sample Input

```
2020-01-10 09:00:00
2020-01-11 18:0
```

Sample Output

```
Event time should be in the format 'YYYY-MM-DD HH:MM:S
```

Sample Input

```
15-10-2012 08:00:00
16-10-2012 09:00:00
```

Sample Output

```
Event time should be in the format 'YYYY-MM-DD HH:MM:S
```

Time Limit: 100 ms Memory Limit: 256 kb Code Size: 1024 kb

Answer Key & Solution

Section 1 - Coding

Q1

Test Case

Input

```
1
2
3
4
```

Output

```
# of PINK is 0 # of GREEN is 1 # of RED is 1 #
# of PINK is 1 # of GREEN is 0 # of RED is 0 #
# of PINK is 1 # of GREEN is 2 # of RED is 0 #
```

Weightage - 15

Input

```
2
3
4
5
```

Output

```
# of PINK is 0 # of GREEN is 2 # of RED is 0 #
# of PINK is 1 # of GREEN is 0 # of RED is 1 #
# of PINK is 3 # of GREEN is 0 # of RED is 0 #
```

Weightage - 15

Input

```
4
5
6
7
```

Output

```
# of PINK is 0 # of GREEN is 0 # of RED is 0 #
# of PINK is 0 # of GREEN is 3 # of RED is 1 #
# of PINK is 1 # of GREEN is 1 # of RED is 2 #
```

Weightage - 15

Input

```
9
8
7
6
```

Output

```
# of PINK is 0 # of GREEN is 0 # of RED is 0 #
Total combinations is 1
Minimum number of tickets to print is 1
```

Weightage - 10

Input

```
5
4
6
8
```

Output

```
# of PINK is 0 # of GREEN is 1 # of RED is 1 #
# of PINK is 2 # of GREEN is 0 # of RED is 0 #
Total combinations is 2
```

Weightage - 15

Input

```
8
4
2
6
```

Output

```
# of PINK is 0 # of GREEN is 0 # of RED is 0 #
# of PINK is 0 # of GREEN is 0 # of RED is 3 #
# of PINK is 0 # of GREEN is 0 # of RED is 6 #
```

Weightage - 20

Input

```
1
2
3
4
```

Output

```
# of PINK is 0 # of GREEN is 0 # of RED is 1 #
# of PINK is 1 # of GREEN is 1 # of RED is 0 #
# of PINK is 3 # of GREEN is 0 # of RED is 0 #
```

Weightage - 10

Sample Input

```
1
2
3
4
```

Sample Output

```
# of PINK is 0 # of GREEN is 0 # of RED is 1 #
# of PINK is 1 # of GREEN is 1 # of RED is 0 #
# of PINK is 3 # of GREEN is 0 # of RED is 0 #
```

Sample Input

```
1
2
4
3
```

Sample Output

```
# of PINK is 0 # of GREEN is 0 # of RED is 1 #
# of PINK is 0 # of GREEN is 2 # of RED is 0 #
# of PINK is 1 # of GREEN is 0 # of RED is 0 #
```

Solution

```
p=int(input())
g=int(input())
r=int(input())
o=int(input())
t=int(input())
count = 0
c = t
for i in range(0,t+1):
    for j in range(0,t+1):
        for k in range(0,t+1):
            for l in range(0,t+1):
                if(p*i+g*j+r*k+o*l==t):
                    print("# of PINK is",i,"# of GREEN is",j,"# of RED is",k,"# of ORANGE is",l)
                    count+=1
                    if c>(i+j+k+l):
                        c=i+j+k+l
print("Total combinations is",count)
print("Minimum number of tickets to print is",c)
```

Q2

Test Case

Input

```
36
```

Output

```
30 35 38 41 54 53 78 71 110 95 150 125 198 161
```

Weightage - 15

Input

```
1
```

Output

```
30
```

Weightage - 15

Input

```
50
```

Output

```
30 35 38 41 54 53 78 71 110 95 150 125 198 161
```

Weightage - 15

Input

```
97
```

Output

```
30 35 38 41 54 53 78 71 110 95 150 125 198 161 
```

Weightage - 15

Input

```
25
```

Output

```
30 35 38 41 54 53 78 71 110 95 150 125 198 161 
```

Weightage - 15

Input

```
12
```

Output

```
30 35 38 41 54 53 78 71 110 95 150 125
```

Weightage - 15

Input

```
5
```

Output

```
30 35 38 41 54
```

Weightage - 10

Sample Input

```
5
```

Sample Output

```
30 35 38 41 54
```

Sample Input

```
10
```

Sample Output

```
30 35 38 41 54 53 78 71 110 95
```

Solution

```
n = int(input())
st = 30
st1 = 35
for i in range(0,n):
    if(i%2==0):
        st = st+(i//2)*8
        print(st,end=" ")
    else:
        st1 = st1+(i//2)*6
        print(st1,end=" ")
```

