विद्ययाऽमृतमश्नुते

Computer Science Practical File

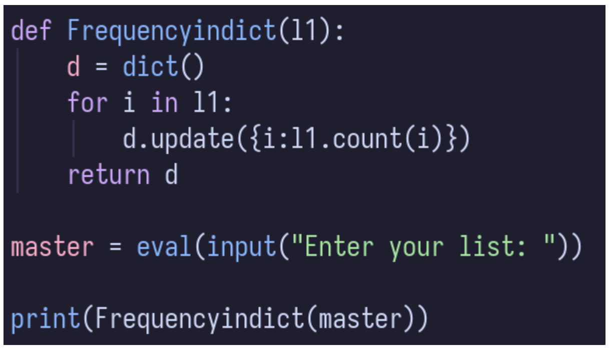
For session 2022-23

-Karttikeya Sinha (XII-C)

Roll No: ________

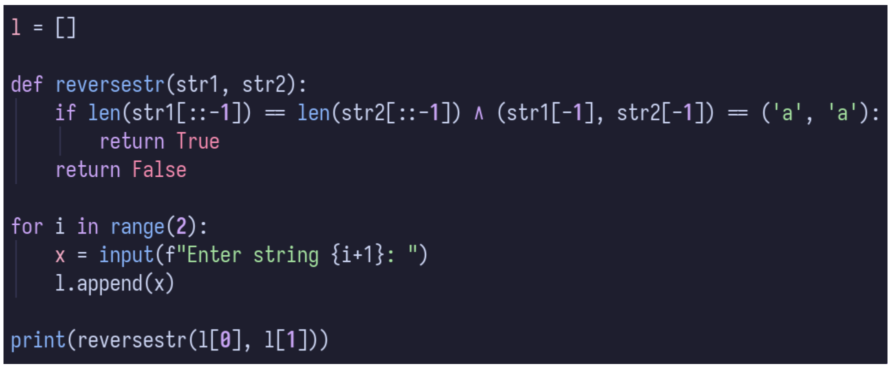
SECTION 1: PYTHON

Q1) Write a program to define a function Frequencyindict(list) to pass a list through function and return a dictionary having the elements as the number(key) and frequency of the number (value). Display the dictionary and list at the end.



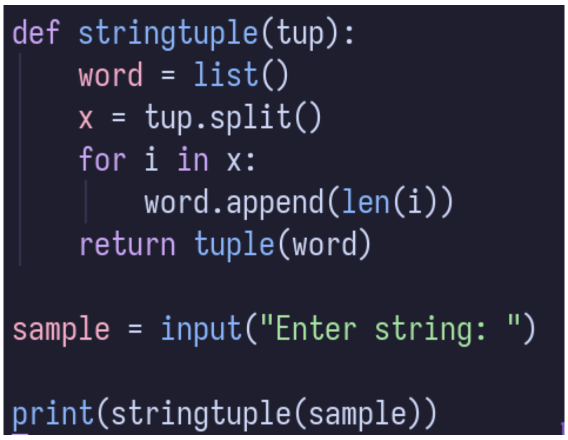
```python
def Frequencyindict(l1):
    d = dict()
    for i in l1:
        d.update({i:l1.count(i)})
    return d

master = eval(input("Enter your list: "))

print(Frequencyindict(master))
```

Q2) Write a program to define a function reversestr (string1, string2) which receives two strings as parameter and find their reverse, if both the reverse has the same length and start with ‘A’, return True else False.



```python
l = []

def reversestr(str1, str2):
    if len(str1[::-1]) == len(str2[::-1]) ∧ (str1[-1], str2[-1]) == ('a', 'a'):
        return True
    return False

for i in range(2):
    x = input(f"Enter string {i+1}: ")
    l.append(x)

print(reversestr(l[0], l[1]))
```

Q3) Write a program to define a function stringtuple(tuple) to pass a string of words and create a tuple having length of each and every word. Return this tuple from the function and display in main function.



```
def stringtuple(tup):
    word = list()
    x = tup.split()
    for i in x:
        word.append(len(i))
    return tuple(word)

sample = input("Enter string: ")

print(stringtuple(sample))
```

Q4) Write a program to define a dictionary STUDENT having scholarno and name of the students. The dictionary is empty initially. Write the functions to have the following functionalities and should be invoked in a menu driven option.:

1) Add the record in a dictionary, if scholar number doesn't exist, then only it should be added.

2) Ask the scholar number to modify the record, if scholar number existing ask the new data to be updated else display the message not found.

3) Ask the scholar number to delete the record, if scholar number does not exist, display the

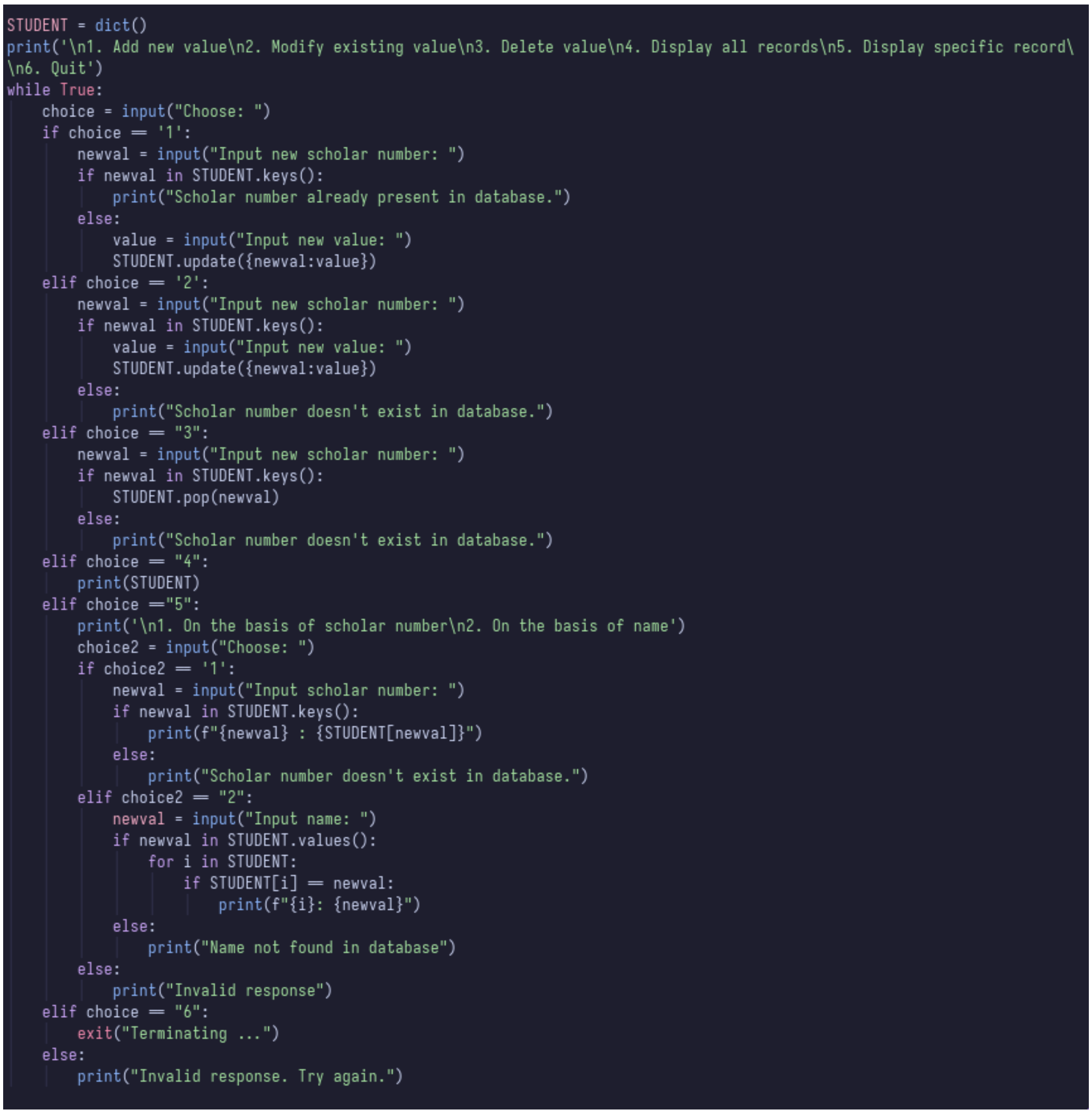
appropriate message, else delete.

4) Display all the records in dictionary.

5) Display the specific record

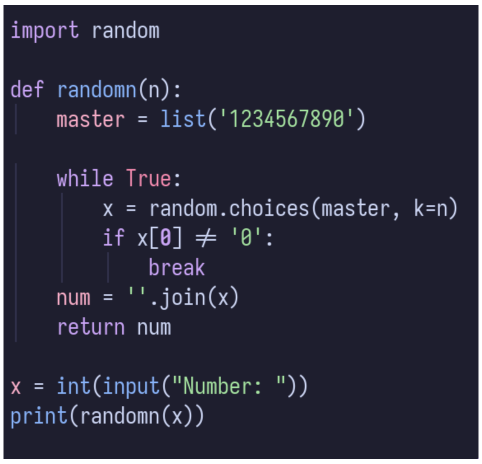
    i. On the basis of scholar number

    ii. On the basis of name



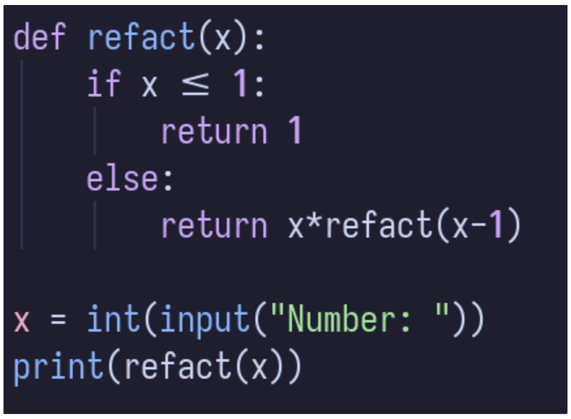
```python
STUDENT = dict()
print('\n1. Add new value\n2. Modify existing value\n3. Delete value\n4. Display all records\n5. Display specific record\
\n6. Quit')
while True:
    choice = input("Choose: ")
    if choice == '1':
        newval = input("Input new scholar number: ")
        if newval in STUDENT.keys():
            print("Scholar number already present in database.")
        else:
            value = input("Input new value: ")
            STUDENT.update({newval:value})
    elif choice == '2':
        newval = input("Input new scholar number: ")
        if newval in STUDENT.keys():
            value = input("Input new value: ")
            STUDENT.update({newval:value})
        else:
            print("Scholar number doesn't exist in database.")
    elif choice == "3":
        newval = input("Input new scholar number: ")
        if newval in STUDENT.keys():
            STUDENT.pop(newval)
        else:
            print("Scholar number doesn't exist in database.")
    elif choice == "4":
        print(STUDENT)
    elif choice =="5":
        print('\n1. On the basis of scholar number\n2. On the basis of name')
        choice2 = input("Choose: ")
        if choice2 == '1':
            newval = input("Input scholar number: ")
            if newval in STUDENT.keys():
                print(f"{newval} : {STUDENT[newval]}")
            else:
                print("Scholar number doesn't exist in database.")
        elif choice2 == "2":
            newval = input("Input name: ")
            if newval in STUDENT.values():
                for i in STUDENT:
                    if STUDENT[i] == newval:
                        print(f"{i}: {newval}")
            else:
                print("Name not found in database")
        else:
            print("Invalid response")
    elif choice == "6":
        exit("Terminating ...")
    else:
        print("Invalid response. Try again.")
```

Q5) Write a program to define a function randomn () to take an input n and return a randomly generated number having exactly n digits (not starting with zero). for example, if n is 2, function can generate any number between 10 to 99 but 07 type numbers are not allowed.



```
import random

def randomn(n):
    master = list('1234567890')

    while True:
        x = random.choices(master, k=n)
        if x[0] ≠ '0':
            break
    num = ''.join(x)
    return num

x = int(input("Number: "))
print(randomn(x))
```

Q6) Write a program to define a function recfact(number) to pass an integer as an argument to function and return the factorial of that number using recursion method.



```
def refact(x):
    if x ≤ 1:
        return 1
    else:
        return x*refact(x-1)

x = int(input("Number: "))
print(refact(x))
```

Q7) Write a program to define a module lengthconversion.py that stores functions for various length conversions:

a. miletokm ()

b. kmtomil ()

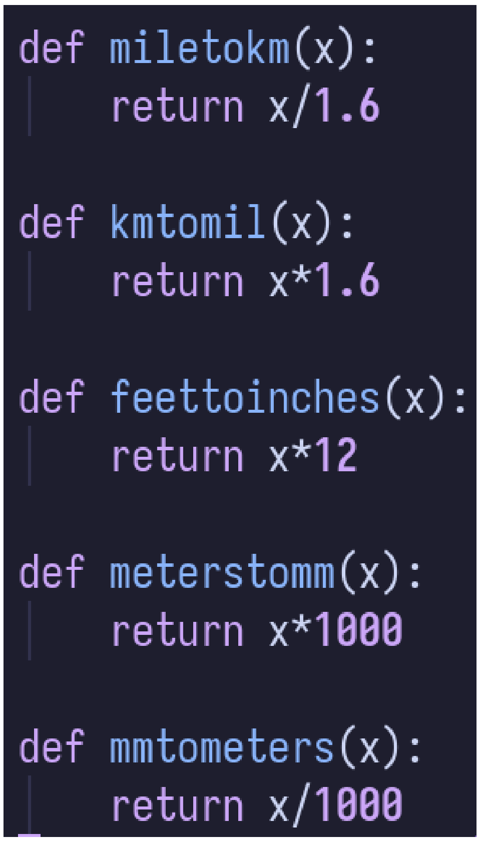
c. feettoinches ()

d. inchestofeet ()

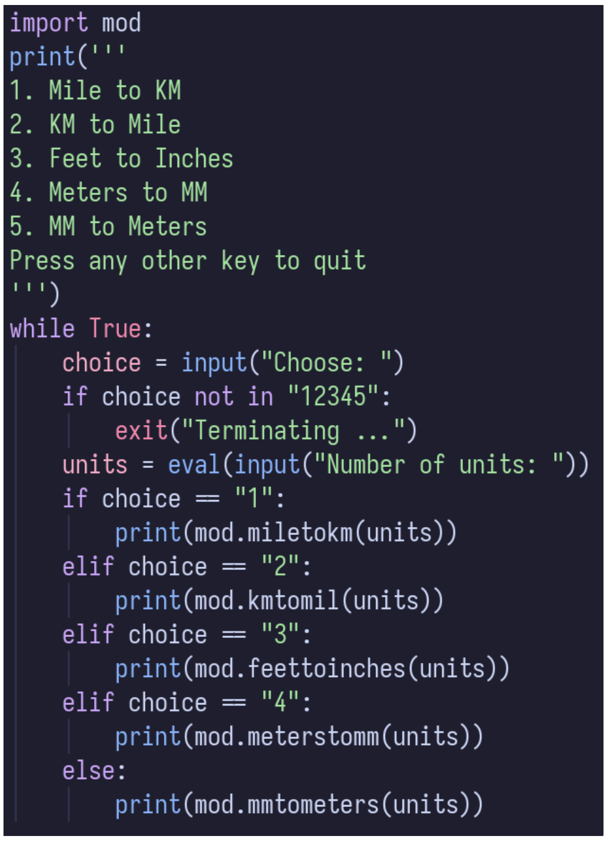
e. meterstomm ()

f. mmtometers ()

constants can also be stored, write another program where this module to be imported to convert the lengths as required. The program must be the menu driven program to display all the options and should invoke appropriate functions from the module.

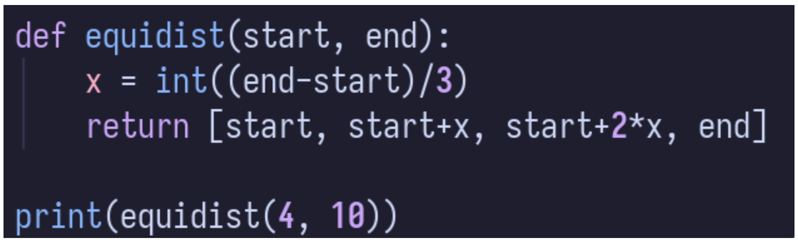


```python
def miletokm(x):
    return x/1.6

def kmtomil(x):
    return x*1.6

def feettoinches(x):
    return x*12

def meterstomm(x):
    return x*1000

def mmtometers(x):
    return x/1000
```



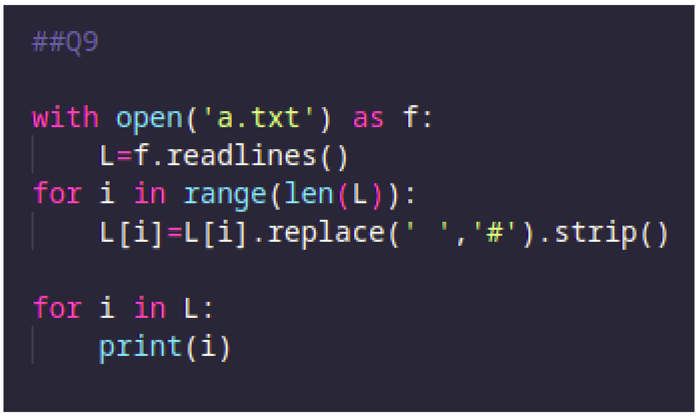
```python
import mod
print('''
1. Mile to KM
2. KM to Mile
3. Feet to Inches
4. Meters to MM
5. MM to Meters
Press any other key to quit
''')
while True:
    choice = input("Choose: ")
    if choice not in "12345":
        exit("Terminating ...")
    units = eval(input("Number of units: "))
    if choice == "1":
        print(mod.miletokm(units))
    elif choice == "2":
        print(mod.kmtomil(units))
    elif choice == "3":
        print(mod.feettoinches(units))
    elif choice == "4":
        print(mod.meterstomm(units))
    else:
        print(mod.mmtometers(units))
```

Q8) Write a program that generates a series using a function which takes first and last values of the series and thengenerates four terms that are equidistant example if two numbers passed are 4 and 10, it should return a list of [4,6,8,10]



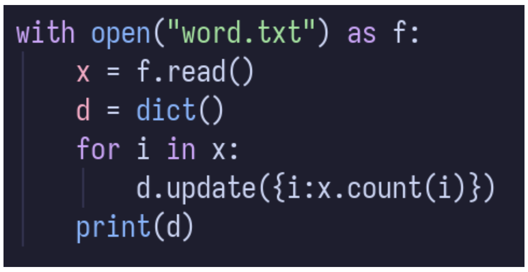
```python
def equidist(start, end):
    x = int((end-start)/3)
    return [start, start+x, start+2*x, end]

print(equidist(4, 10))
```

Q9) Write a program to read a text file a1.txt, line by line and display each word separated by a #.



