Practice Problems in Python

BStat, 1st Year 2023–2024

Date: 01 August, 2023

PRACTICING INSTRUCTIONS

1. Write down the best possible implementation of the problems listed below.
2. Take care of the error cases too.
3. After you finish the coding, look at the model answers at the end and understand.

NOTE: The programs are to be written in Python and should be well commented. All programs should take the required inputs from standard input buffer and print the desired outputs to the standard output buffer, until otherwise stated.

Q1. (EASY) Write a program to count the number of unique characters (including special characters too) in a string given as user input.

Q2. (EASY) Given a list of strings, write a program to return the palindromes within it.

Q3. (EASY) Given a list of strings, write a program to count the number of strings where the string length is no less than 3 and the first and last characters are the same.

Q4. (EASY) Write a program to crop out the alphanumeric characters from a string given as user input.

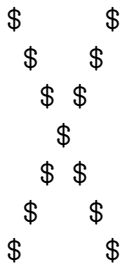
Q5. (EASY) Write a program to check whether a number given as user input is a power of 4 or not. Accordingly, print POWER OF 4 or NOT POWER OF 4.

Q6. (EASY) Write a program to take three numbers as input from the user and verify whether all of them are same or not. Accordingly, print SAME or DIFFERENT.

Q7. (EASY) Let us define a number as CONJOINED TWIN if it can be represented as the sum of two distinct numbers that are reverse of each other (e.g., 12 and 21 are reverse of each other). Note that, for being reverse to each other they must have same number of significant digits. Hence, 100 is not reverse of 1 (considering 1 as 001). Write a program to show all the CONJOINED TWIN numbers that are no more than n in increasing order.

Q8. (EASY) Recall that a semi-prime number can be represented as $n = p * q$, the multiplication of two prime factors. Write a program to verify whether a number is semi-prime or not. Accordingly, print SEMI-PRIME or NOT SEMI-PRIME.

Q9. (EASY) Write a program to print the following pattern given the line number as user input.



Q10. (MEDIUM) Write a program to verify whether an input matrix is square or not. If it is not a square matrix, print NOT SQUARE. Otherwise, further check whether it is singular (determinant is 0) or unimodular (determinant is 1). Accordingly, print SQUARE - SINGULAR or SQUARE - UNIMODULAR, otherwise print SQUARE - OTHER.

Q11. (MEDIUM) Let us define a string, comprising English alphabets, as NICE if each vowel within it are equidistant from its successor and predecessor vowel, if applicable. E.g., "rhythm", "cool", "malayalam" are NICE strings. Write a program to verify whether a given string is NICE or not. You are required to take the string as a direct input without asking for its length.

Q12. (MEDIUM) Two elements $A[i]$ and $A[j]$ of a list A are said to form an inversion pair if $A[i] > A[j]$ but $i < j$. Write a program to count the number of inversion pairs in a list A containing distinct integers.

Q13. (MEDIUM) Given two binary strings representing two integers, find the product of the two integers using Karatsuba's algorithm.

Q14. (HARD) A group of n friends $F_1, F_2, \ldots, F_n$ decide to try their luck at a lottery. Each person F_i buys a number X_i of lottery tickets. Each lottery ticket has a digit (0-9) printed on it. The rule for winning the jackpot is as follows. Each person is asked to announce the largest multiple of 3 that can be formed by selecting and arranging the digits on his lottery tickets. The person who has the highest such multiple wins the jackpot. Note that a person's tickets may not have distinct digits on them.

Q15. (HARD) Suppose we define the angle between a pair of hands (denoting hour, minute, and second) of a clock as the lowest of the two possible angles. Let the hands are said to be in EQUIANGULAR position if the angles between them are the same. Further assume that the hands of a clock are said to be in SEMI-EQUIANGULAR position if the angles between any two pairs of hands are the same. Write a program that takes the time as user input (in the 24-hour format) and returns whether the clock is in EQUIANGULAR, SEMI-EQUIANGULAR, or NONE of the above positions. Consider that there is a tolerance level to ensure whether a pair of angles are same or not. Given the two angles α and β, they are said to be same for a tolerance level δ if $|\alpha - \beta| \leq \delta$.