Q3

Test Case

Input

```
1
1
```

Output

```
1
```

Weightage - 10

Input

```
8
9001
```

Output

```
7876
```

Weightage - 15

Input

```
654
765482
```

Output

```
-1
```

Weightage - 15

Input

```
87
888
```

Output

```
-1
```

Weightage - 15

Input

```
23
76
```

Output

```
43
```

Weightage - 15

Input

```
32
854
```

Output

```
-1
```

Weightage - 15

Input

```
5
27
```

Output

```
11
```

Weightage - 15

Sample Input

```
6
10
```

Sample Output

```
-1
```

Sample Input

```
4
17
```

Sample Output

```
13
```

Solution

```
def findValue(n1,n2):
    s=0
    for i in range(0,1000000):
        if((n2*i+1)%n1==0):
            s=1
            return ((n2*i+1)//n1)
    if s==0:
        return -1
n1=int(input())
n2=int(input())
print(findValue(n1,n2))
```

Q4

Test Case

Input

```
6
```

Output

```
6 is a perfect number
```

Weightage - 10

Input

```
28
```

Output

```
28 is a perfect number
```

Weightage - 10

Input

```
496
```

Output

```
496 is a perfect number
```

Weightage - 10

Input

```
8128
```

Output

```
8128 is a perfect number
```

Weightage - 20

Input

```
4865
```

Output

```
4865 is not a perfect number
```

Weightage - 20

Input

```
88884
```

Output

```
88884 is not a perfect number
```

Weightage - 20

Input

```
484
```

Output

```
484 is not a perfect number
```

Weightage - 10

Sample Input

```
6
```

Sample Output

```
6 is a perfect number
```

Solution

```
def perfect_number(n):
    sum = 0
    for x in range(1, n):
        if n % x == 0:
            sum += x
    return sum == n

num = int(input())
res = perfect_number(num)
if res == True:
    print(num,"is a perfect number")
else:
    print(num,"is not a perfect number")
```

Q5

Test Case

Input

```
100 200 45 7 6 8 9
9 200 4 5
```

Output

```
[9, 200]
```

Weightage - 20

Input

```
1 4 5 6 7 8 9 10 11
0 4 5 1 6 7 8
```

Output

```
[1, 4, 5, 6, 7, 8]
```

Weightage - 20

Input

```
90 34 8 5 1 0 121 345 6789 345 90 12 47 58 69 3
90 6789 3 4 1 0
```

Output

```
[0, 1, 90, 6789]
```

Weightage - 20

Input

```
123 5 67 90 34 56 89 10 578
1 2 3 4 5 6 7 8 9 0 123 89 0 6 4 6 8 3 5 7 8
```

Output

```
[5, 5, 89, 123]
```

Weightage - 20

Input

```
1 2
1 2
```

Output

```
[1, 2]
```

Weightage - 20

Sample Input

```
1 2 3 4 5 6 7 8
2 3 1
```

Sample Output

```
[1, 2, 3]
```

Solution

```
inp_list1 = input().split()

#optional
for i in range(0, len(inp_list1)):
    inp_list1[i] = int(inp_list1[i])

inp_list2 = input().split()

#optional
for i in range(0, len(inp_list2)):
    inp_list2[i] = int(inp_list2[i])

result = sorted(list(filter(lambda x: x in inp_list1, inp_list2)))
print(result)
```

Q6

Test Case

Input **Output**

6	[0, 1, 1, 4, 9, 25]

Weightage - 20

Input **Output**

5	[0, 1, 1, 4, 9]

Weightage - 20

Input **Output**

8	[0, 1, 1, 4, 9, 25, 64, 169]

Weightage - 20

Input **Output**

10	[0, 1, 1, 4, 9, 25, 64, 169, 441, 1156]

Weightage - 20

Input **Output**

12	[0, 1, 1, 4, 9, 25, 64, 169, 441, 1156, 3025, 79

Weightage - 20

Sample Input **Sample Output**

5	[0, 1, 1, 4, 9]

Solution

```
N = int(input())
a=b=1
L = [0, 1, 1]
for i in range(4, N+1):
    L.append(a+b)
    a,b = b,a+b
square = lambda a: a**2
print (list(map(square, L[:N])))
```

