def count\_frequencies(arr):

freq={}

for num in arr:

freq[num]=freq.get(num,0)+1

return freq

arr=[1,2,3,4,4,3,2,1,1,1,2,2,3,3,4,4]

frequencies=count\_frequencies(arr)

for i,count in frequencies.items():

print(i,count)

[30-05-2025 11:53]

class Solution:

# Function to count the frequency of all elements from 1 to N in the array.

def frequencyCount(self, arr):

freq={}

ans=[]

n=len(arr)

for i in arr:

freq[i]=freq.get(i,0)+1

for i in range(1,n+1):

ans.append(freq.get(i,0))

return ans

[30-05-2025 12:56]

n=int(input())

for i in range(n):

for j in range(n-i-1):

print(" ",end=" ")

for j in range(i+1):

print("\*",end=" ")

print()

[30-05-2025 13:37]

n=int(input())

for i in range(n):

for j in range(n-i-1):

print(" ",end=" ")

for j in range(i+1):

print("\*",end=" ")

print()

n=int(input())

for i in range(n):

for j in range(i):

print(" ",end=" ")

for j in range(n-i):

print("\*",end=" ")

print()

n=6

for x in range(1,n+1):

for y in range(0,x):

print(str((x+y)%2)+"",end="")

print()

n=5

for i in range(1,n+1):

print(" "\*(n-i),end="")

x=1

for j in range(1,i+1):

print(x,end="")

x=x\*(i-j)//j

print()

for i in range(1,5):

for j in range(4-i):

print(end="")

for j in range(1,i\*2):

print("\*",end=" ")

print()

for i in range(3,0,-1):

for j in range(4-i):

print(end=" ")

for j in range(1,i\*2):

print("\*",end="")

print()

def floyd\_triangle(n):

num=1

for i in range(1,n+1):

for k in range(n-i,0,-1):

print(end=" ")

for j in range(1,i+1):

print("%2"%num,end=" ")

num=num+1

print("")

floyd\_triangle(6)

[30-05-2025 14:34]

class Solution:

#Function to rotate an array by d elements in counter-clockwise direction.

def rotateArr(self, arr, d):

n=len(arr)

temp=[0]\*n

d=d%n

for i in range(len(arr)):

temp[(n-d+i)%n]=arr[i]

for i in range(len(arr)):

arr[i]=temp[i]

return arr

[30-05-2025 14:48]

def reverse(arr, start, end):

while start < end:

arr[start], arr[end] = arr[end], arr[start]

start += 1

end -= 1

def rotate\_array(arr, k):

n = len(arr)

k = k % n

reverse(arr, 0, n - 1)

reverse(arr, 0, k - 1)

reverse(arr, k, n - 1)

return arr

DAY-12

[30-05-2025 10:04]

def count\_frequencies(arr):

freq={}

for num in arr:

freq[num]=freq.get(num,0)+1

return freq

arr=[1,2,3,4,4,3,2,1,1,1,2,2,3,3,4,4]

frequencies=count\_frequencies(arr)

for i,count in frequencies.items():

print(i,count)

[30-05-2025 11:53]

class Solution:

# Function to count the frequency of all elements from 1 to N in the array.

def frequencyCount(self, arr):

freq={}

ans=[]

n=len(arr)

for i in arr:

freq[i]=freq.get(i,0)+1

for i in range(1,n+1):

ans.append(freq.get(i,0))

return ans

[30-05-2025 12:56]

n=int(input())

for i in range(n):

for j in range(n-i-1):

print(" ",end=" ")

for j in range(i+1):

print("\*",end=" ")

print()

[30-05-2025 13:37]

n=int(input())

for i in range(n):

for j in range(n-i-1):

print(" ",end=" ")

for j in range(i+1):

print("\*",end=" ")

print()

n=int(input())

for i in range(n):

for j in range(i):

print(" ",end=" ")

for j in range(n-i):

print("\*",end=" ")

print()

n=6

for x in range(1,n+1):

for y in range(0,x):

print(str((x+y)%2)+"",end="")

print()

n=5

for i in range(1,n+1):

print(" "\*(n-i),end="")

x=1

for j in range(1,i+1):

print(x,end="")

x=x\*(i-j)//j

print()

for i in range(1,5):

for j in range(4-i):

print(end="")

for j in range(1,i\*2):

print("\*",end=" ")

print()

for i in range(3,0,-1):

for j in range(4-i):

print(end=" ")

for j in range(1,i\*2):

print("\*",end="")

print()

def floyd\_triangle(n):

num=1

for i in range(1,n+1):

for k in range(n-i,0,-1):

print(end=" ")

for j in range(1,i+1):

print("%2"%num,end=" ")

num=num+1

print("")

floyd\_triangle(6)