Q16. (HARD) Let there be a complex matrix script denoted by a grid of strings. It consists of

alphanumeric characters, spaces and special symbols. To decode the script, you need to read each column from top to bottom, starting from the leftmost column, and select only the alphanumeric characters and connect them. If there are symbols or spaces between two alphanumeric characters of the decoded script, then replace them with a single space ' ' for better readability. Note that the alphanumeric characters consist of: {0-9, a-z, A-Z}. Note that you are not allowed to use `if` conditions for decoding. Consider the following input.

```
7 3
Tsi
h%x
i #
sM
$a
#t%
ir!
```

The sample output for this will be as follows.

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

Model Answers:

Q1. GENERAL APPROACH

```
str = input('Enter a string: ')
set = set(str)
print(len(set))
```

BETTER APPROACH

```
print(len(set(input('Enter a string: '))))
```

Q2. GENERAL APPROACH

```
ls = ['m', 'malay', 'malayalam']
lsNew = []
for l in ls:
    if l == l[::-1]:
        lsNew.append(l)
print(lsNew)
```

BETTER APPROACH

```
ls = ['m', 'malay', 'malayalam']
print([l for l in ls if l == l[::-1]])
```

Q3. GENERAL APPROACH

```
ls = ['m', 'malay', 'malayalam']
count = 0
for l in ls:
    if len(l) >= 3 and l[0] == l[-1]:
        count = count + 1
print(count)
```

BETTER APPROACH

```
ls = ['m', 'malay', 'malayalam']
print(len([l for l in ls if len(l) >= 3 and l[0] == l[-1]]))
```

Q4. GENERAL APPROACH

```
str = input('Enter a string: ')
for s in str:
    if not (s.isalpha() or s.isnumeric()):
        print(s, end = '')
```

BETTER APPROACH

```
str = input('Enter a string: ')
print(' '.join([s for s in str if not (s.isalpha() or s.isnumeric())]))
```

Q5. GENERAL APPROACH

```
n = int(input('Enter a number: '))
flag = 0
if n & (n - 1) == 0:
    while n & 3 == 0:
    n >>= 2
    if n & 3 == 1:
        flag = 1
print ('POWER OF 4' if flag else 'NOT POWER OF 4')
```

BETTER APPROACH

```
n = int(input('Enter a number: '))
flag = 0
if n != 0 and (n & (n - 1) == 0) and not(n & 0xAAAAAAAA):
    flag = 1
print ('POWER OF 4' if flag else 'NOT POWER OF 4')
```

Q6. GENERAL APPROACH

```
a, b, c = input('Enter 3 numbers: ').split()
if a == b and b == c:
    print('SAME')
else:
    print('DIFFERENT')
```

BETTER APPROACH

```
a, b, c = input('Enter 3 numbers: ').split()
if id(a) == id(b) and id(b) == id(c):
    print('SAME')
else:
    print('DIFFERENT')
```

Q7. GENERAL APPROACH

...

BETTER APPROACH

...

Q8. GENERAL APPROACH

...

BETTER APPROACH

...

Q9. GENERAL APPROACH

```
ln = int(input('Line number: '))
for i in range(ln):
    for j in range(ln):
        if i+j == ln-1 or i == j:
            print('*', end = '')
        else:
            print(' ', end = '')
    print()
```

BETTER APPROACH

```
ln = int(input('Line number: '))
for i in range(ln):
    print(''.join('*' if i+j == ln-1 or i == j else ' ' for j in range(ln)))
```

Q10. GENERAL APPROACH

...

BETTER APPROACH

...

Q11. GENERAL APPROACH

...

BETTER APPROACH

...

Q12. GENERAL APPROACH

...

BETTER APPROACH

...

Q13. GENERAL APPROACH

...

BETTER APPROACH

...

Q14. GENERAL APPROACH

...

BETTER APPROACH

...

Q15. GENERAL APPROACH

...

BETTER APPROACH

...

Q16. GENERAL APPROACH

...

BETTER APPROACH

```
import re
row, col = input().split()
MAT = ['' for _ in range(int(row))]
for i in range(int(row)):
    MAT[i] = input()
CMajor = [[MAT[i][j] for i in range(int(row))] for j in range(int(col))]
flatCMajor = [c for innerCMajor in CMajor for c in innerCMajor]
strCMajor = ''.join(flatCMajor)
CODE = re.sub('[^0-9a-zA-Z]+', ' ', strCMajor)
pos = re.search('[0-9a-zA-Z]', strCMajor[::-1])
print(CODE.rstrip()+strCMajor[len(strCMajor)-pos.start():])
```