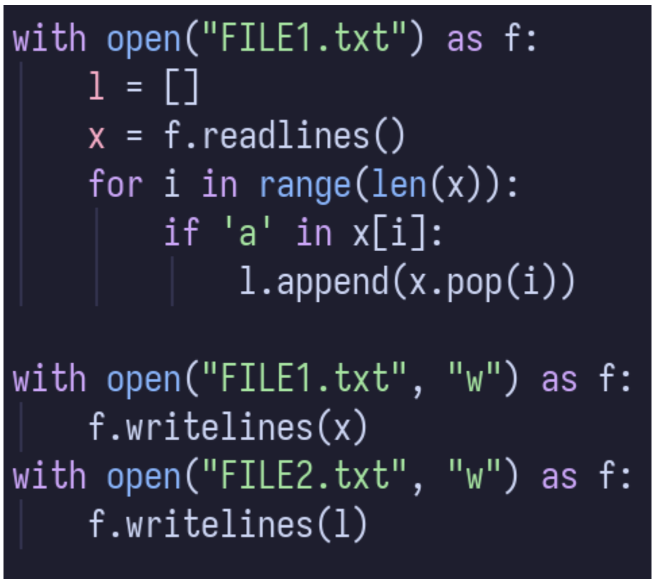
```python
##Q9

with open('a.txt') as f:
    L=f.readlines()
for i in range(len(L)):
    L[i]=L[i].replace(' ','#').strip()

for i in L:
    print(i)
```

Q10) Write a program to read a text file word.txt and display the number of spaces/words/digits vowels/ consonants/ uppercase/ lowercase characters in the file.



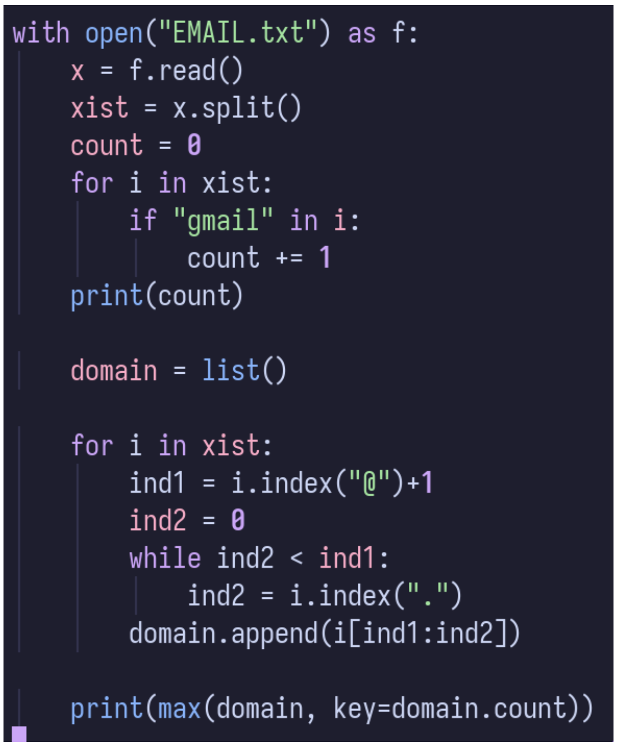
```python
with open("word.txt") as f:
    x = f.read()
    d = dict()
    for i in x:
        d.update({i:x.count(i)})
    print(d)
```

Q11) Write a program to read a file FILE1.TXT , Remove all the lines that contain the character `a' and write those to another file FILE2.TXT.



```python
with open("FILE1.txt") as f:
    l = []
    x = f.readlines()
    for i in range(len(x)):
        if 'a' in x[i]:
            l.append(x.pop(i))

with open("FILE1.txt", "w") as f:
    f.writelines(x)
with open("FILE2.txt", "w") as f:
    f.writelines(l)
```

Q12) Write a program to read a text file EMAIL.TXT and display the number of gmail ids .Also the program should display the most occurring word in the file.

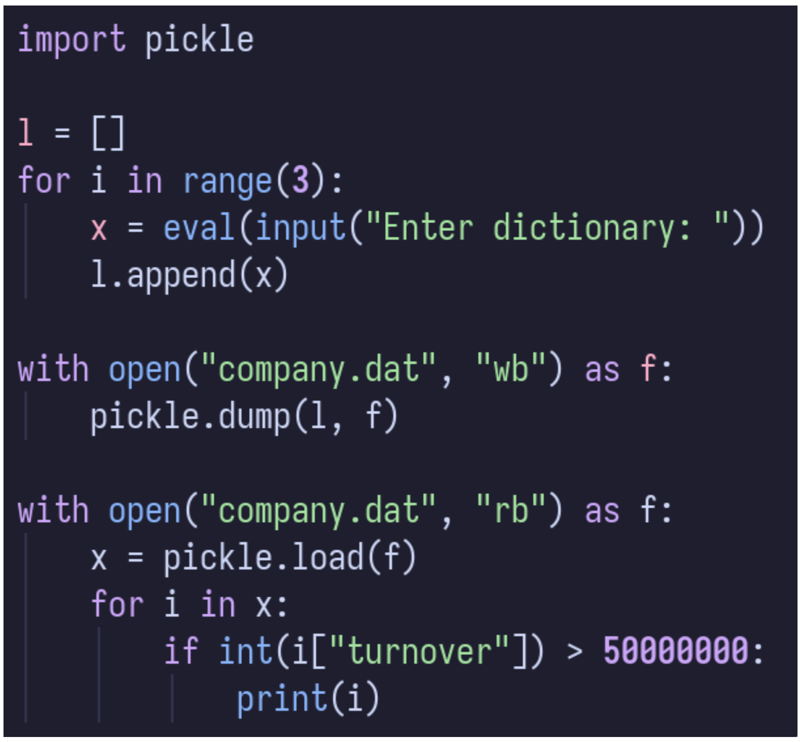


```python
with open("EMAIL.txt") as f:
    x = f.read()
    xist = x.split()
    count = 0
    for i in xist:
        if "gmail" in i:
            count += 1
    print(count)

    domain = list()

    for i in xist:
        ind1 = i.index("@")+1
        ind2 = 0
        while ind2 < ind1:
            ind2 = i.index(".")
        domain.append(i[ind1:ind2])

    print(max(domain, key=domain.count))
```

Q13) Considering the dictionary Company { ‘Company name ’ : , ‘Turnover’ : }

i. Take the input from the user for 3 companies and write in COMPANY.DAT

II. Display those members where TURNOVER of the company is more than 5 Crores.



```python
import pickle

l = []
for i in range(3):
    x = eval(input("Enter dictionary: "))
    l.append(x)

with open("company.dat", "wb") as f:
    pickle.dump(l, f)

with open("company.dat", "rb") as f:
    x = pickle.load(f)
    for i in x:
        if int(i["turnover"]) > 50000000:
            print(i)
```

14) Write a program to display the following:

Menu driven program

1. Add data

2. Read entire file

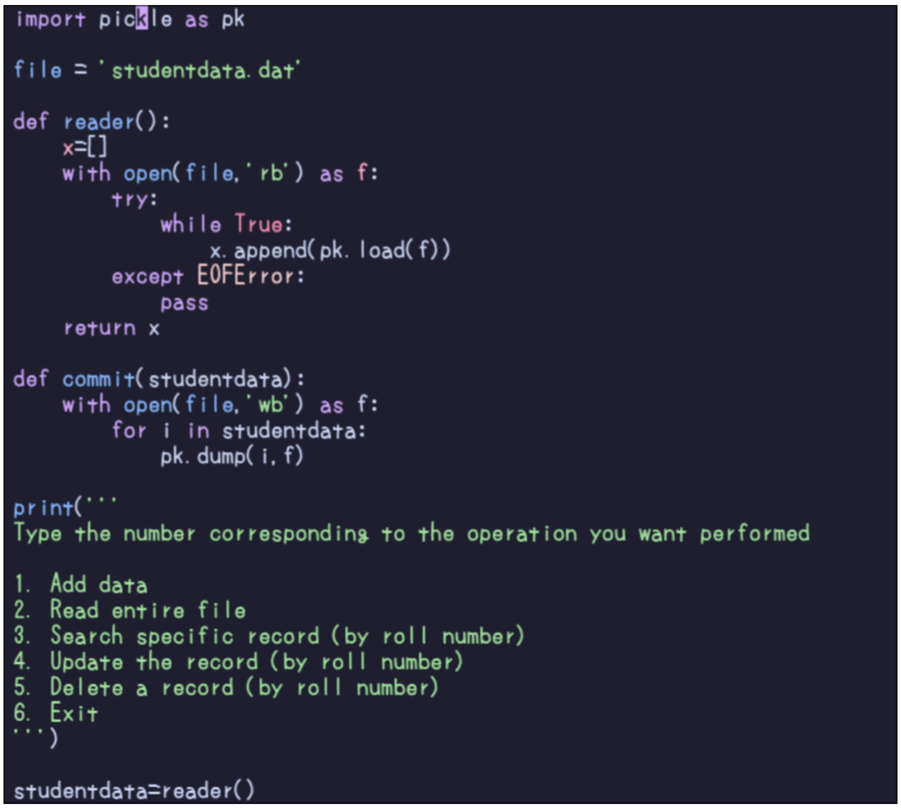
3. Search specific record (ask for the roll number)

4. Update the record (ask for the roll number)

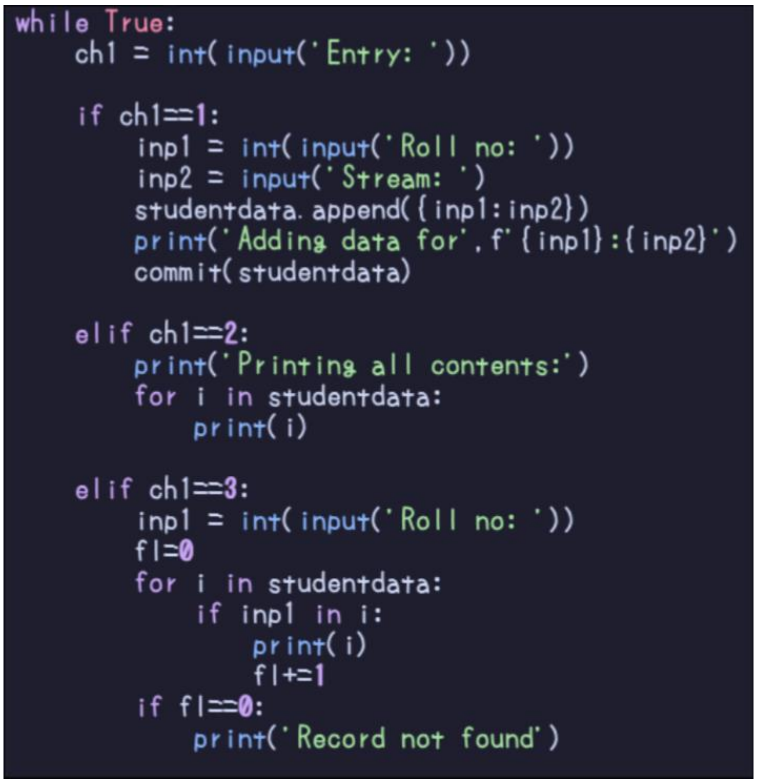
5. Delete

6. Exit

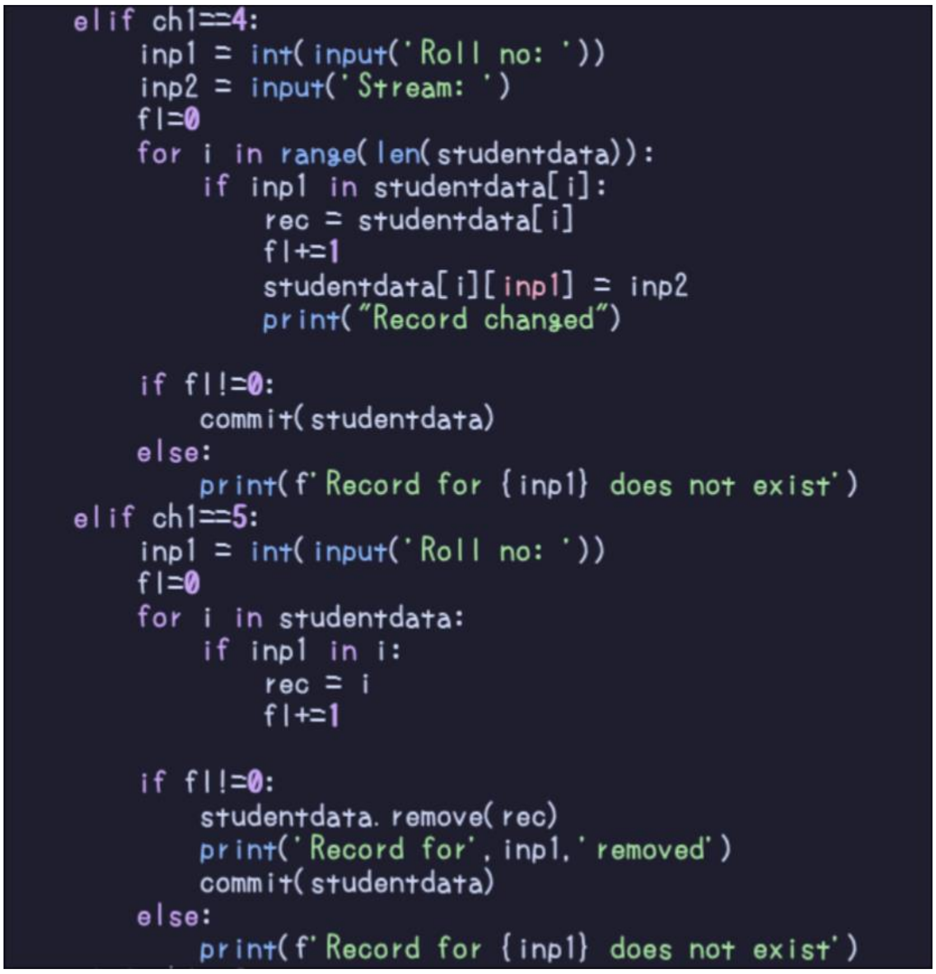
The above should be able to perform the functionality on a dictionary STUDENTDATA(having roll number and stream). Also end of file and file not found error should be handled.