[30-05-2025 14:34]

class Solution:

#Function to rotate an array by d elements in counter-clockwise direction.

def rotateArr(self, arr, d):

n=len(arr)

temp=[0]\*n

d=d%n

for i in range(len(arr)):

temp[(n-d+i)%n]=arr[i]

for i in range(len(arr)):

arr[i]=temp[i]

return arr

[30-05-2025 14:48]

def reverse(arr, start, end):

while start < end:

arr[start], arr[end] = arr[end], arr[start]

start += 1

end -= 1

def rotate\_array(arr, k):

n = len(arr)

k = k % n

reverse(arr, 0, n - 1)

reverse(arr, 0, k - 1)

reverse(arr, k, n - 1)

return arr

DAY-13

[30-05-2025 10:04]

def count\_frequencies(arr):

freq={}

for num in arr:

freq[num]=freq.get(num,0)+1

return freq

arr=[1,2,3,4,4,3,2,1,1,1,2,2,3,3,4,4]

frequencies=count\_frequencies(arr)

for i,count in frequencies.items():

print(i,count)

[30-05-2025 11:53]

class Solution:

# Function to count the frequency of all elements from 1 to N in the array.

def frequencyCount(self, arr):

freq={}

ans=[]

n=len(arr)

for i in arr:

freq[i]=freq.get(i,0)+1

for i in range(1,n+1):

ans.append(freq.get(i,0))

return ans

[30-05-2025 12:56]

n=int(input())

for i in range(n):

for j in range(n-i-1):

print(" ",end=" ")

for j in range(i+1):

print("\*",end=" ")

print()

[30-05-2025 13:37]

n=int(input())

for i in range(n):

for j in range(n-i-1):

print(" ",end=" ")

for j in range(i+1):

print("\*",end=" ")

print()

n=int(input())

for i in range(n):

for j in range(i):

print(" ",end=" ")

for j in range(n-i):

print("\*",end=" ")

print()

n=6

for x in range(1,n+1):

for y in range(0,x):

print(str((x+y)%2)+"",end="")

print()

n=5

for i in range(1,n+1):

print(" "\*(n-i),end="")

x=1

for j in range(1,i+1):

print(x,end="")

x=x\*(i-j)//j

print()

for i in range(1,5):

for j in range(4-i):

print(end="")

for j in range(1,i\*2):

print("\*",end=" ")

print()

for i in range(3,0,-1):

for j in range(4-i):

print(end=" ")

for j in range(1,i\*2):

print("\*",end="")

print()

def floyd\_triangle(n):

num=1

for i in range(1,n+1):

for k in range(n-i,0,-1):

print(end=" ")

for j in range(1,i+1):

print("%2"%num,end=" ")

num=num+1

print("")

floyd\_triangle(6)

[30-05-2025 14:34]

class Solution:

#Function to rotate an array by d elements in counter-clockwise direction.

def rotateArr(self, arr, d):

n=len(arr)

temp=[0]\*n

d=d%n

for i in range(len(arr)):

temp[(n-d+i)%n]=arr[i]

for i in range(len(arr)):

arr[i]=temp[i]

return arr

[30-05-2025 14:48]

def reverse(arr, start, end):

while start < end:

arr[start], arr[end] = arr[end], arr[start]

start += 1

end -= 1

def rotate\_array(arr, k):

n = len(arr)

k = k % n

reverse(arr, 0, n - 1)

reverse(arr, 0, k - 1)

reverse(arr, k, n - 1)

return arr

DAY-14

[02-06-2025 15:36]

create database student\_management;

use student\_management;

create table students(

roll\_no varchar(20) primary key,

name varchar(100),

class varchar(20),

section varchar(5),

password varchar(100),

email varchar(100)

);

create table marks(

roll\_no varchar(20),

subject varchar(50),

marks INT,

primary key (roll\_no,subject),

foreign key(roll\_no) references

students(roll\_no)

);

create table timetable(

class varchar(20),

section varchar(5),

day varchar(20),

period1 varchar(50),

period2 varchar(50),

period3 varchar(50),

period4 varchar(50),

primary key(class,section,day)

);