Q7

Test Case

Input

```
PLACEMENT
SEASON
```

Output

```
placement.season@company.com
```

Weightage - 20

Input

```
EXAM
PREPARATION
```

Output

```
exam.preparation@company.com
```

Weightage - 20

Input

```
nandhini
p
```

Output

```
nandhini.p@company.com
```

Weightage - 20

Input

```
anbu
anbarasi
```

Output

```
anbu.anbarasi@company.com
```

Weightage - 20

Input

```
placement
preparation
```

Output

```
placement.preparation@company.com
```

Weightage - 20

Sample Input

```
ANBU
ARUT
```

Sample Output

```
anbu.arut@company.com
```

Sample Input

```
PLACEMENT
SEASON
```

Sample Output

```
placement.season@company.com
```

Solution

```
class Employee:
    def __init__(self, firstname, lastname):
```

```
        self.firstname = firstname
        self.lastname = lastname
    def description(self):
        print('{}.{}@company.com'.format(firstname, lastname).lower())


firstname=input()
lastname=input()
employee1=Employee(firstname,lastname)
employee1.description()
```

Q8

Test Case

Input

```
-10 6 7 -13 8 2
```

Output

```
[[-13, 6, 7], [-10, 2, 8]]
```

Weightage - 25

Input

```
-10 6 7 -13 8 2 -6 4 2
```

Output

```
[[-13, 6, 7], [-10, 2, 8], [-10, 4, 6], [-6, 2, 4
```

Weightage - 25

Input

```
-10 6 7 -13 8 2 -3 6 3
```

Output

```
[[-13, 6, 7], [-10, 2, 8], [-10, 3, 7]]
```

Weightage - 25

Input

```
-21 7 3 8 3 -9 6 4
```

Output

```
[[-9, 3, 6]]
```

Weightage - 25

Sample Input

```
-10 6 7 -13 8 2
```

Sample Output

```
[[-13, 6, 7], [-10, 2, 8]]
```

Sample Input

```
-21 7 3 8 3 -9 6 4
```

Sample Output

```
[[-9, 3, 6]]
```

Solution

```
class Sum:
 def threeSum(self, nums):
        nums, result, i = sorted(nums), [], 0
        while i < len(nums) - 2:
            j, k = i + 1, len(nums) - 1
            while j < k:
                if nums[i] + nums[j] + nums[k] < 0:
                    j += 1
                elif nums[i] + nums[j] + nums[k] > 0:
                    k -= 1
                else:
                    result.append([nums[i], nums[j], nums[k]])
                    j, k = j + 1, k - 1
                    while j < k and nums[j] == nums[j - 1]:
                        j += 1
                    while j < k and nums[k] == nums[k + 1]:
                        k -= 1
            i += 1
            while i < len(nums) - 2 and nums[i] == nums[i - 1]:
                i += 1
        return result
n=list(map(int,input().split()))
print(Sum().threeSum(n))
```

Q9

Test Case

Input	Output
	An exception occurred

Weightage - 100

Sample Input	Sample Output
	An exception occurred

Solution

```
try:
    print(x)
except:
    print('An exception occurred')
```

Q10

Test Case

Input	Output
2019-02-02 08:00:00 2019-02-04 13:00:00	2019-02-02 08:00:00 2019-02-04 13:00:00

Weightage - 20

Input

```
2016-02-02
2019-02-10 10:00:00
```

Output

```
Event time should be in the format 'YYYY-MM-DD HH
```

Weightage - 20

Input

```
1988-09-23 08:00:00
1988-09-23 15:00:00
```

Output

```
1988-09-23 08:00:00
1988-09-23 15:00:00
```

Weightage - 20

Input

```
15-10-2020 11:30:00
16-10-2020 15:00:00
```

Output

```
Event time should be in the format 'YYYY-MM-DD HH
```

Weightage - 20

Input

```
2000-8-10 9:0:0
2000-8-15 15:0:0
```

Output

```
2000-8-10 9:0:0
2000-8-15 15:0:0
```

Weightage - 20

Sample Input

```
2020-01-10 09:0:0
2020-01-11 17:0:0
```

Sample Output

```
2020-01-10 09:0:0
2020-01-11 17:0:0
```

Sample Input

```
2020-01-10
2020-01-11 04:00:00
```

Sample Output

```
Event time should be in the format 'YYYY-MM-DD HH
```

Sample Input

```
2020-01-10 09
11-01-2020 04:00:00
```

Sample Output

```
Event time should be in the format 'YYYY-MM-DD HH
```

Sample Input

```
2020-01-10 09:00:00
2020-01-11 18:0
```

Sample Output

```
Event time should be in the format 'YYYY-MM-DD HH
```

Sample Input

```
15-10-2012 08:00:00
16-10-2012 09:00:00
```

Sample Output

```
Event time should be in the format 'YYYY-MM-DD HH
```

Solution

```
import datetime
def validate(date_text1,date_text2):
    try:
        datetime.datetime.strptime(date_text1, '%Y-%m-%d %H:%M:%S')
        datetime.datetime.strptime(date_text2, '%Y-%m-%d %H:%M:%S')
        print(date_text1)
        print(date_text2)
    except:
        print("Event time should be in the format 'YYYY-MM-DD HH:MM:SS")

date_text1 = input()
date_text2 = input()
validate(date_text1,date_text2)
```