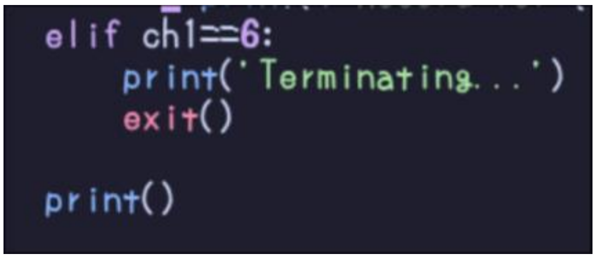
```
import pickle as pk

file = 'studentdata.dat'

def reader():
    x=[]
    with open(file,'rb') as f:
        try:
            while True:
                x.append(pk.load(f))
        except EOFError:
            pass
    return x

def commit(studentdata):
    with open(file,'wb') as f:
        for i in studentdata:
            pk.dump(i,f)

print('''
Type the number corresponding to the operation you want performed

1. Add data
2. Read entire file
3. Search specific record (by roll number)
4. Update the record (by roll number)
5. Delete a record (by roll number)
6. Exit
''')

studentdata=reader()
```



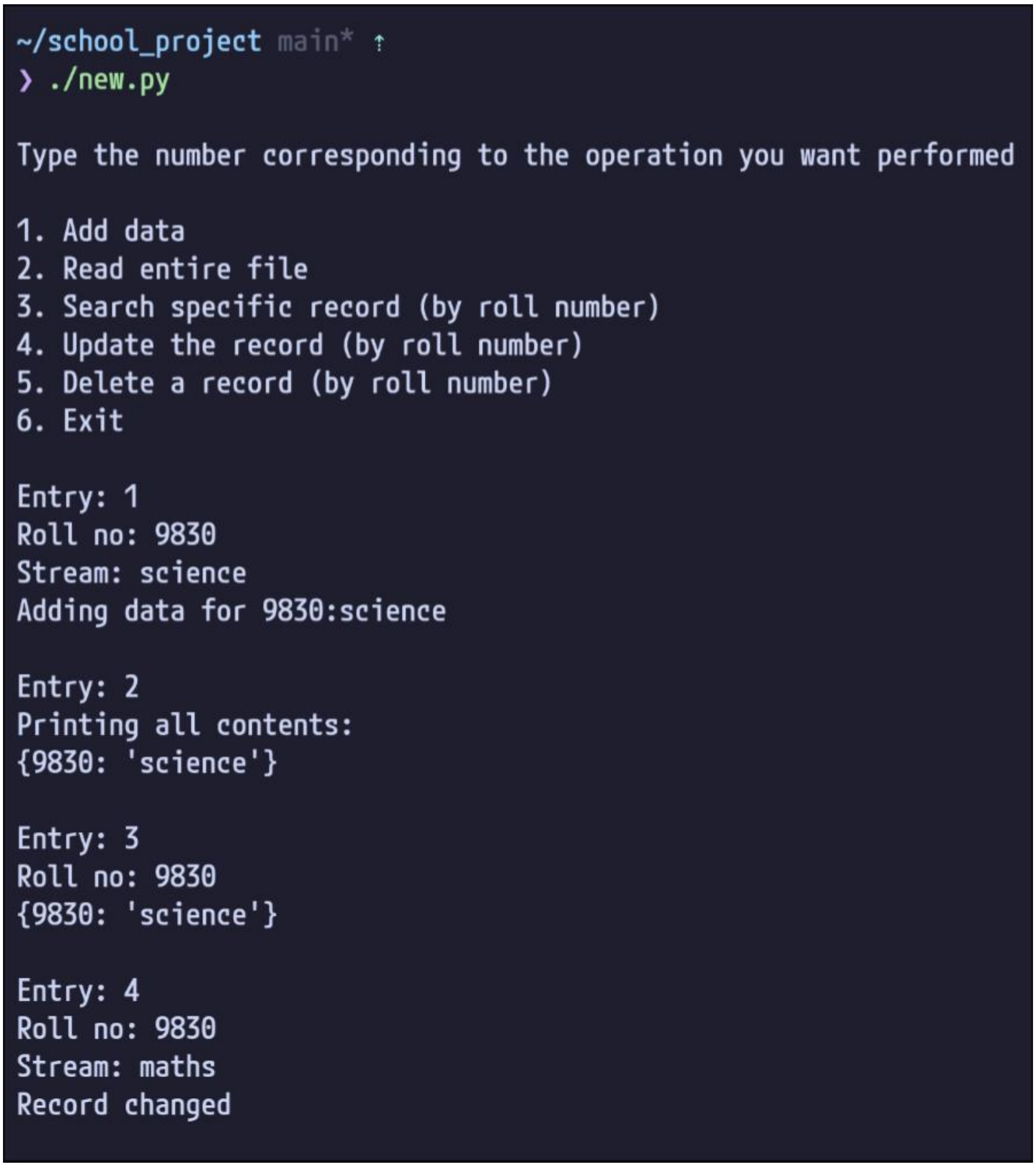
```
while True:
    ch1 = int(input('Entry: '))

    if ch1==1:
        inp1 = int(input('Roll no: '))
        inp2 = input('Stream: ')
        studentdata.append({inp1:inp2})
        print('Adding data for',f'{inp1}:{inp2}')
        commit(studentdata)

    elif ch1==2:
        print('Printing all contents:')
        for i in studentdata:
            print(i)

    elif ch1==3:
        inp1 = int(input('Roll no: '))
        fl=0
        for i in studentdata:
            if inp1 in i:
                print(i)
                fl+=1
        if fl==0:
            print('Record not found')
```



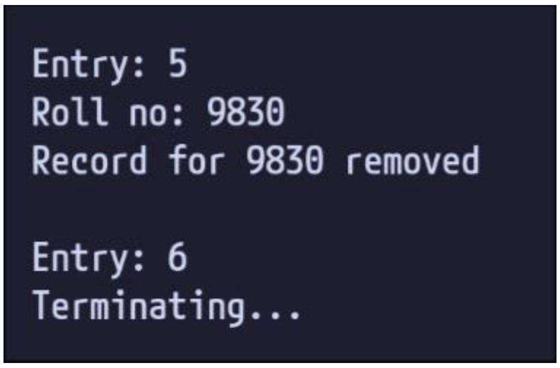
```
    elif ch1==4:
        inp1 = int(input('Roll no: '))
        inp2 = input('Stream: ')
        fl=0
        for i in range(len(studentdata)):
            if inp1 in studentdata[i]:
                rec = studentdata[i]
                fl+=1
                studentdata[i][inp1] = inp2
                print("Record changed")

        if fl!=0:
            commit(studentdata)
        else:
            print(f'Record for {inp1} does not exist')
    elif ch1==5:
        inp1 = int(input('Roll no: '))
        fl=0
        for i in studentdata:
            if inp1 in i:
                rec = i
                fl+=1

        if fl!=0:
            studentdata.remove(rec)
            print('Record for', inp1, 'removed')
            commit(studentdata)
        else:
            print(f'Record for {inp1} does not exist')
```



```
elif ch1==6:
    print('Terminating...')
    exit()

print()
```

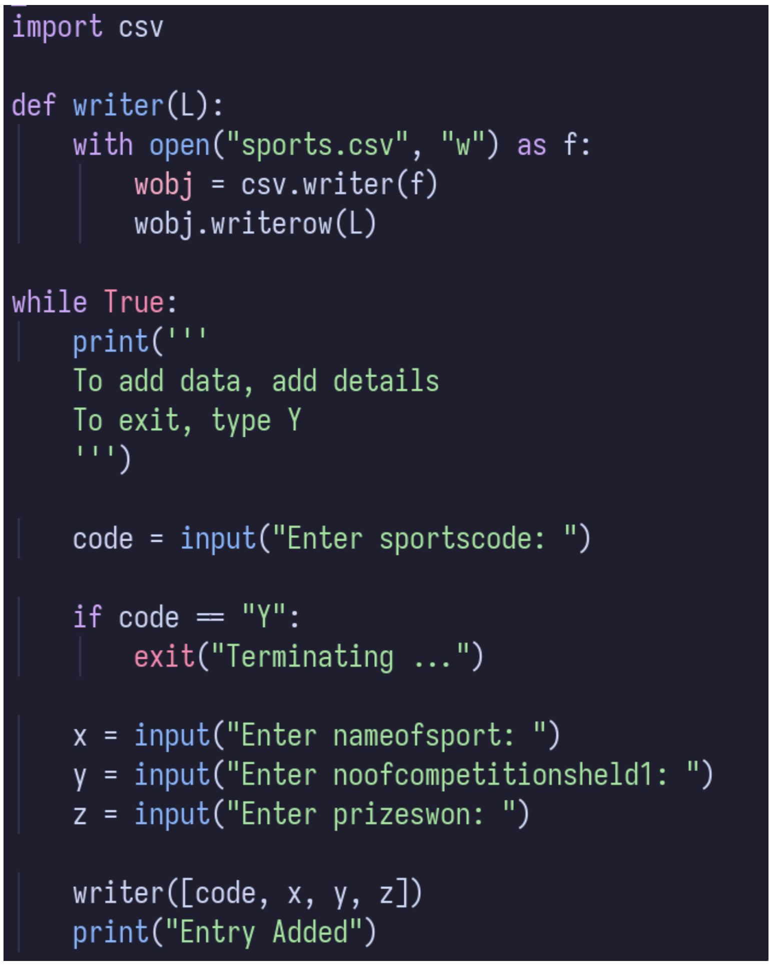


```
~/school_project main* ↑
❯ ./new.py

Type the number corresponding to the operation you want performed

1. Add data
2. Read entire file
3. Search specific record (by roll number)
4. Update the record (by roll number)
5. Delete a record (by roll number)
6. Exit

Entry: 1
Roll no: 9830
Stream: science
Adding data for 9830:science

Entry: 2
Printing all contents:
{9830: 'science'}

Entry: 3
Roll no: 9830
{9830: 'science'}

Entry: 4
Roll no: 9830
Stream: maths
Record changed
```



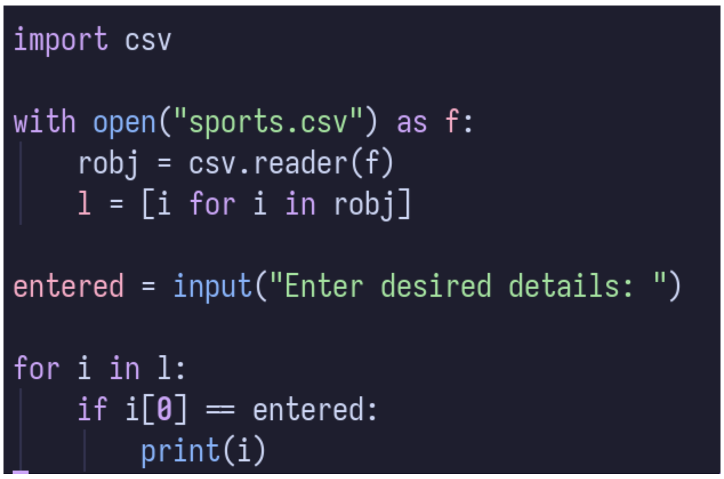
```
Entry: 5
Roll no: 9830
Record for 9830 removed

Entry: 6
Terminating...
```

Q15) Write a program to create a csv file sports.csv (through the list) to take inputs sportscode, nameofthesport, noofcompetitionsheld and prizeswon. Ask for the choice from the user and keep adding till user enters ‘Y’ as choice.



```python
import csv

def writer(L):
    with open("sports.csv", "w") as f:
        wobj = csv.writer(f)
        wobj.writerow(L)

while True:
    print('''
    To add data, add details
    To exit, type Y
    ''')

    code = input("Enter sportscode: ")

    if code == "Y":
        exit("Terminating ...")

    x = input("Enter nameofsport: ")
    y = input("Enter noofcompetitionsheld1: ")
    z = input("Enter prizeswon: ")

    writer([code, x, y, z])
    print("Entry Added")
```

Q16) Considering the same file above in question 15, display all the records from the file where nameofthesport is entered by the user.



```python
import csv

with open("sports.csv") as f:
    robj = csv.reader(f)
    l = [i for i in robj]

entered = input("Enter desired details: ")

for i in l:
    if i[0] == entered:
        print(i)
```

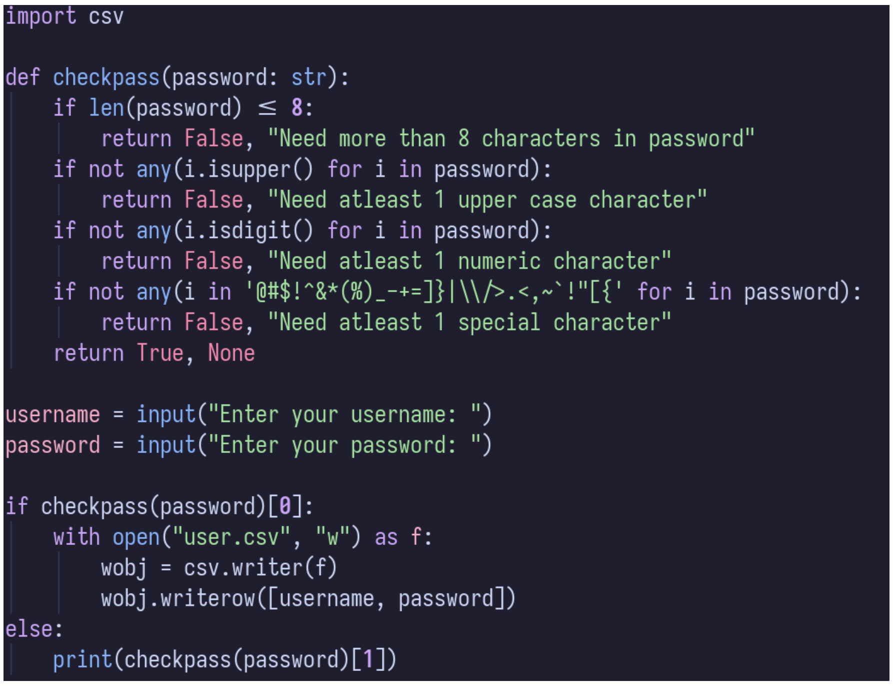
Q17) Jacqline of class 12 is writing a program to create a CSV file “user.csv” which will contain username and password for some entries. While taking input from the user, password should be checked for the following conditions:

1. at least 8 characters long

2. should have one upper case character

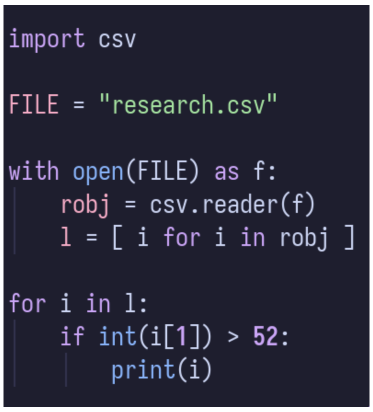
3. should have at least one digit.

4. should have one special character.



```python
import csv

def checkpass(password: str):
    if len(password) ≤ 8:
        return False, "Need more than 8 characters in password"
    if not any(i.isupper() for i in password):
        return False, "Need atleast 1 upper case character"
    if not any(i.isdigit() for i in password):
        return False, "Need atleast 1 numeric character"
    if not any(i in '@#$!^&*(%)_-+=]}|\\/>.<,~`!"[{' for i in password):
        return False, "Need atleast 1 special character"
    return True, None

username = input("Enter your username: ")
password = input("Enter your password: ")

if checkpass(password)[0]:
    with open("user.csv", "w") as f:
        wobj = csv.writer(f)
        wobj.writerow([username, password])
else:
    print(checkpass(password)[1])
```

Q18) Professor Islam has told students to create a csv file having research id and data collected (numeric) from an experiment. The file should have at least 5 records. Write a program to read a file RESEARCH.CSV created and display only those records where data collected is more than 52.



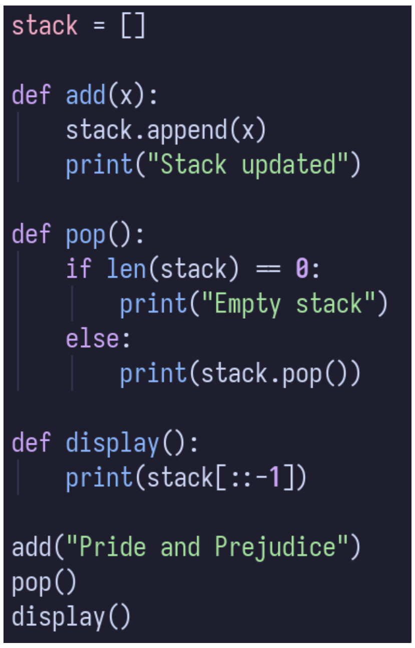
```python
import csv

FILE = "research.csv"

with open(FILE) as f:
    robj = csv.reader(f)
    l = [ i for i in robj ]

for i in l:
    if int(i[1]) > 52:
        print(i)
```

Q19) Considering a list of book details (bookno and name) , write a program to implement STACK defining following three functions:

1. Add a book in stack

2. Delete a book from stack

3. Display the elements of stack.



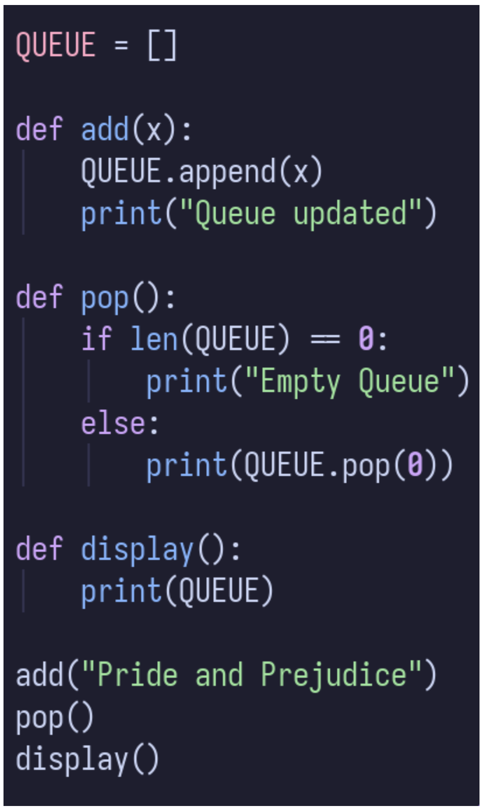
```python
stack = []

def add(x):
    stack.append(x)
    print("Stack updated")

def pop():
    if len(stack) == 0:
        print("Empty stack")
    else:
        print(stack.pop())

def display():
    print(stack[::-1])

add("Pride and Prejudice")
pop()
display()
```

Q20) Considering a list of book details (bookno and name) , write a program to implement QUEUE defining following three functions:

1. Add a book in a queue

2. Delete a book from queue

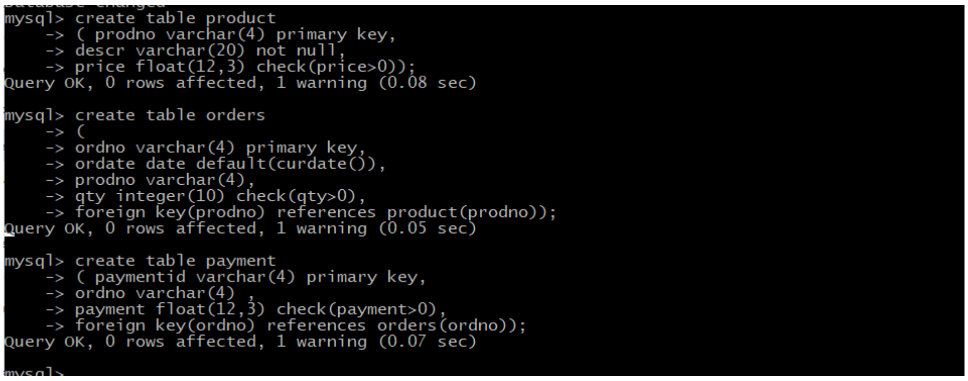
3. Display the elements of queue



```python
QUEUE = []

def add(x):
    QUEUE.append(x)
    print("Queue updated")

def pop():
    if len(QUEUE) == 0:
        print("Empty Queue")
    else:
        print(QUEUE.pop(0))

def display():
    print(QUEUE)

add("Pride and Prejudice")
pop()
display()
```