[02-06-2025 15:39]

from db import connection

conn = connection()

if conn:

print("Connection established successfully")

else:

print("Connection failed")

def insert\_data():

roll\_no = int(input("Enter roll\_no: "))

name = input("Enter name: ")

branch = input("Enter branch: ")

cursor = conn.cursor()

query = "INSERT INTO Student(roll\_no, name, branch) VALUES (%s, %s, %s)"

values = (roll\_no, name, branch)

cursor.execute(query, values)

conn.commit()

print("Data inserted successfully")

def fetch\_data():

cursor=conn.cursor()

query="select\* from Student"

cursor.execute(query)

results=cursor.fetchall()

for row in results:

print(row)

fetch\_data()

def update\_data():

roll\_no = int(input("Enter roll\_no to update: "))

name = input("Enter name: ")

branch = input("Enter branch: ")

cursor=conn.cursor()

query="update Student set name=%s,branch=%s where roll\_no=%s"

values=(name,branch,roll\_no)

cursor.execute(query,values)

conn.commit()

print("data updated successfully")

print("1.insert data")

print("2.fetch data")

print("3.update data")

print("Enter 4 to exit")

while True:

choice=int(input("enter your choice(1-4):"))

if choice==1:

insert\_data()

elif choice==2:

fetch\_data()

elif choice==3:

updata\_data()

elif choice==4:

exiting\_data()

break

[02-06-2025 15:39]

import mysql.connector

def connection():

conn=mysql.connector.connect(

host='localhost',

user='root',

password="Mouni@123",

database="Practice\_crt"

)

return connection

if(connection()):

print("connection established")

else:

print("connection failed")

[02-06-2025 16:30]

ADMIN FEATURES:

1.ADD STUDENT

2.DELETE STUDENT

3.UPDATE STUDENT

4.TIME TABLE

5.MARKS

'''

from db import connect

def admin():

conn=connect()

cursor=conn.cursor()

print("""\nAdmin Menu:

1.Add student

2.update student details

3.reset student password

4.update Marks

5.view All students

6.update Timetable

7.logout""")

choice=int(input("enter your choice:"))

if choice==1:

add\_student()

elif choice==2:

update\_student()

elif choice==3:

reset\_student()

elif choice==4:

update Marks()

elif choice==5:

view all students()

elif choice==6:

update\_timetable()

elif choice==7

logout()

else:

print("Invalid choice. Please try again.")

def add\_student():

conn=connect()

cursor=conn.cursor()

roll\_no=input("enter roll\_no:")

name=input("enter name:")

class\_name=input=("enter class:")

section=input("enter section:")

password="password@132"

email=input("enter email:")

query="insert into students(roll\_no,name,class\_name,section,password,email) values(%s,%s,%s,%s,%s,%s)

values=(roll\_no,name,class\_name,section,password,email)

cursor.execute(query,values)

conn.commit()

print("student added successfully")

def update\_student():

pass

def reset\_student\_password():

pass

def update\_marks():

pass

def view\_all\_students():

pass

def update\_timetable():

pass

def logout():

print("logging out..")

return

# EXAMPLE USAGE

if\_name=="main\_":

admin()

from db import connection

conn=connection()

if conn:

print("connection established successfully")

else:

print("connection failed")

def insert\_data():

roll\_no=int(input("enter roll number: "))

name=input("enter name: ")

branch=input("enter branch: ")

cursor=conn.cursor()

query="insert into student(roll\_no,name,branch) values(%s,%s,%s)"

values=(roll\_no,name,branch)

cursor.execute(query,values)

conn.commit()

print("data inserted successfully")

def fetch\_data():

cursor=conn.cursor()

query="select \* from Student"

cursor.execute(query)

results=cursor.fetchall()

for row in results:

print(row)

def update\_data():

roll\_no=int(input("enter roll number: "))

name=input("enter name: ")

branch=input("enter branch: ")

cursor=conn.cursor()

query="update student set name=%s,branch=%s where roll\_no=%s"

values=(name,branch,roll\_no)

cursor.execute(query,values)

conn.commit()

print("data updated successfully")

# while loop to perform operation

print("1.insert data")

print("2.fetch data")

print("3.update data")

print("enter 4 to exit")

while True:

choice=int(input("enter your choice (1-4): "))

if choice==1:

insert\_data()

elif choice==2:

fetch\_data()

elif choice==3:

update\_data()

elif choice==4:

print("exiting")

break

from db import connection

conn=connection()

if conn:

print("connection established successfully")

else:

print("connection failed")

def insert\_data():

roll\_no=int(input("enter roll number: "))

name=input("enter name: ")

branch=input("enter branch: ")

cursor=conn.cursor()

query="insert into student(roll\_no,name,branch) values(%s,%s,%s)"

values=(roll\_no,name,branch)

cursor.execute(query,values)

conn.commit()

print("data inserted successfully")

def fetch\_data():

cursor=conn.cursor()

query="select \* from Student"

cursor.execute(query)

results=cursor.fetchall()

for row in results:

print(row)

def update\_data():

roll\_no=int(input("enter roll number: "))

name=input("enter name: ")

branch=input("enter branch: ")

cursor=conn.cursor()

query="update student set name=%s,branch=%s where roll\_no=%s"

values=(name,branch,roll\_no)

cursor.execute(query,values)

conn.commit()

print("data updated successfully")

