**Pentafox Technologies (P) Ltd. Practise Exercises - Test on Logic Ability**

**#1**

A bicyclist cycles around a circular park with a pathway connecting two opposite end points of the path of length 7kms. Develop a logic that computes the total distance covered by the cyclist for a given set of rounds cycled.

**Main1.py**

def add\_num(a,b):

multiply=num1\*3.14\*num2;

return multiply;

num1=int(input(" Enter No. of Rounds: "))

num2=7

print("‘x’ Kms travelled:",add\_num(num1,num2))

**#2**

A fixed set of positive integers is dictated by the mathematics professor during a puzzle contest.The professor asks the students to find a pair of numbers that result in a given sum. Code a logic that can automate this puzzle. Use the below input for your exercise.

**Main2.py**

from collections import deque

def find\_target(values, target):

dq = deque(sorted([(val, idx) for idx, val in enumerate(values)]))

while True:

if len(dq) < 2:

print('No match found')

s = dq[0][0] + dq[-1][0]

if s > target:

dq.pop()

elif s < target:

dq.popleft()

else:

break

return dq[0], dq[-1]

values = [1, 2, 3, 4, 9]

target = 8

sol = find\_target(values, target)

print(sol[0][0],sol[1][0])

**#3**

Alice is a cryptanalyst who is in charge of transmitting messages to bob without any intruder getting hands on it. Alice thinks of transmitting the message by reversing it with a random character appended as prefix to the encoded message.

**Main3.py**

import random

a=input().lower()

a=a[-1::-1]

a=str(chr(random.randint(65,90)).upper())+a

print(a)

**#4**

As a computer engineer, you are requested to reduce the storage space needed to store the textual content in the computer. Write a logic that can compress the content as given in the below example.

**Main4.py**

a=input()

i=0

b=[]

while(i<len(a)-1) :

count=1

while(a[i]==a[i+1]):

i+=1

count+=1

if(i+1==len(a)):

break

if(count==1):

b.append(str(a[i]))

else:

b.append(str(a[i])+str(count))

i+=1

if(a[-1]!=a[-2]):

b.append(a[-1])

c="".join(i for i in b)

print(c)

**#5**

In a puzzle contest, the chairman of your English club posts a problem to compare a given pair of words and eliminate all common characters in them. To speed up the process of judging, the computer club head was requested to prepare computer logic. Please code a solution to the above problem applying your own skillset.

**Main5.py**

a=input()

b=input()

c=[]

for i in a:

if i not in b:

c.append(i)

for i in b:

if i not in a:

c.append(i)

d="".join(i for i in c)

print(d)

**#6**

A school camp is organized by a school to support the process of preparing their students for an examination. They are in need of a study timetable that has following assumptions:

**Assumptions:**

1. Total Days of Camp – 5 Days

2. Total Hours a day -- 5 Days

3. Total Subjects – 5 Subjects

**Note:** The timetable should not follow the same order and should be in random everyday. Prepare code logic to help the School

**Main6.py**

import random

l=[]

A=int(input())

for i in range (A):

n=random.sample(range(0,A),A)

l.append(n)

s=[]

for i in range(A):

x=input('enter name')

s.append(x)

for i in range(A):

print("Day-",i+1,end=' ')

for j in range(A):

c=l[i][j]

print(s[c],end=' ')

print("\n")

**#7**

The alphabetical value is represented from 1-26 for characters A-Z respectively. Using this principle generate a crypto decoder that can generate the message for transmitted sequence of alphabetical values.

**Main7.py**

str = list( input("Enter").split(",") )

for i in str:

a = chr( int( i ) + 64 )

print(a, end =" ")

**#8**

Implement CRUD operations as an API using Python Flask and a DB in backend (prefer MySQL or MariaDB).