SECTION II : DATABASE MANAGEMENT AND SQL

Q2-A : Create the following three tables in the database FACTORY and perform the operations asked :



```
mysql> create table product
    -> ( prodno varchar(4) primary key,
    -> descr varchar(20) not null,
    -> price float(12,3) check(price>0));
Query OK, 0 rows affected, 1 warning (0.08 sec)

mysql> create table orders
    -> (
    -> ordno varchar(4) primary key,
    -> ordate date default(curdate()),
    -> prodno varchar(4),
    -> qty integer(10) check(qty>0),
    -> foreign key(prodno) references product(prodno));
Query OK, 0 rows affected, 1 warning (0.05 sec)

mysql> create table payment
    -> ( paymentid varchar(4) primary key,
    -> ordno varchar(4) ,
    -> payment float(12,3) check(payment>0),
    -> foreign key(ordno) references orders(ordno));
Query OK, 0 rows affected, 1 warning (0.07 sec)
```

1. Display the tables existing in the database.

2. Display the structure of all the tables.

2.1 Insert at least three rows in all three tables.

3. Display the data from product.

4. Display the data from order table.

5. Display the data from payment table.

6. Display payment id and payment from payment table.

7. Display order no , order date , qty from order table.

8. Display the discounted price of all products from product table where discount is 5% of the price.

9. Display the data from payment table , arranged on payment id.

10. Display the data from order table , arranged in descending order on order id.

11. Display those order details from order table where qty is more than 10.

12. Display those payment details order id starts with O and ends with P.

13. Display those product details where price is between 1000 to 2000(both included)

14. Display those orders which were placed in the month of February 2020.

15. Display those orders where products are ‘P01’, ‘P03’,’P05’ (using in operator)16. Display those orders where products are ‘P01’, ‘P03’,’P05’ (using or operator)

17. Display all the orders from payment table (orders shouldn’t be duplicated)

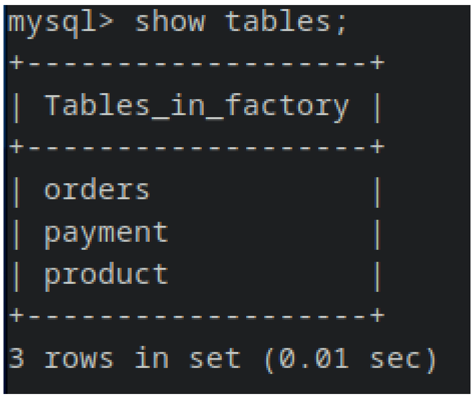
18. Display those orders which are placed today.

19. Display those products where description is more than 30 characters long.

20. Display prices of product, rounded to one place.

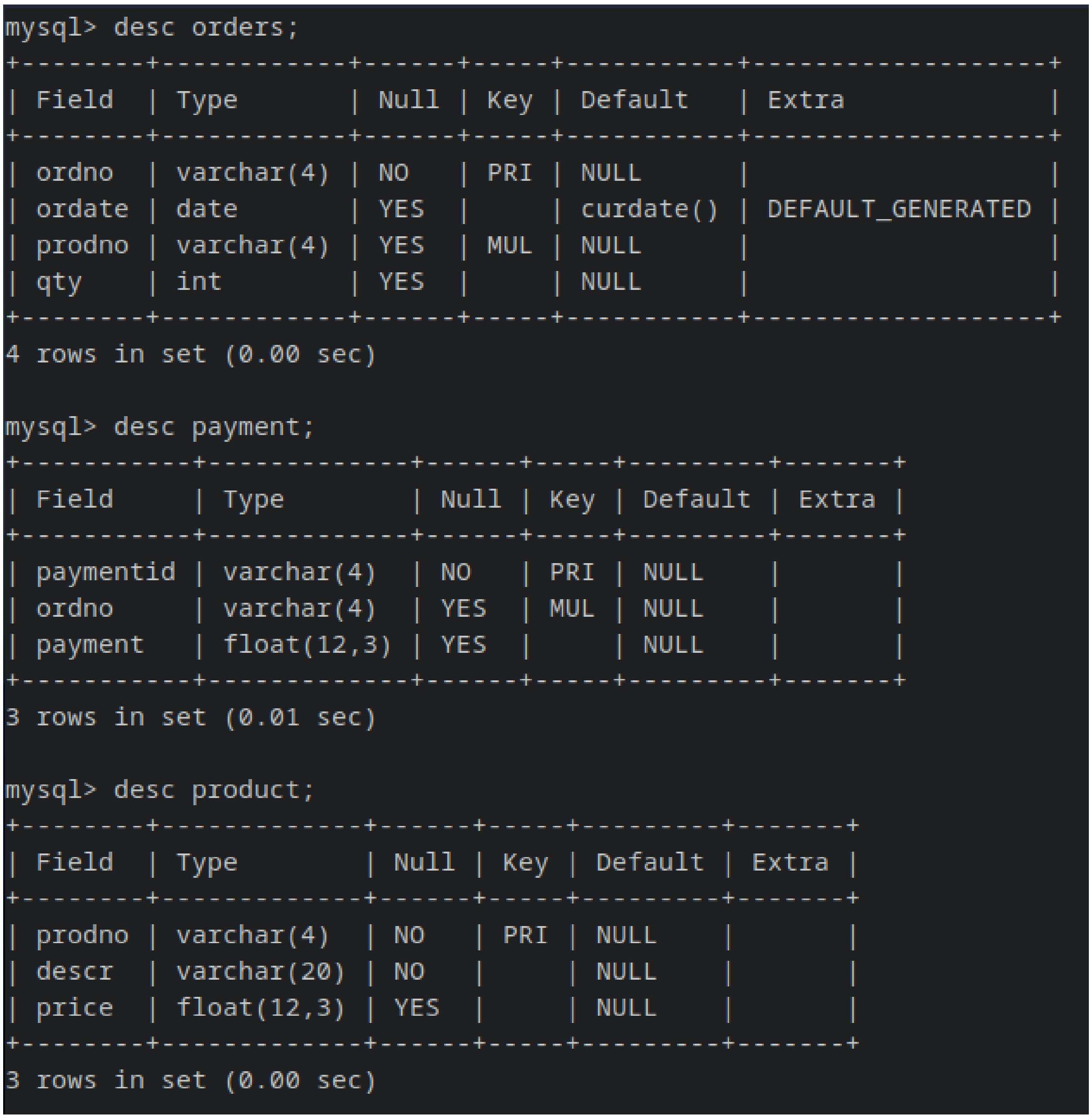
Answers:

Q1)



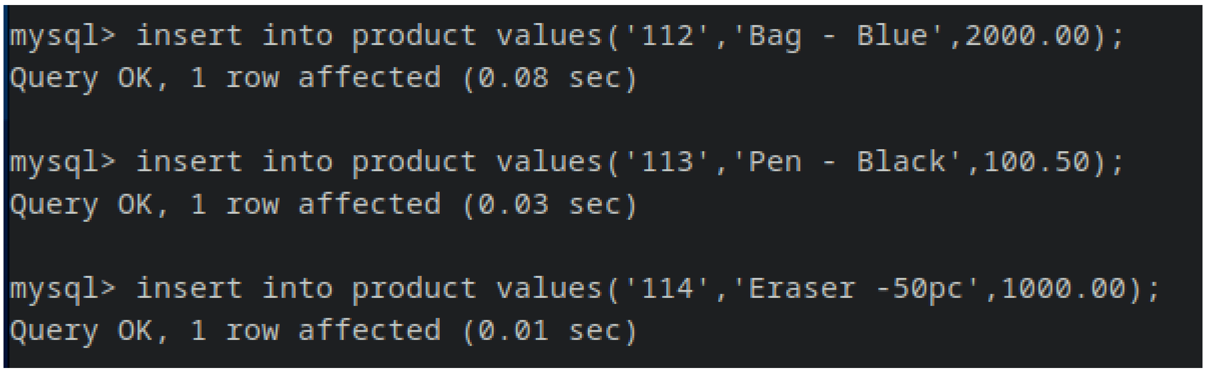
```
mysql> show tables;
+-------------------+
| Tables_in_factory |
+-------------------+
| orders            |
| payment           |
| product           |
+-------------------+
3 rows in set (0.01 sec)
```

Q2)

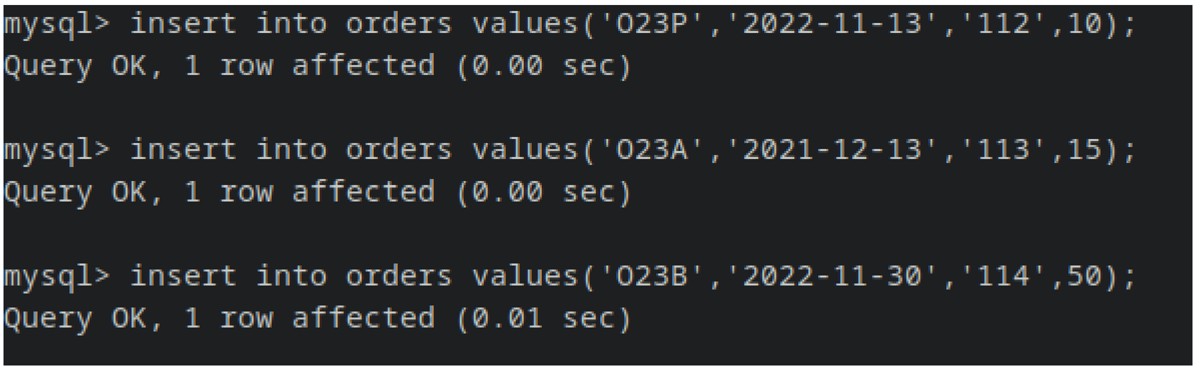


```
mysql> desc orders;
+--------+------------+------+-----+-----------+-------------------+
| Field  | Type       | Null | Key | Default   | Extra             |
+--------+------------+------+-----+-----------+-------------------+
| ordno  | varchar(4) | NO   | PRI | NULL      |                   |
| ordate | date       | YES  |     | curdate() | DEFAULT_GENERATED |
| prodno | varchar(4) | YES  | MUL | NULL      |                   |
| qty    | int        | YES  |     | NULL      |                   |
+--------+------------+------+-----+-----------+-------------------+
4 rows in set (0.00 sec)

mysql> desc payment;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| paymentid | varchar(4)  | NO   | PRI | NULL    |       |
| ordno     | varchar(4)  | YES  | MUL | NULL    |       |
| payment   | float(12,3) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
3 rows in set (0.01 sec)

mysql> desc product;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| prodno | varchar(4)  | NO   | PRI | NULL    |       |
| descr  | varchar(20) | NO   |     | NULL    |       |
| price  | float(12,3) | YES  |     | NULL    |       |
+--------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

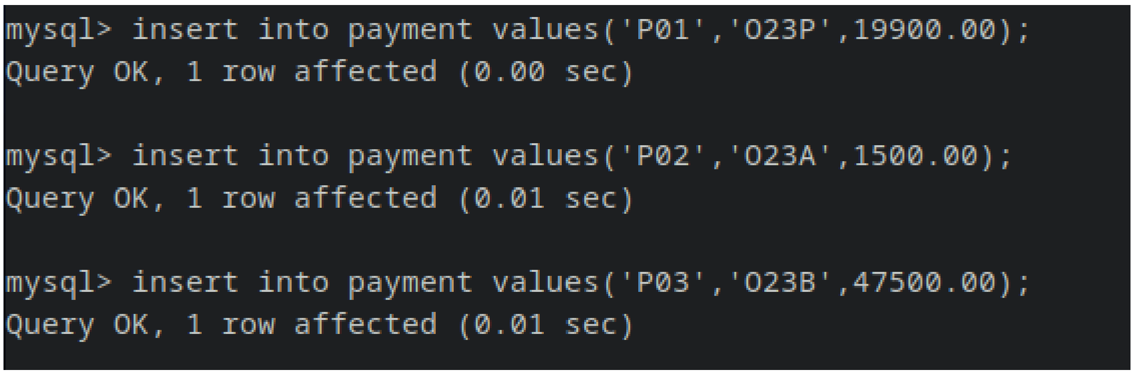
Q2.1)



```
mysql> insert into product values('112','Bag - Blue',2000.00);
Query OK, 1 row affected (0.08 sec)

mysql> insert into product values('113','Pen - Black',100.50);
Query OK, 1 row affected (0.03 sec)

mysql> insert into product values('114','Eraser -50pc',1000.00);
Query OK, 1 row affected (0.01 sec)
```

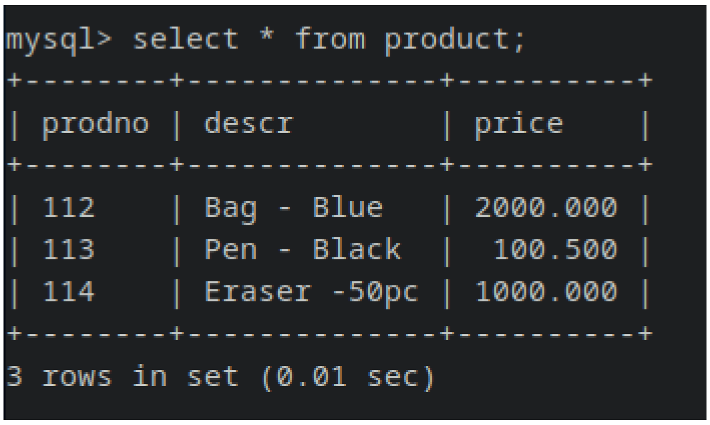


```
mysql> insert into orders values('O23P','2022-11-13','112',10);
Query OK, 1 row affected (0.00 sec)

mysql> insert into orders values('O23A','2021-12-13','113',15);
Query OK, 1 row affected (0.00 sec)

mysql> insert into orders values('O23B','2022-11-30','114',50);
Query OK, 1 row affected (0.01 sec)
```



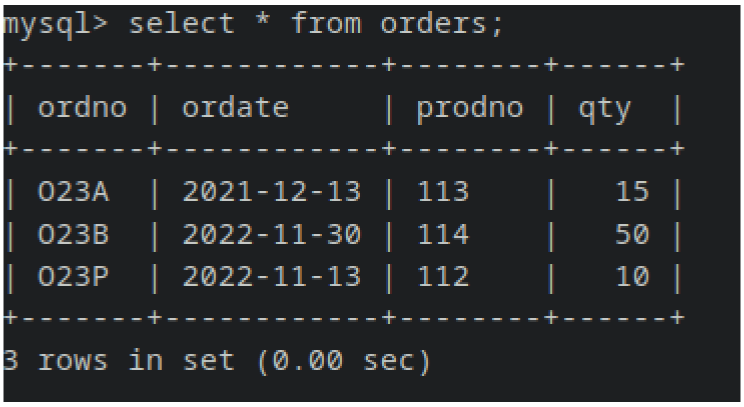
```
mysql> insert into payment values('P01','O23P',19900.00);
Query OK, 1 row affected (0.00 sec)

mysql> insert into payment values('P02','O23A',1500.00);
Query OK, 1 row affected (0.01 sec)