# while loop to perform operation

print("1.insert data")

print("2.fetch data")

print("3.update data")

print("enter 4 to exit")

while True:

choice=int(input("enter your choice (1-4): "))

if choice==1:

insert\_data()

elif choice==2:

fetch\_data()

elif choice==3:

update\_data()

elif choice==4:

print("exiting")

break

DAY 15

'''

admin features:

1.add student

2.delete student

3.update student

4.time table

5.marks

'''

from db import connection

def admin():

conn=connection()

cursor=conn.cursor()

print("""\nadmin menu:

1.add student

2.update student details

3.reset student password

4.update marks

5.view all students

6.update timetable

7.logout

8.add marks

9.view marks""")

ch=int(input("Enter your choice: "))

if ch==1:

add\_student()

elif ch==2:

update\_student()

elif ch==3:

reset\_student\_password()

elif ch==4:

update\_marks()

elif ch==5:

view\_all\_students()

elif ch==6:

update\_timetable()

elif ch==7:

logout()

elif ch==8:

add\_marks()

elif ch==9:

view\_marks()

else:

print("invalid choice.please try again.")

def add\_student():

conn=connection()

cursor=conn.cursor()

roll\_no=input("Enter roll no: ")

name=input("Enter name: ")

class\_name=input("Enter class: ")

section=input("Enter section: ")

password="student123"

email=input("Enter email: ")

query="insert into students(roll\_no,name,class,section,password,email) values(%s,%s,%s,%s,%s,%s)"

values=(roll\_no,name,class\_name,section,password,email)

cursor.execute(query,values)

conn.commit()

print("student added successfully.")

def update\_student():

conn=connection()

cursor=conn.cursor()

roll\_no=input("Enter roll no of student to update: ")

name=input("Enter new name: ")

class\_name=input("Enter new class: ")

section=input("Enter new section: ")

email=input("Enter new email: ")

query="update students set name=%s,class=%s,section=%s,email=%s where roll\_no=%s"

values=(name,class\_name,section,email,roll\_no)

cursor.execute(query,values)

conn.commit()

print("student updated successfully")

def reset\_student\_password():

pass

def update\_marks():

conn=connection()

cursor=conn.cursor()

roll\_no=input("enter roll no of student to update marks: ")

subject=input("enter subject:")

marks=input("enter marks: ")

query="update marks set marks=%s where roll\_no=%s and subject=%s"

values=(marks,roll\_no,subject)

cursor.execute(query,values)

conn.commit()

print("marks updated successfully.")

def add\_marks():

conn=connection()

cursor=conn.cursor()

roll\_no=input("enter roll no of student to added marks: ")

subject=input("enter subject:")

marks=input("enter marks: ")

query="insert into marks(roll\_no,subject,marks) values(%s,%s,%s)"

values=(roll\_no,subject,marks)

cursor.execute(query,values)

conn.commit()

print("marks added successfully.")

def view\_marks():

conn=connection()

cursor=conn.cursor()

query="select \* from marks"

cursor.execute(query)

results=cursor.fetchall()

for row in results:

print(row)

def view\_all\_students():

conn=connection()

cursor=conn.cursor()

query="select \* from students"

cursor.execute(query)

results=cursor.fetchall()

for row in results:

print(row)

def update\_timetable():

pass

def logout():

print("logging out..")

return

if name =="main":

admin()

from db import connect

def student\_menu(roll\_no):

while True:

print("""\nStudent Menu:

1.View Details

2.View Marks

3.View Timetable

4.Logout""")

choice=input("Enter choice: ")

if choice=='1':

view\_details(roll\_no)

elif choice=='2':

view\_marks(roll\_no)

elif choice=='3':

view\_timetable(roll\_no)

elif choice=='4':

change\_password(roll\_no)

print("Logging out...")

else:

print("Invalid choice.")

def view\_details(roll\_no):

con = get\_connection()

cur = con.cursor()

cur.execute("SELECT \* FROM students WHERE roll\_no=%s",(roll\_no,))

print("Student Details:")

print(cur.fetchone())

con.close()

view\_details(432)