**Main.py**

import pymysql

from app import app

from db\_config import mysql

from flask import jsonify

from flask import flash, request

from werkzeug.security import generate\_password\_hash, check\_password\_hash

@app.route('/add', methods=['POST'])

def add\_employee():#add employee

try:

\_json = request.json

\_name = \_json['name']

\_email = \_json['email']

\_password = \_json['password']

if \_name and \_email and \_password and request.method == 'POST':

\_hashed\_password = generate\_password\_hash(\_password)

sql = "INSERT INTO tbl\_employee(emp\_name, emp\_email, emp\_password) VALUES(%s, %s, %s)"

data = (\_name, \_email, \_hashed\_password,)

conn = mysql.connect()

cursor = conn.cursor()

cursor.execute(sql, data)

conn.commit()

resp = jsonify('employee added successfully!')

resp.status\_code = 200

return resp

else:

return not\_found()

except Exception as e:

print(e)

finally:

cursor.close()

conn.close()

@app.route('/employees')

def employees():

try:

conn = mysql.connect()

cursor = conn.cursor(pymysql.cursors.DictCursor)

cursor.execute("SELECT \* FROM tbl\_employee")

rows = cursor.fetchall()

resp = jsonify(rows)

resp.status\_code = 200

return resp

except Exception as e:

print(e)

finally:

cursor.close()

conn.close()

@app.route('/employee/<int:id>')

def employee(id):

try:

conn = mysql.connect()

cursor = conn.cursor(pymysql.cursors.DictCursor)

cursor.execute("SELECT \* FROM tbl\_employee WHERE emp\_id=%s", id)

row = cursor.fetchone()

resp = jsonify(row)

resp.status\_code = 200

return resp

except Exception as e:

print(e)

finally:

cursor.close()

conn.close()

@app.route('/update', methods=['POST'])

def update\_employee():#update the employee details

try:

\_json = request.json

\_id = \_json['id']

\_name = \_json['name']

\_email = \_json['email']

\_password = \_json['password']

if \_name and \_email and \_password and \_id and request.method == 'POST':

\_hashed\_password = generate\_password\_hash(\_password)

sql = "UPDATE tbl\_employee SET emp\_name=%s, emp\_email=%s, emp\_password=%s WHERE emp\_id=%s"

data = (\_name, \_email, \_hashed\_password, \_id,)

conn = mysql.connect()

cursor = conn.cursor()

cursor.execute(sql, data)

conn.commit()

resp = jsonify('employee updated successfully!')

resp.status\_code = 200

return resp

else:

return not\_found()

except Exception as e:

print(e)

finally:

cursor.close()

conn.close()

@app.route('/delete/<int:id>')

def delete\_employee(id):#delete the employee details

try:

conn = mysql.connect()

cursor = conn.cursor()

cursor.execute("DELETE FROM tbl\_employee WHERE emp\_id=%s", (id,))

conn.commit()

resp = jsonify('employee deleted successfully!')

resp.status\_code = 200

return resp

except Exception as e:

print(e)

finally:

cursor.close()

conn.close()

@app.errorhandler(404)

def not\_found(error=None):

message = {

'status': 404,

'message': 'Not Found: ' + request.url,

}

resp = jsonify(message)

resp.status\_code = 404

return resp

if \_\_name\_\_ == "\_\_main\_\_":

app.run()

**db\_config.py**

from app import app

from flaskext.mysql import MySQL

mysql = MySQL()

# MySQL configurations

app.config['MYSQL\_DATABASE\_USER'] = 'root'

app.config['MYSQL\_DATABASE\_PASSWORD'] = ''

app.config['MYSQL\_DATABASE\_DB'] = 'fine'

app.config['MYSQL\_DATABASE\_HOST'] = 'localhost'

mysql.init\_app(app)

**app.py**

from flask import Flask

app = Flask(\_\_name\_\_)

**queres**:

CREATE TABLE `tbl\_employee` (

`emp\_id` bigint COLLATE utf8mb4\_unicode\_ci NOT NULL AUTO\_INCREMENT,

`emp\_name` varchar(50) COLLATE utf8mb4\_unicode\_ci DEFAULT NULL,

`emp\_email` varchar(50) COLLATE utf8mb4\_unicode\_ci DEFAULT NULL,

`emp\_password` varchar(300) COLLATE utf8mb4\_unicode\_ci DEFAULT NULL,

PRIMARY KEY (`emp\_id`)

) ENGINE=InnoDB AUTO\_INCREMENT=1 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

Add employee

{

"name":"Sheikfaaruk",

"email":"sheikfaaruk@gmail.com",

"password":"pwd@123"

}

update employee

{

"id":2,

"name":"vinobala",

"email":"vinosimp@gmail.com",

"password":"pwd@456"

}

Delete the employee

http://127.0.0.1:5000/delete/enter\_the\_id\_no

Display the Employees

http://127.0.0.1:5000/employees