mysql> insert into payment values('P03','O23B',47500.00);
Query OK, 1 row affected (0.01 sec)
```

Q3)



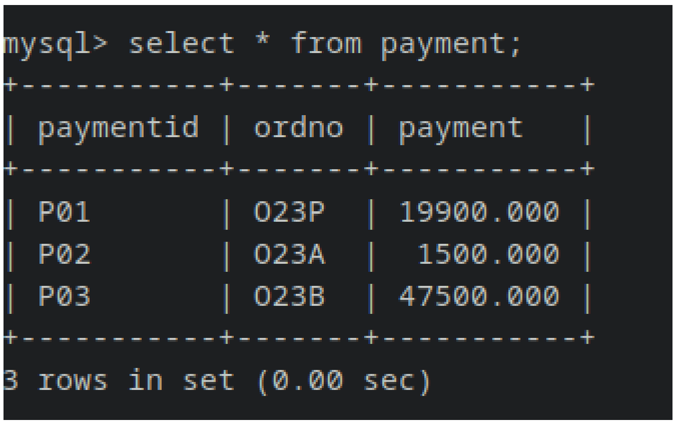
```
mysql> select * from product;
+--------+--------------+----------+
| prodno | descr        | price    |
+--------+--------------+----------+
| 112    | Bag - Blue   | 2000.000 |
| 113    | Pen - Black  |  100.500 |
| 114    | Eraser -50pc | 1000.000 |
+--------+--------------+----------+
3 rows in set (0.01 sec)
```

Q4)



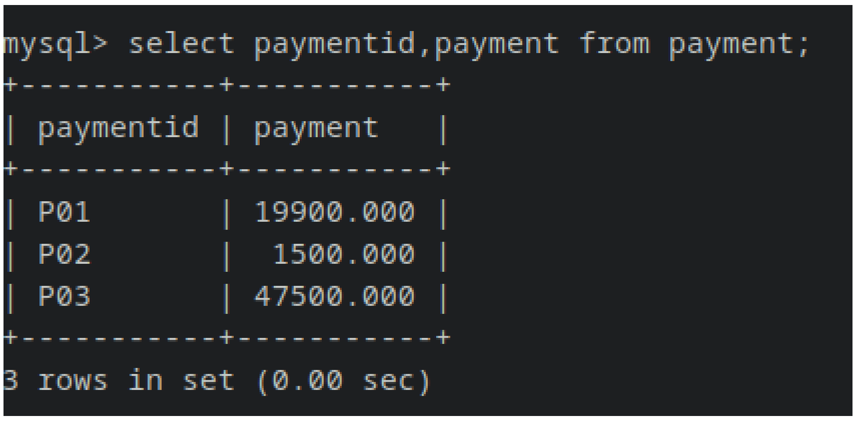
```
mysql> select * from orders;
+-------+------------+--------+------+
| ordno | ordate     | prodno | qty  |
+-------+------------+--------+------+
| O23A  | 2021-12-13 | 113    |   15 |
| O23B  | 2022-11-30 | 114    |   50 |
| O23P  | 2022-11-13 | 112    |   10 |
+-------+------------+--------+------+
3 rows in set (0.00 sec)
```

Q5)



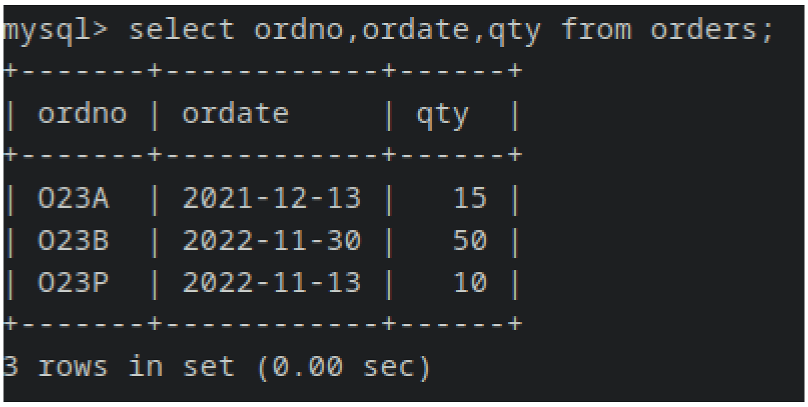
```
mysql> select * from payment;
+-----------+-------+-----------+
| paymentid | ordno | payment   |
+-----------+-------+-----------+
| P01       | O23P  | 19900.000 |
| P02       | O23A  |  1500.000 |
| P03       | O23B  | 47500.000 |
+-----------+-------+-----------+
3 rows in set (0.00 sec)
```

Q6)



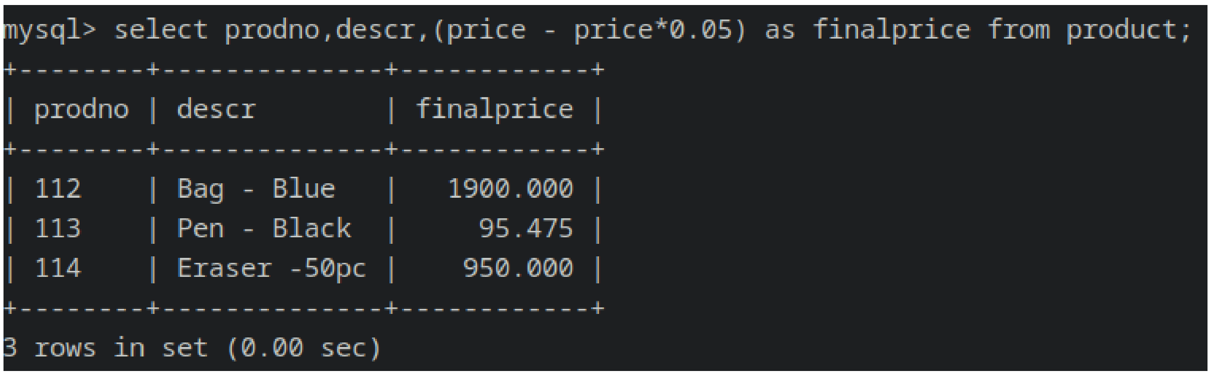
```
mysql> select paymentid,payment from payment;
+-----------+-----------+
| paymentid | payment   |
+-----------+-----------+
| P01       | 19900.000 |
| P02       |  1500.000 |
| P03       | 47500.000 |
+-----------+-----------+
3 rows in set (0.00 sec)
```

Q7)



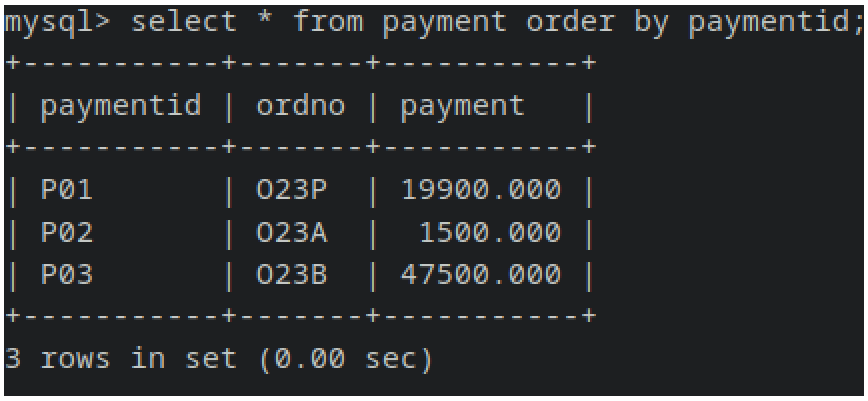
```
mysql> select ordno,ordate,qty from orders;
+-------+------------+------+
| ordno | ordate     | qty  |
+-------+------------+------+
| O23A  | 2021-12-13 |   15 |
| O23B  | 2022-11-30 |   50 |
| O23P  | 2022-11-13 |   10 |
+-------+------------+------+
3 rows in set (0.00 sec)
```

Q8)



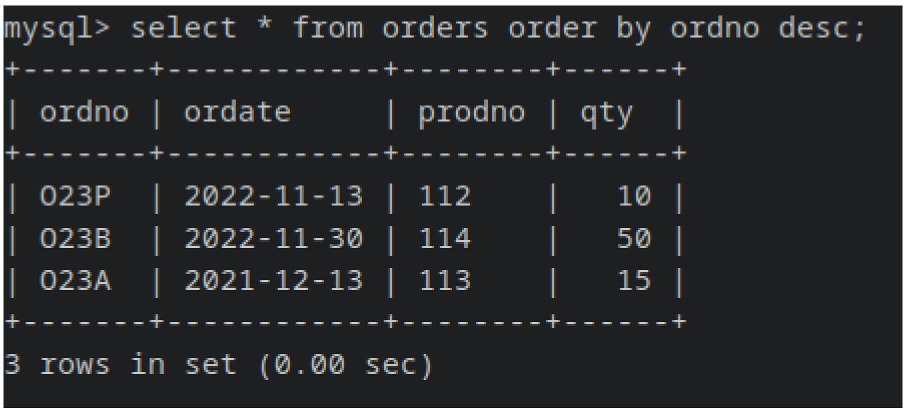
```
mysql> select prodno,descr,(price - price*0.05) as finalprice from product;
+--------+-------------+------------+
| prodno | descr       | finalprice |
+--------+-------------+------------+
| 112    | Bag - Blue  |   1900.000 |
| 113    | Pen - Black |     95.475 |
| 114    | Eraser -50pc |    950.000 |
+--------+-------------+------------+
3 rows in set (0.00 sec)
```

Q9)



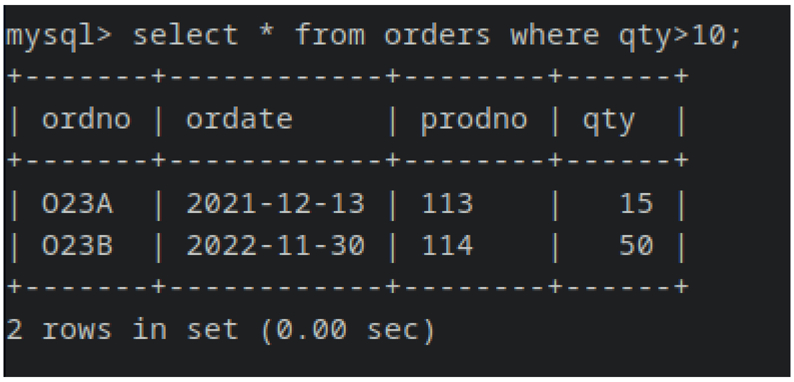
```
mysql> select * from payment order by paymentid;
+-----------+-------+-----------+
| paymentid | ordno | payment   |
+-----------+-------+-----------+
| P01       | O23P  | 19900.000 |
| P02       | O23A  |  1500.000 |
| P03       | O23B  | 47500.000 |
+-----------+-------+-----------+
3 rows in set (0.00 sec)
```

Q10)



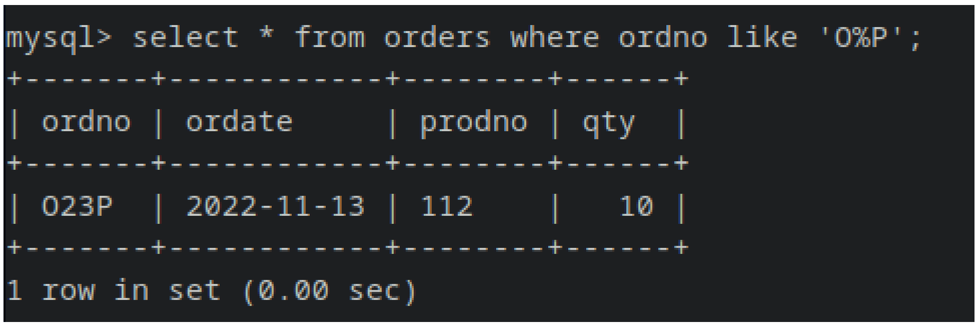
```
mysql> select * from orders order by ordno desc;
+-------+------------+--------+------+
| ordno | ordate     | prodno | qty  |
+-------+------------+--------+------+
| O23P  | 2022-11-13 | 112    |   10 |
| O23B  | 2022-11-30 | 114    |   50 |
| O23A  | 2021-12-13 | 113    |   15 |
+-------+------------+--------+------+
3 rows in set (0.00 sec)
```

Q11)



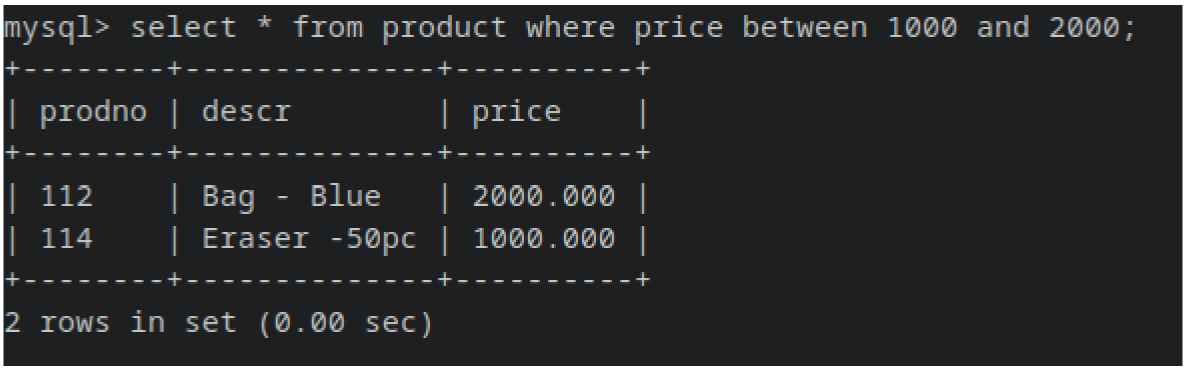
```
mysql> select * from orders where qty>10;
+-------+------------+--------+------+
| ordno | ordate     | prodno | qty  |
+-------+------------+--------+------+
| O23A  | 2021-12-13 | 113    |   15 |
| O23B  | 2022-11-30 | 114    |   50 |
+-------+------------+--------+------+
2 rows in set (0.00 sec)
```

Q12)



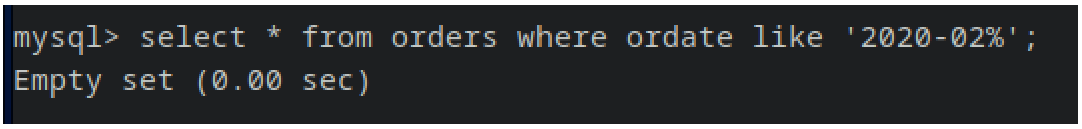
```
mysql> select * from orders where ordno like 'O%P';
+-------+------------+--------+------+
| ordno | ordate     | prodno | qty  |
+-------+------------+--------+------+
| O23P  | 2022-11-13 | 112    |   10 |
+-------+------------+--------+------+
1 row in set (0.00 sec)
```

Q13)



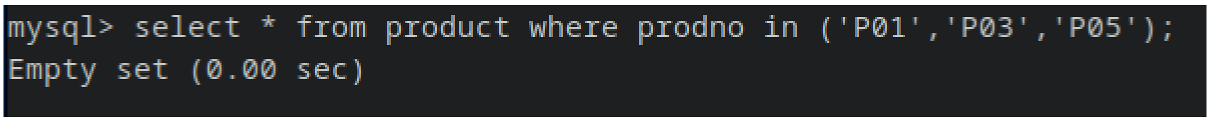
```
mysql> select * from product where price between 1000 and 2000;
+--------+--------------+----------+
| prodno | descr        | price    |
+--------+--------------+----------+
| 112    | Bag - Blue   | 2000.000 |
| 114    | Eraser -50pc | 1000.000 |
+--------+--------------+----------+
2 rows in set (0.00 sec)
```

Q14)



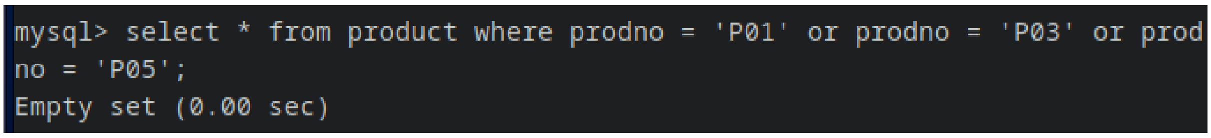
```
mysql> select * from orders where ordate like '2020-02%';
Empty set (0.00 sec)
```

Q15)



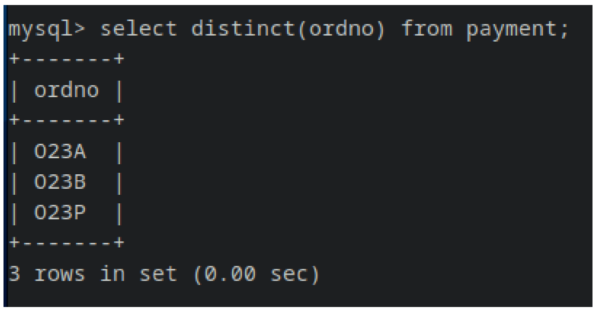
```
mysql> select * from product where prodno in ('P01','P03','P05');
Empty set (0.00 sec)
```

Q16)



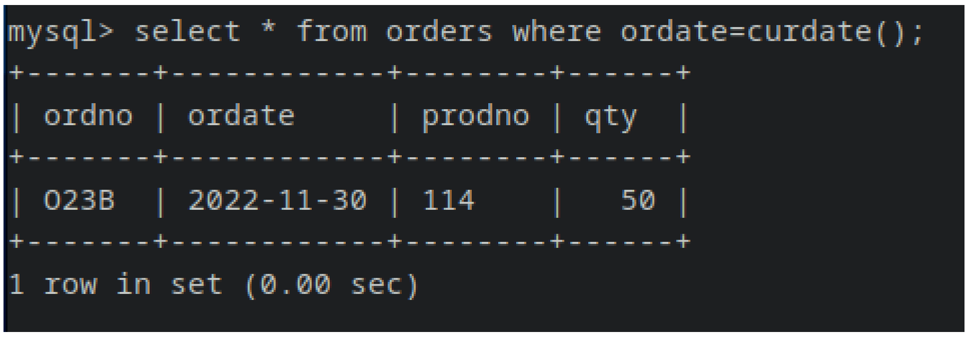
```
mysql> select * from product where prodno = 'P01' or prodno = 'P03' or prod
no = 'P05';
Empty set (0.00 sec)
```

Q17)



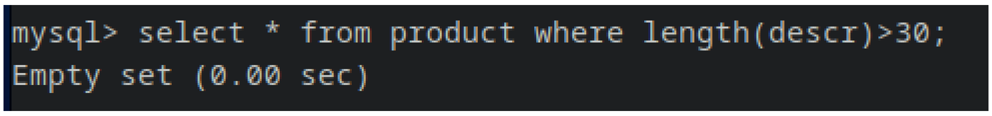
```
mysql> select distinct(ordno) from payment;
+-------+
| ordno |
+-------+
| O23A  |
| O23B  |
| O23P  |
+-------+
3 rows in set (0.00 sec)
```

Q18)



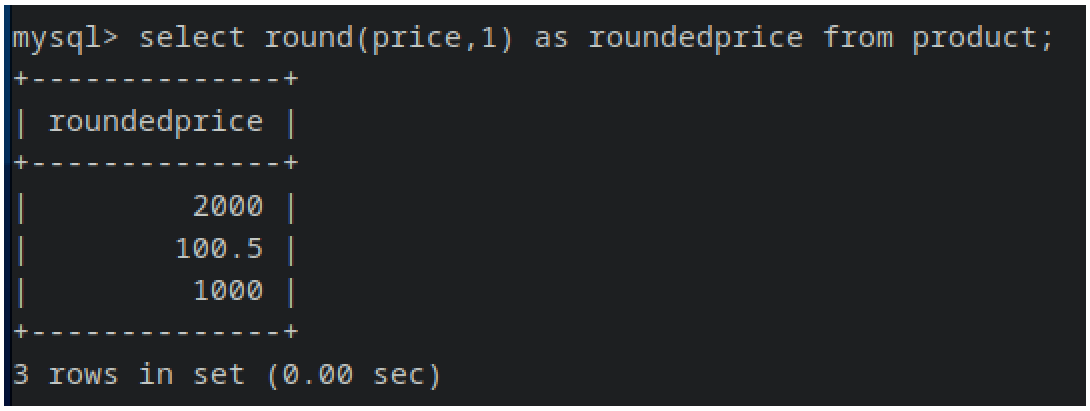
```
mysql> select * from orders where ordate=curdate();
+-------+------------+--------+------+
| ordno | ordate     | prodno | qty  |
+-------+------------+--------+------+
| O23B  | 2022-11-30 | 114    |   50 |
+-------+------------+--------+------+
1 row in set (0.00 sec)
```

Q19)



```
mysql> select * from product where length(descr)>30;
Empty set (0.00 sec)
```

Q20)



```
mysql> select round(price,1) as roundedprice from product;
+--------------+
| roundedprice |
+--------------+
|         2000 |
|        100.5 |
|         1000 |
+--------------+
3 rows in set (0.00 sec)
```

Q2-B Considering the same tables created, write the queries for the following:

1. Display the total of price from the product table.

2. Display the product having minimum price.

3. Display the number of orders placed in the month of February 2020.

4. Display the average of price of all the products.

5. Display the maximum payment paid on order ‘O01’

6. Display the payment truncated to two places from the payment table and differentiate between

Round () and truncate() function in MySQL.

7. Display the count of items ordered in the year 2019.

8. Display the no of orders, product wise from the table orders.

9. Display the data from order and product table with the matching product number.

10. Display orderno, orderdate, product no, descr, payment id, payment(qty*price) from three tables.

11. Display the product wise total qty ordered.

12. Display the total payment collected for each product ordered.

13. Display the total payment collected for only those products which were ordered in January 2019.

14. Display the total payment collected for only those products where total payment collected is more than 20000.

15. Add a column REMARKS in payment table.

16. Update the value of REMARKS with “paid” for all the payment ids.

17. insert one row in payment table where remarks should not be entered.

18. Display those payments which are not to be paid (having NULL).

19. Display those payments which are not null.

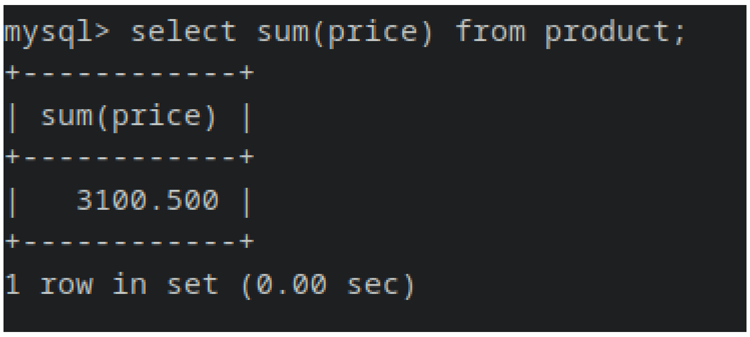
20.

a. Difference between delete and drop

b. Write the DEGREE and CARDINALITY of each table.

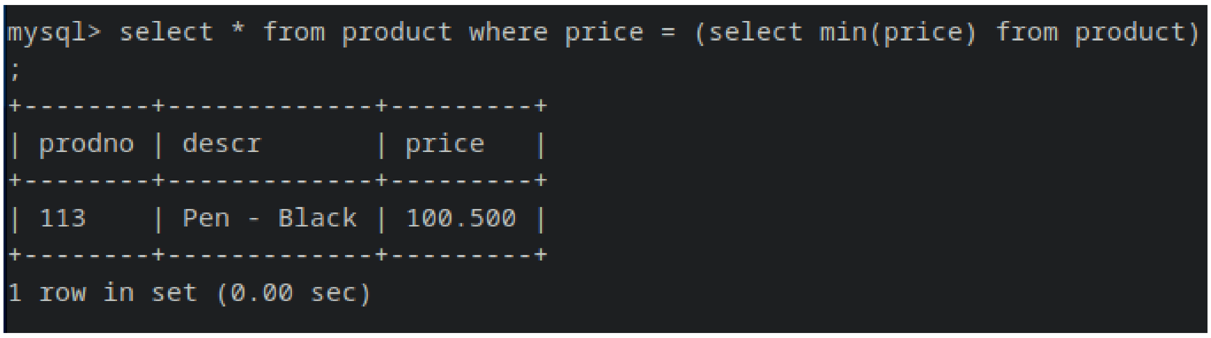
Answers:

Q1)



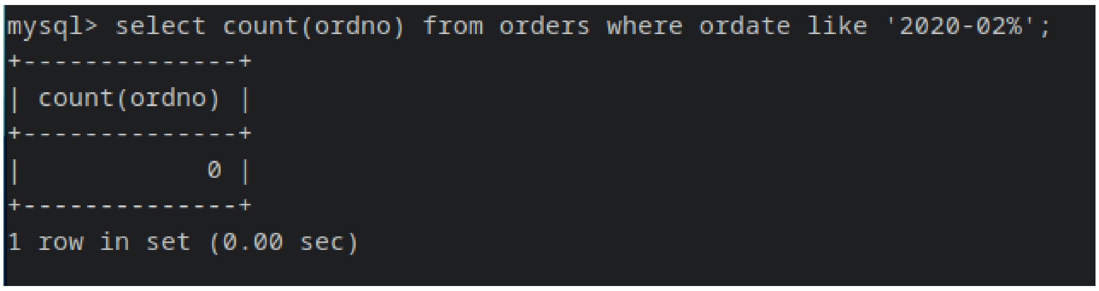
```
mysql> select sum(price) from product;
+------------+
| sum(price) |
+------------+
|   3100.500 |
+------------+
1 row in set (0.00 sec)
```

Q2)



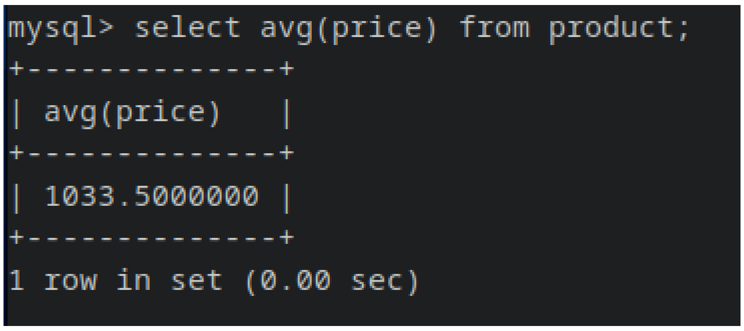
```
mysql> select * from product where price = (select min(price) from product)
;
+--------+-------------+---------+
| prodno | descr       | price   |
+--------+-------------+---------+
| 113    | Pen - Black | 100.500 |
+--------+-------------+---------+
1 row in set (0.00 sec)
```

Q3)



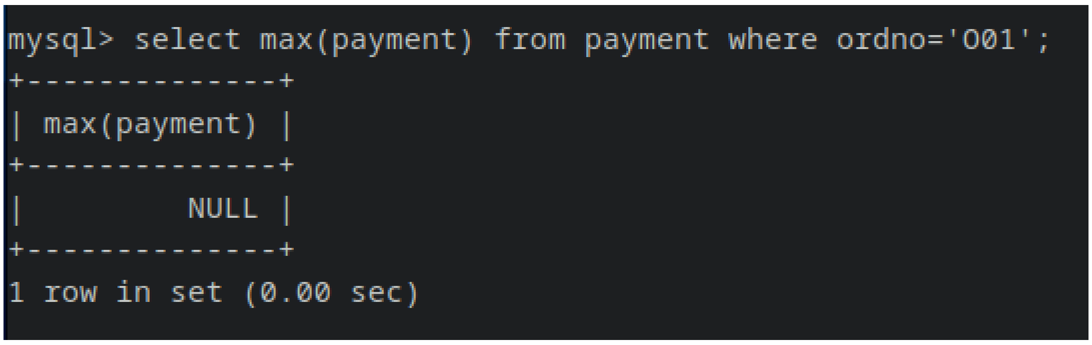
```
mysql> select count(ordno) from orders where ordate like '2020-02%';
+--------------+
| count(ordno) |
+--------------+
|            0 |
+--------------+
1 row in set (0.00 sec)
```

Q4)



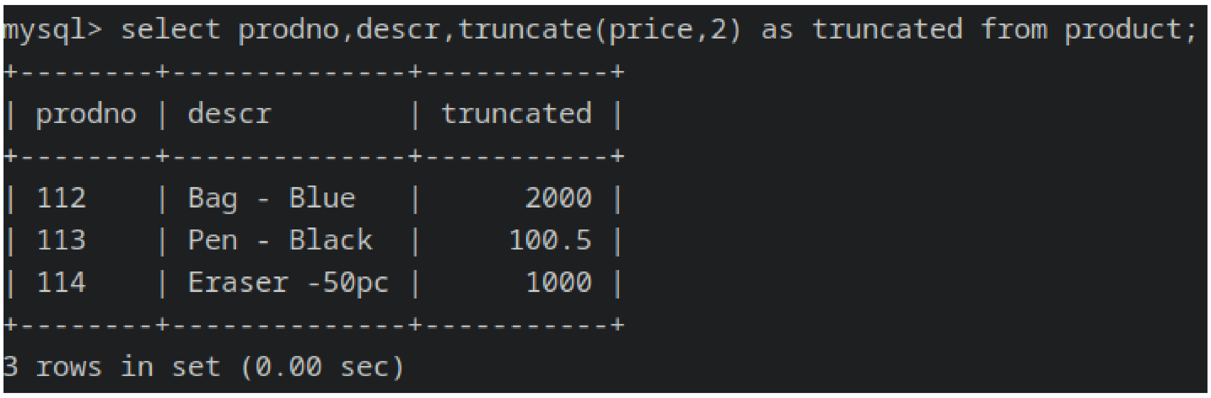
```
mysql> select avg(price) from product;
+--------------+
| avg(price)   |
+--------------+
| 1033.5000000 |
+--------------+
1 row in set (0.00 sec)
```

Q5)



```
mysql> select max(payment) from payment where ordno='O01';
+--------------+
| max(payment) |
+--------------+
|         NULL |
+--------------+
1 row in set (0.00 sec)
```

Q6)



```
mysql> select prodno,descr,truncate(price,2) as truncated from product;
+--------+-------------+-----------+
| prodno | descr       | truncated |
+--------+-------------+-----------+
| 112    | Bag - Blue  |      2000 |
| 113    | Pen - Black |     100.5 |
| 114    | Eraser -50pc|      1000 |
+--------+-------------+-----------+
3 rows in set (0.00 sec)
```

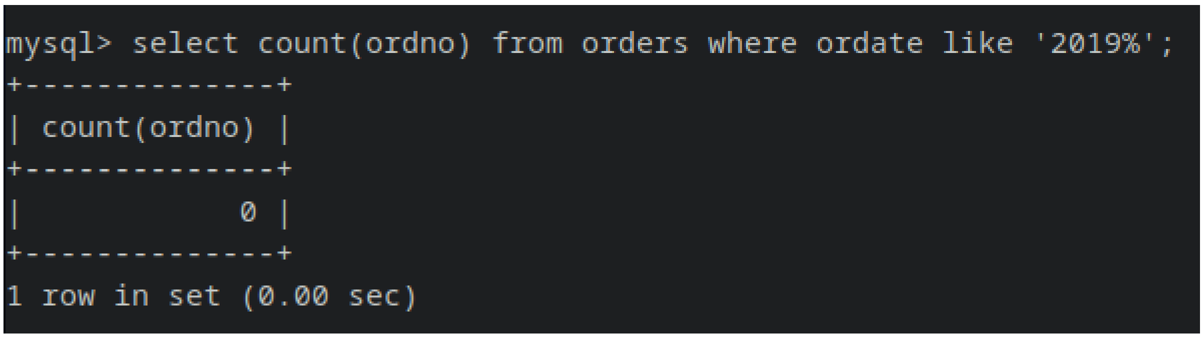
The truncate() function removes all the digits after the specified number and the round() function rounds off the value till the given digits.

For example:

truncate(9.999,2) will return 9.99

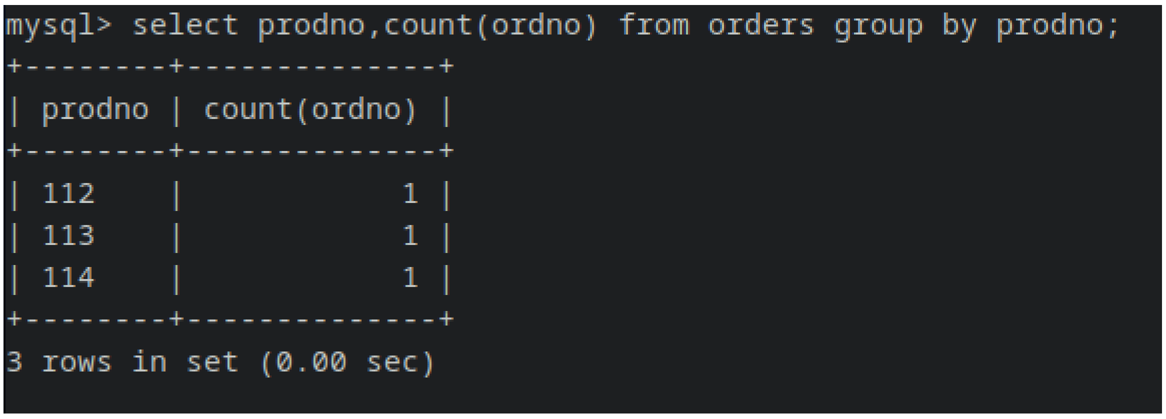
round(9.999,2) will return 10.00

Q7)



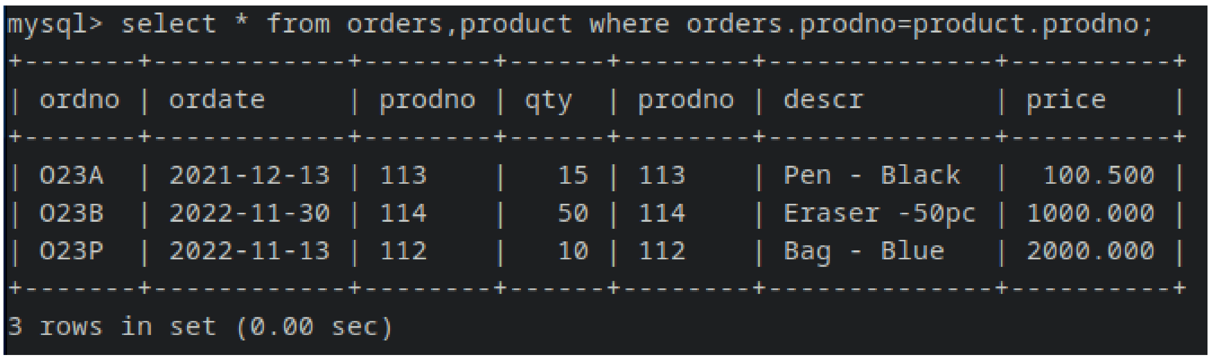
```
mysql> select count(ordno) from orders where ordate like '2019%';
+--------------+
| count(ordno) |
+--------------+
|            0 |
+--------------+
1 row in set (0.00 sec)
```

Q8)



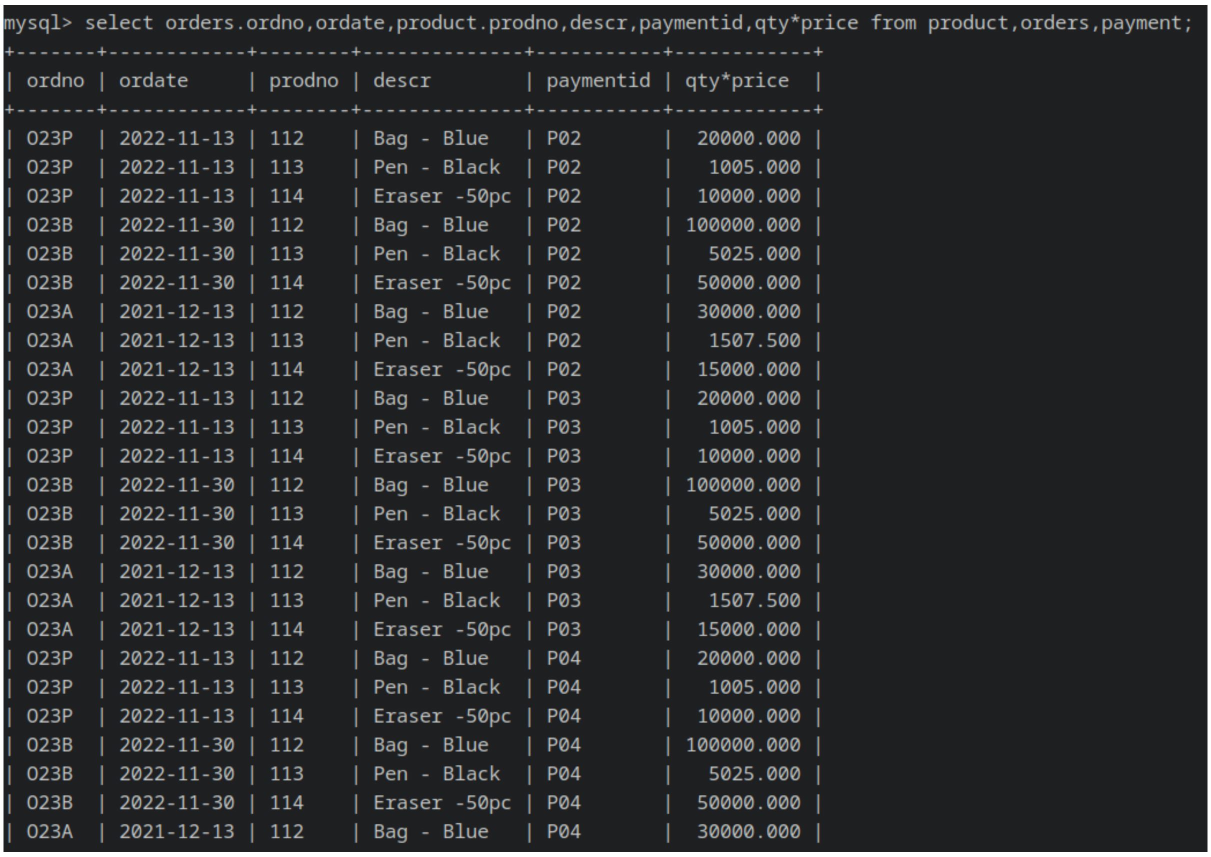
```
mysql> select prodno,count(ordno) from orders group by prodno;
+--------+--------------+
| prodno | count(ordno) |
+--------+--------------+
| 112    |            1 |
| 113    |            1 |
| 114    |            1 |
+--------+--------------+
3 rows in set (0.00 sec)
```

Q9)

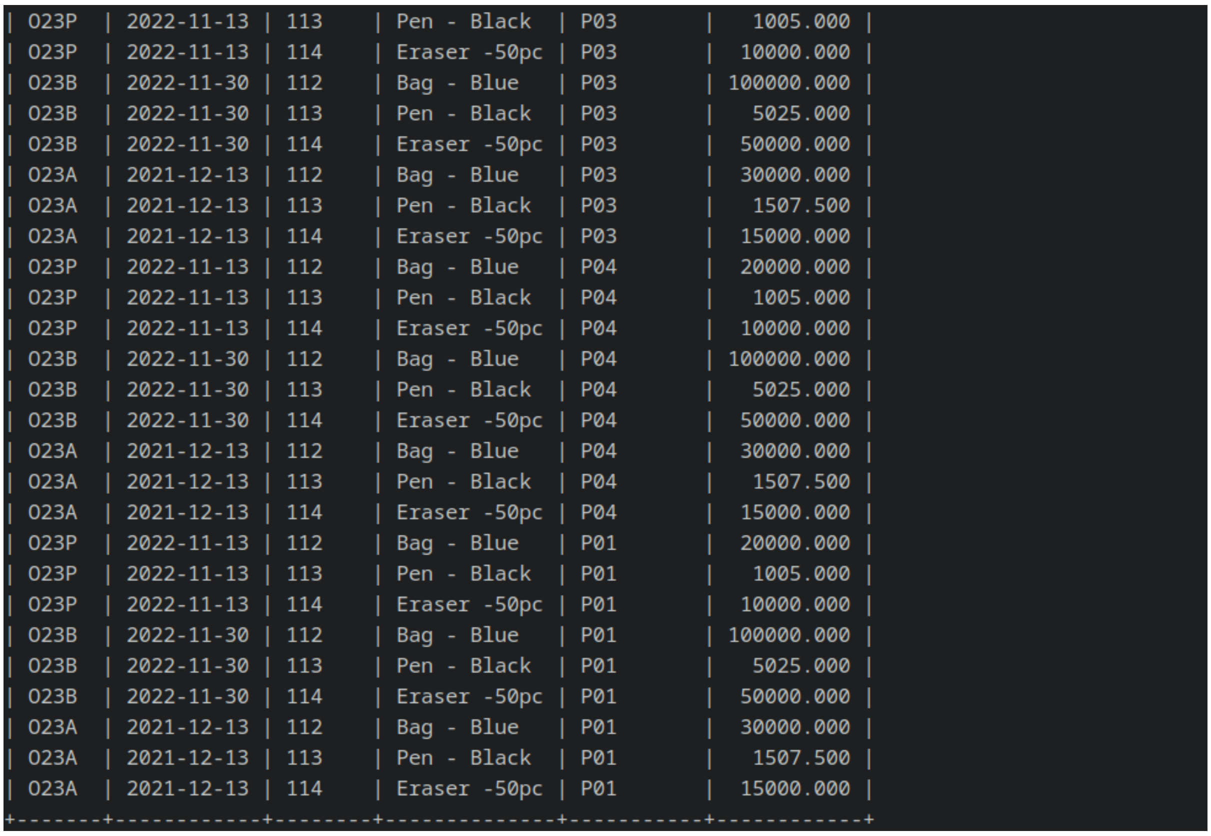


```
mysql> select * from orders,product where orders.prodno=product.prodno;
+-------+------------+--------+------+--------+--------------+----------+
| ordno | ordate     | prodno | qty  | prodno | descr        | price    |
+-------+------------+--------+------+--------+--------------+----------+
| O23A  | 2021-12-13 | 113    |   15 | 113    | Pen - Black  |  100.500 |
| O23B  | 2022-11-30 | 114    |   50 | 114    | Eraser -50pc | 1000.000 |
| O23P  | 2022-11-13 | 112    |   10 | 112    | Bag - Blue   | 2000.000 |
+-------+------------+--------+------+--------+--------------+----------+
3 rows in set (0.00 sec)
```

Q10)

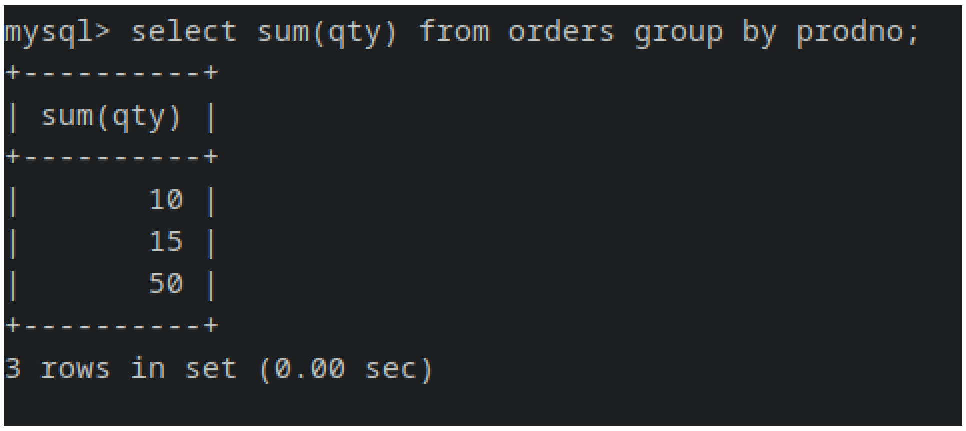


```
mysql> select orders.ordno,ordate,product.prodno,descr,paymentid,qty*price from product,orders,payment;
+-------+------------+--------+--------------+-----------+------------+
| ordno | ordate     | prodno | descr        | paymentid | qty*price  |
+-------+------------+--------+--------------+-----------+------------+
| O23P  | 2022-11-13 | 112    | Bag - Blue   | P02       |  20000.000 |
| O23P  | 2022-11-13 | 113    | Pen - Black  | P02       |   1005.000 |
| O23P  | 2022-11-13 | 114    | Eraser -50pc | P02       |  10000.000 |
| O23B  | 2022-11-30 | 112    | Bag - Blue   | P02       | 100000.000 |
| O23B  | 2022-11-30 | 113    | Pen - Black  | P02       |   5025.000 |
| O23B  | 2022-11-30 | 114    | Eraser -50pc | P02       |  50000.000 |
| O23A  | 2021-12-13 | 112    | Bag - Blue   | P02       |  30000.000 |
| O23A  | 2021-12-13 | 113    | Pen - Black  | P02       |   1507.500 |
| O23A  | 2021-12-13 | 114    | Eraser -50pc | P02       |  15000.000 |
| O23P  | 2022-11-13 | 112    | Bag - Blue   | P03       |  20000.000 |
| O23P  | 2022-11-13 | 113    | Pen - Black  | P03       |   1005.000 |
| O23P  | 2022-11-13 | 114    | Eraser -50pc | P03       |  10000.000 |
| O23B  | 2022-11-30 | 112    | Bag - Blue   | P03       | 100000.000 |
| O23B  | 2022-11-30 | 113    | Pen - Black  | P03       |   5025.000 |
| O23B  | 2022-11-30 | 114    | Eraser -50pc | P03       |  50000.000 |
| O23A  | 2021-12-13 | 112    | Bag - Blue   | P03       |  30000.000 |
| O23A  | 2021-12-13 | 113    | Pen - Black  | P03       |   1507.500 |
| O23A  | 2021-12-13 | 114    | Eraser -50pc | P03       |  15000.000 |
| O23P  | 2022-11-13 | 112    | Bag - Blue   | P04       |  20000.000 |
| O23P  | 2022-11-13 | 113    | Pen - Black  | P04       |   1005.000 |
| O23P  | 2022-11-13 | 114    | Eraser -50pc | P04       |  10000.000 |
| O23B  | 2022-11-30 | 112    | Bag - Blue   | P04       | 100000.000 |
| O23B  | 2022-11-30 | 113    | Pen - Black  | P04       |   5025.000 |
| O23B  | 2022-11-30 | 114    | Eraser -50pc | P04       |  50000.000 |
| O23A  | 2021-12-13 | 112    | Bag - Blue   | P04       |  30000.000 |
```



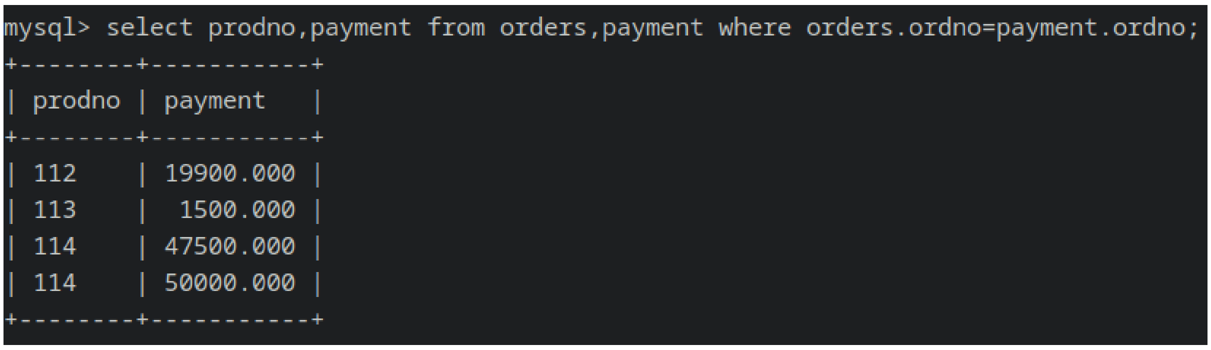
```
| O23P  | 2022-11-13 | 113    | Pen - Black   | P03      |   1005.000 |
| O23P  | 2022-11-13 | 114    | Eraser -50pc  | P03      |  10000.000 |
| O23B  | 2022-11-30 | 112    | Bag - Blue    | P03      | 100000.000 |
| O23B  | 2022-11-30 | 113    | Pen - Black   | P03      |   5025.000 |
| O23B  | 2022-11-30 | 114    | Eraser -50pc  | P03      |  50000.000 |
| O23A  | 2021-12-13 | 112    | Bag - Blue    | P03      |  30000.000 |
| O23A  | 2021-12-13 | 113    | Pen - Black   | P03      |   1507.500 |
| O23A  | 2021-12-13 | 114    | Eraser -50pc  | P03      |  15000.000 |
| O23P  | 2022-11-13 | 112    | Bag - Blue    | P04      |  20000.000 |
| O23P  | 2022-11-13 | 113    | Pen - Black   | P04      |   1005.000 |
| O23P  | 2022-11-13 | 114    | Eraser -50pc  | P04      |  10000.000 |
| O23B  | 2022-11-30 | 112    | Bag - Blue    | P04      | 100000.000 |
| O23B  | 2022-11-30 | 113    | Pen - Black   | P04      |   5025.000 |
| O23B  | 2022-11-30 | 114    | Eraser -50pc  | P04      |  50000.000 |
| O23A  | 2021-12-13 | 112    | Bag - Blue    | P04      |  30000.000 |
| O23A  | 2021-12-13 | 113    | Pen - Black   | P04      |   1507.500 |
| O23A  | 2021-12-13 | 114    | Eraser -50pc  | P04      |  15000.000 |
| O23P  | 2022-11-13 | 112    | Bag - Blue    | P01      |  20000.000 |
| O23P  | 2022-11-13 | 113    | Pen - Black   | P01      |   1005.000 |
| O23P  | 2022-11-13 | 114    | Eraser -50pc  | P01      |  10000.000 |
| O23B  | 2022-11-30 | 112    | Bag - Blue    | P01      | 100000.000 |
| O23B  | 2022-11-30 | 113    | Pen - Black   | P01      |   5025.000 |
| O23B  | 2022-11-30 | 114    | Eraser -50pc  | P01      |  50000.000 |
| O23A  | 2021-12-13 | 112    | Bag - Blue    | P01      |  30000.000 |
| O23A  | 2021-12-13 | 113    | Pen - Black   | P01      |   1507.500 |
| O23A  | 2021-12-13 | 114    | Eraser -50pc  | P01      |  15000.000 |
+-------+------------+--------+--------------+----------+------------+
```

Q11)



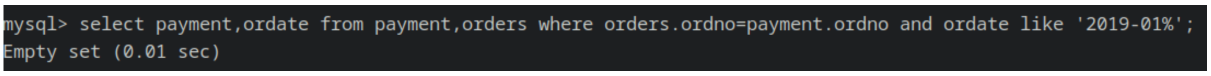
```
mysql> select sum(qty) from orders group by prodno;
+----------+
| sum(qty) |
+----------+
|       10 |
|       15 |
|       50 |
+----------+
3 rows in set (0.00 sec)
```

Q12)



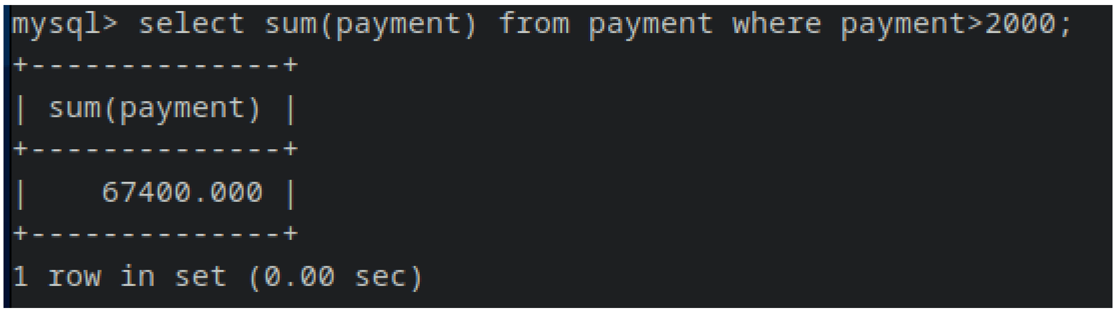
```
mysql> select prodno,payment from orders,payment where orders.ordno=payment.ordno;
+--------+-----------+
| prodno | payment   |
+--------+-----------+
| 112    | 19900.000 |
| 113    |  1500.000 |
| 114    | 47500.000 |
| 114    | 50000.000 |
+--------+-----------+
```

Q13)



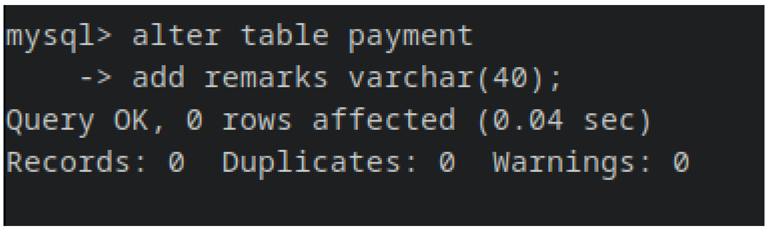
```
mysql> select payment,ordate from payment,orders where orders.ordno=payment.ordno and ordate like '2019-01%';
Empty set (0.01 sec)
```

Q14)



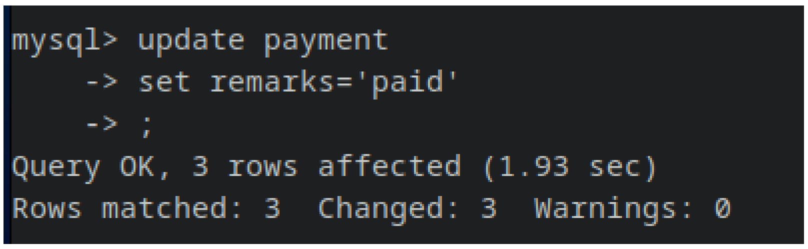
```
mysql> select sum(payment) from payment where payment>2000;
+--------------+
| sum(payment) |
+--------------+
|    67400.000 |
+--------------+
1 row in set (0.00 sec)
```

Q15)

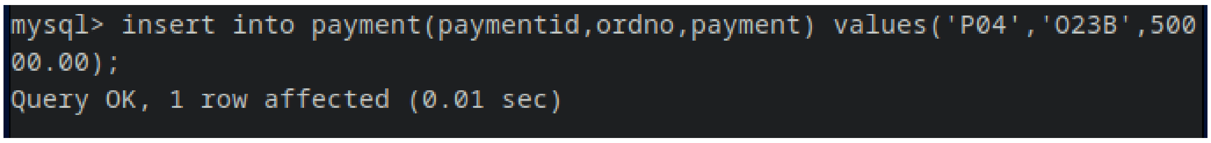


```
mysql> alter table payment
    -> add remarks varchar(40);
Query OK, 0 rows affected (0.04 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

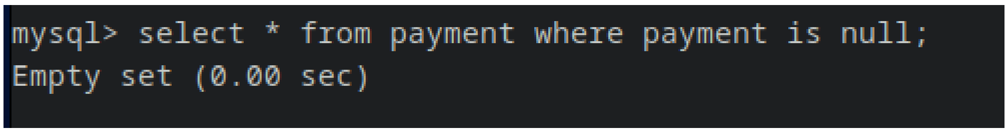
Q16)


```
mysql> update payment
    -> set remarks='paid'
    -> ;
Query OK, 3 rows affected (1.93 sec)
Rows matched: 3  Changed: 3  Warnings: 0
```

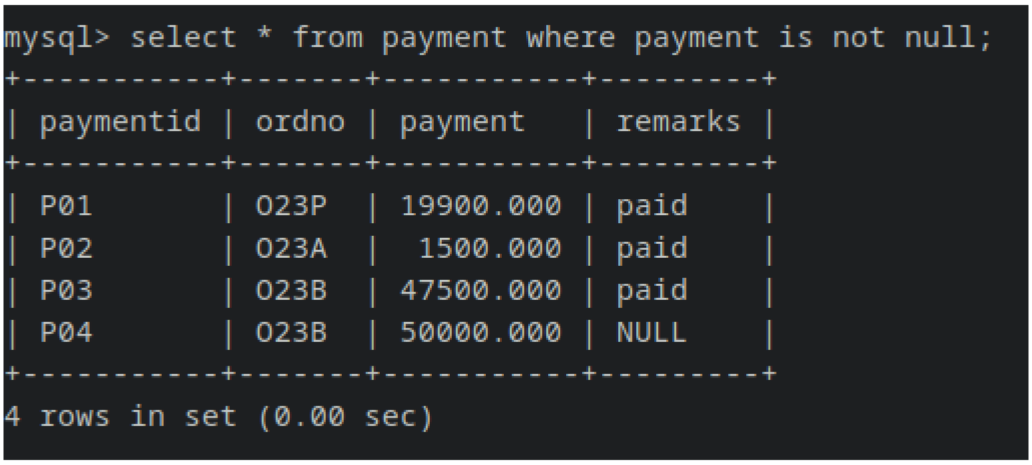
Q17)


```
mysql> insert into payment(paymentid,ordno,payment) values('P04','O23B',500
00.00);
Query OK, 1 row affected (0.01 sec)
```

Q18)


```
mysql> select * from payment where payment is null;
Empty set (0.00 sec)
```

Q19)


```
mysql> select * from payment where payment is not null;
+-----------+-------+-----------+---------+
| paymentid | ordno | payment   | remarks |
+-----------+-------+-----------+---------+
| P01       | O23P  | 19900.000 | paid    |
| P02       | O23A  |  1500.000 | paid    |
| P03       | O23B  | 47500.000 | paid    |
| P04       | O23B  | 50000.000 | NULL    |
+-----------+-------+-----------+---------+
4 rows in set (0.00 sec)
```

Q20)

A) Delete command is used to alter the rows (the data present). It is a DML operation. Drop command is used to delete tables,columns or even databases(the structure). It is a DDL operation.
B) Payment Table: Cardinality = 4, Degree = 4

   Orders Table:Cardinality = 3, Degree = 4

   Product Table: Cardinality = 3, Degree = 3

SECTION-III : CONNECTIVITY OF PYTHON WITH SQL DATABASE

Q3-A Using the above tables and database , perform the following functionalities while writing a PYTHON program:

1. Create following two tables:a. JOB (jobcode text primary key, area text not null, app_date ,salary decimal should be positive,,retd_date ,dept)

b. PERSONAL(empno text primary key, name text not null, dobirth ,nativeplace text, hobby text,jobcode text and

foreign key)

c. Insert three rows in both the tables.

d. Display the entire data in the order of jobcode from JOB and empno from PERSONAL.

e. Enter the value of jobcode to update the area with “Saket New Delhi “.

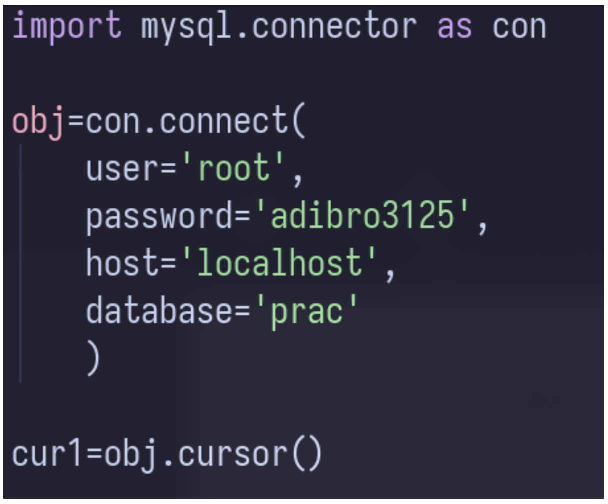
f. Enter the value of empno to delete the record from the table.

g. Fetch one by one record from the result set and display on the screen. Also display the number of rows retrieved

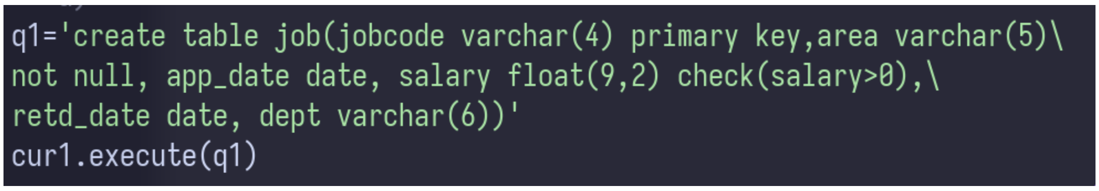
From the result set.

h. Fetch all the rows from the result set and display .

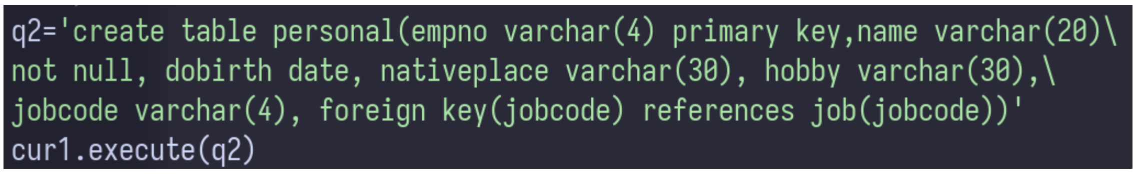
Answers:



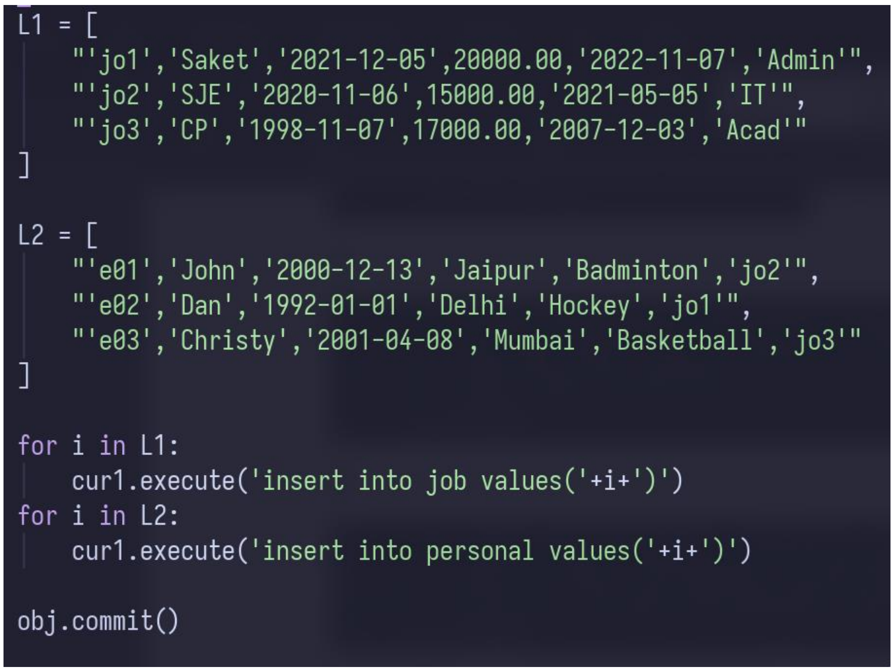
```python
import mysql.connector as con

obj=con.connect(
    user='root',
    password='adibro3125',
    host='localhost',
    database='prac'
    )

cur1=obj.cursor()
```



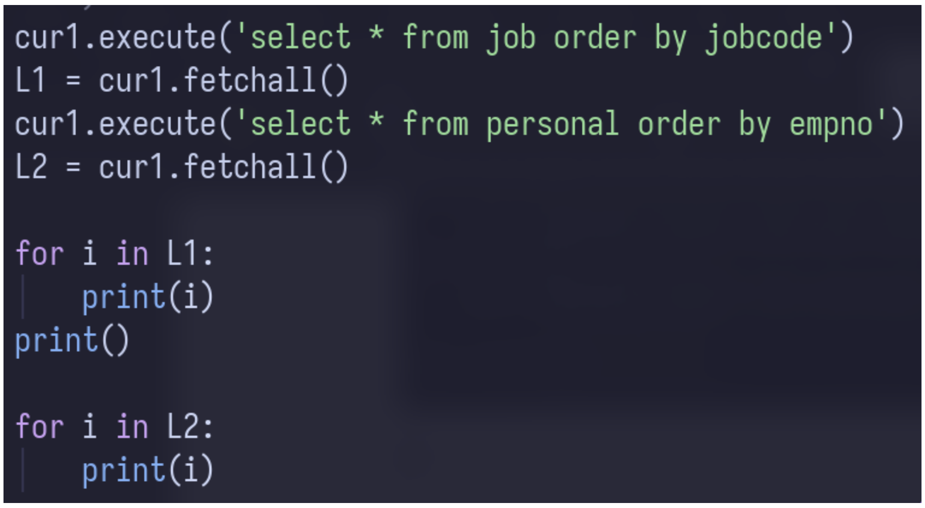
```python
q1='create table job(jobcode varchar(4) primary key,area varchar(5)\
not null, app_date date, salary float(9,2) check(salary>0),\
retd_date date, dept varchar(6))'
cur1.execute(q1)
```



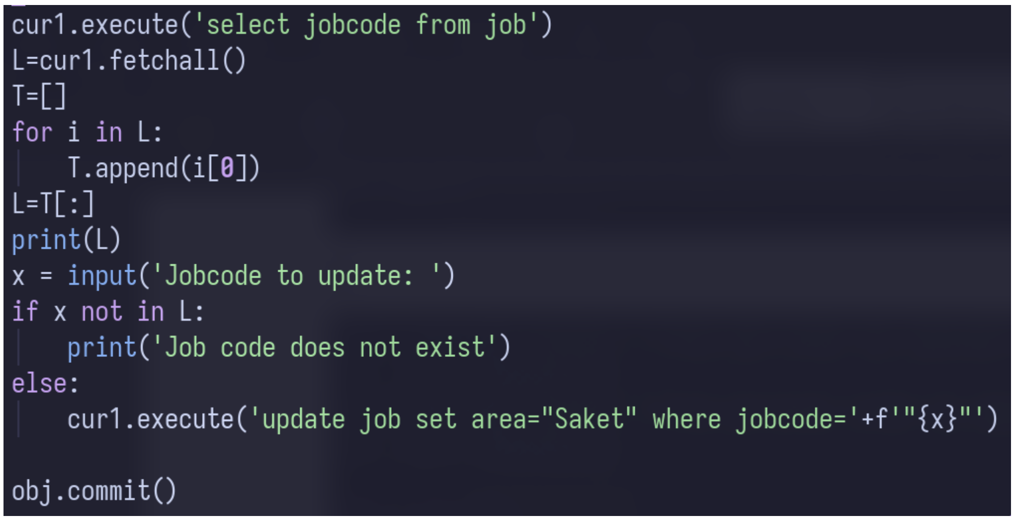
```python
q2='create table personal(empno varchar(4) primary key,name varchar(20)\
not null, dobirth date, nativeplace varchar(30), hobby varchar(30),\
jobcode varchar(4), foreign key(jobcode) references job(jobcode))'
cur1.execute(q2)
```



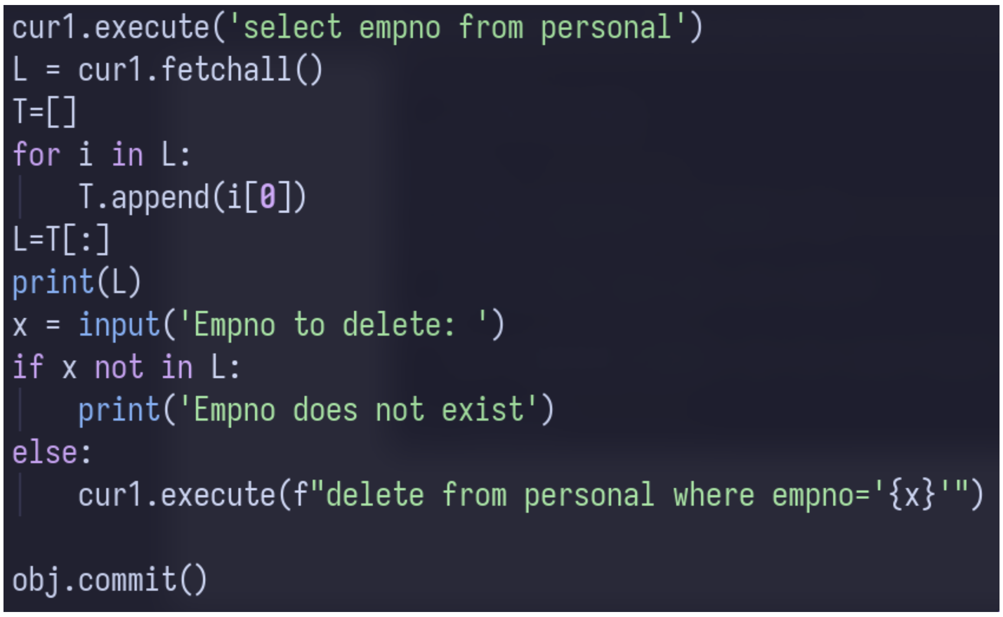
```python
L1 = [
    "'jo1','Saket','2021-12-05',20000.00,'2022-11-07','Admin'",
    "'jo2','SJE','2020-11-06',15000.00,'2021-05-05','IT'",
    "'jo3','CP','1998-11-07',17000.00,'2007-12-03','Acad'"
]

L2 = [
    "'e01','John','2000-12-13','Jaipur','Badminton','jo2'",
    "'e02','Dan','1992-01-01','Delhi','Hockey','jo1'",
    "'e03','Christy','2001-04-08','Mumbai','Basketball','jo3'"
]

for i in L1:
    cur1.execute('insert into job values('+i+')')
for i in L2:
    cur1.execute('insert into personal values('+i+')')

obj.commit()
```



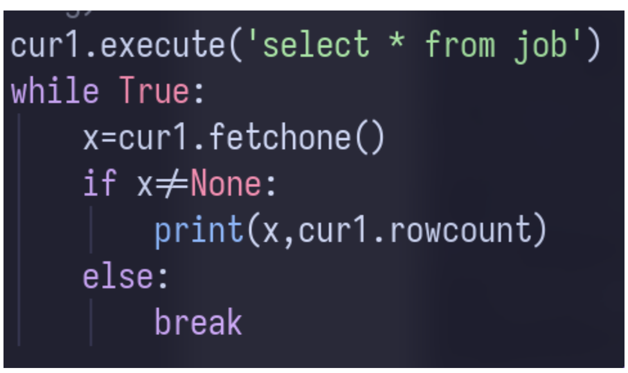
```
cur1.execute('select * from job order by jobcode')
L1 = cur1.fetchall()
cur1.execute('select * from personal order by empno')
L2 = cur1.fetchall()

for i in L1:
    print(i)
print()

for i in L2:
    print(i)
```



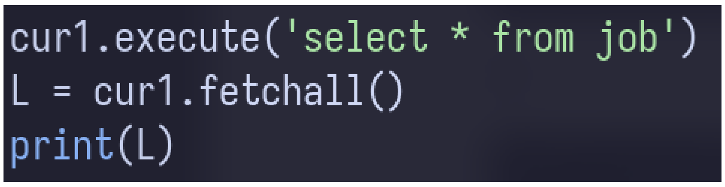
```
cur1.execute('select jobcode from job')
L=cur1.fetchall()
T=[]
for i in L:
    T.append(i[0])
L=T[:]
print(L)
x = input('Jobcode to update: ')
if x not in L:
    print('Job code does not exist')
else:
    cur1.execute('update job set area="Saket" where jobcode='+f'"{x}"')

obj.commit()
```



```
cur1.execute('select empno from personal')
L = cur1.fetchall()
T=[]
for i in L:
    T.append(i[0])
L=T[:]
print(L)
x = input('Empno to delete: ')
if x not in L:
    print('Empno does not exist')
else:
    cur1.execute(f"delete from personal where empno='{x}'")

obj.commit()
```



```python
cur1.execute('select * from job')
while True:
    x=cur1.fetchone()
    if x≠None:
        print(x,cur1.rowcount)
    else:
        break
```



```python
cur1.execute('select * from job')
L = cur1.fetchall()
print(L)
```