STUDENT EXAMINATION PORTAL

**Submitted by**

Name of the Student: Asmit Sil

Enrollment Number: 12022002011054

Section: I

Class Roll Number: 54  
Stream: Electrical Engineering

Subject: Programming for Problem Solving with Python

Subject Code: IVC101

Department: Basic Science and Humanities

Under the supervision of

**Dr. Swarnendu Ghosh**

**Prof. Sumana Sinha**

**Academic Year: 2022-26**

*PROJECT REPORT SUBMITTED IN* *PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE FIRST SEMESTER*



**DEPARTMENT OF BASIC SCIENCE AND HUMANITITES**

**INSTITUTE OF ENGINEERING AND MANAGEMENT, KOLKATA**



CERTIFICATE OF RECOMMENDATION

We hereby recommend that the project prepared under our supervision by ***Asmit Sil* ,** entitled STUDENT EXAMINATION PORTAL be accepted in partial fulfillment of the requirements for the degree of partial fulfillment of the first semester.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Head of the Department Project Supervisor

Basic Sciences and Humanities

IEM, Kolkata

# Introduction

*The project ‘STUDENT EXAMINATION PORTAL’ deals with the performances of students under various courses, subjects and departments in their respective exams. It is used for taking, editing or deleting records of the students, their marks in the exams, the subjects in which the exams are being held, the courses which are being offered and the departments in which the students study and arranging the records in a graphical manner or in a tabular manner as desired by the user. It also provides the performance report of the various students in form of a text file and displays the records in a csv file.*

## 1.1. Objective

*The objective of the project is to ensure that the data entry of the students, their marks, the courses and departments can all be processed in one go to save time as much as possible and also ensure that the data stored is safe and furthermore remove redundancy . The project is also made to be easily accessible and also to ensure that the final result has good clarity and is lucid.*

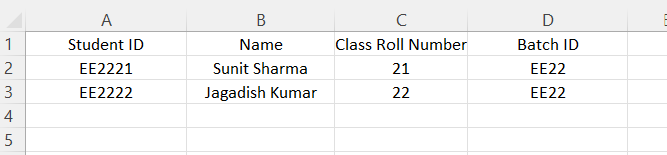
## Organization of the Project

*In this project, the Student table is at the topmost part of the order while Department, Course, Batch fall under it. The system asks for the student data where the marks, department and batch are entered and then, the student table gets prepared and also the department, batch, course data gets filled as per a student.*

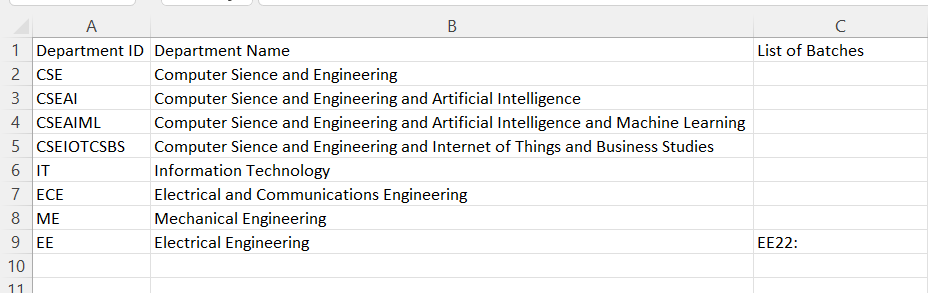
# 2. Database Descriptions

* **Student Database***: It contains the student id, Student Name, Roll number and the Batch id. It contains the basic information related to a student. It contains data in VARCHAR format. The student ID is the Primary Key.*
* **Department Database**: *It contains the list of various departments present and their respective ids along with the batches which were under a particular department. It contains data in VARCHAR format*
* **Course Database**: *It contains the course names which fall under all the departments along with their ids and also the marks received by all the students in the respective courses. It contains data in VARCHAR format.*
* **Batch Database**: *It contains the various batches, their ids, the courses which are offered under those batches and the list of students who are under the batch. It contains data in VARCHAR format.*

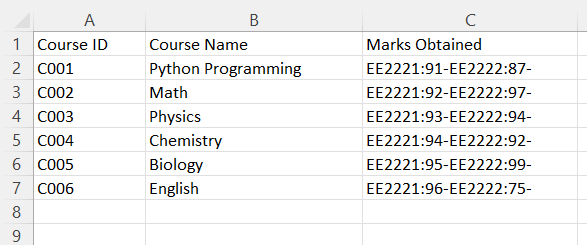
## 2.1. Database Samples:



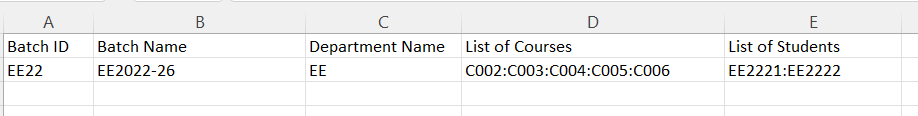
1. *Student Database*



1. *Department Database*

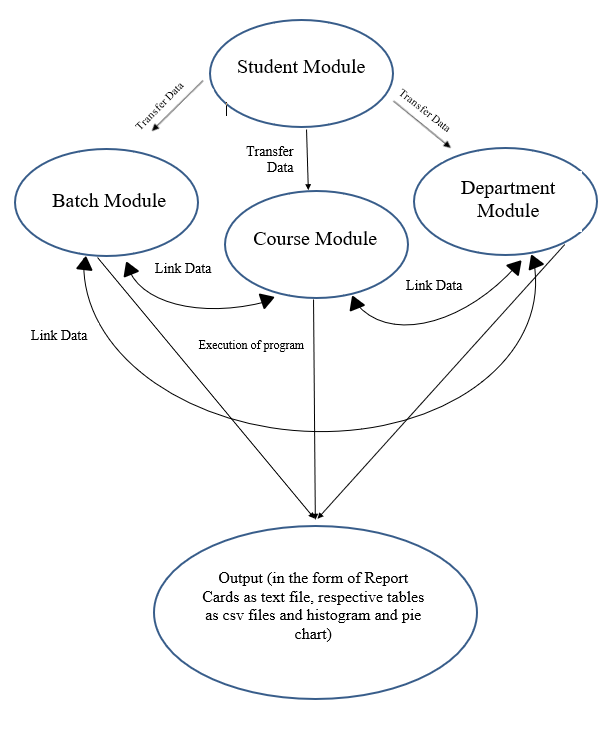


1. *Course Database*



1. *Batch Database*

# Data Flow Diagram:



# Program used:

import os

import csv

import subprocess

import time

import sys

try:

import matplotlib.pyplot as plt

except:

subprocess.run(['pip', 'install', 'matplotlib'])

import matplotlib.pyplot as plt

path='C:/PythonProgrammingProject\_main-folder'

print('-'\*50)

#All the Functions used Throughout the code

def loading\_screen():

for i in range(10):

sys.stdout.write("\rLoading" + "." \* i)

sys.stdout.flush()

time.sleep(0.5)

sys.stdout.write("\rLoading complete!")

def createfile(name,lst):

with open(f'{path}/{name}','a',newline='')as f:

script= csv.writer(f)

script.writerow(lst)

print(f"{name} file has been UPDATED")

def percent(num):

if stream.lower()=='cse' or stream.lower()=='cseai' or stream.lower()=='cseaiml' or stream.lower()=='cseiotcsbs':

num=(num\*100)//600

elif stream.lower()=='it' or stream.lower()=='ece' or stream.lower()=='me' or stream.lower()=='ee':

num=(num\*100)//600

return num

def grade(num):

if num>=90:

return("Outstanding Performance... You have passed the exam with grade A.")

elif num<90 and num>=80:

return("Excellent Performance... You have passed the exam with grade B.")

elif num<80 and num>=70:

return("Good Performance... You have passed the exam with grade C.")

elif num<70 and num>=60:

return("Your performance is average... Work hard... You have passed the exam with grade D.")

elif num<60 and num>=50:

return("Your performance is below average... There is massive scope of improvement... You have barely passed the exam with grade E.")

else:

return("Extremely poor performance... You have Failed the Exam and got F.")

def count(lst):

num=0

for i in lst:

if str(type(i))=="<class 'int'>":

num+=1

else:

pass

return num

def add(lst):

plus=0

for i in lst:

try:

plus+=i

except:

pass

return plus

def duplicate(file,attr,pos=0):

with open(f'{path}/{file}','r') as f:

reader = csv.reader(f)

dup\_lst=[]

for i in reader:

dup\_lst+=[i[pos]]

if attr in dup\_lst:

return True

else:

return False

def choice(stream):

if stream.lower()=='cse' or stream.lower()=='cseai' or stream.lower()=='cseaiml' or stream.lower()=='cseiotcsbs':

return ("C001:C002:C003:C004:C005:C006")

elif stream.lower()=='it' or stream.lower()=='ece' or stream.lower()=='me' or stream.lower()=='ee':

return ("C002:C003:C004:C005:C006")

def get\_batch():

with open(f'{path}/Batch.csv','r') as f:

reader=csv.reader(f)

rows=[row for row in reader]

column=[]

for i in range(len(rows)):

if i==0:

pass

else:

column+=[rows[i][0]]

return column

def remove(string):

with open(f'{path}/Student.csv','r+',newline='') as f:

script=csv.reader(f)

rows=[row for row in script]

for i in rows:

if i[0]==string:

rows[rows.index(i)]=['','','','']

else:

pass

f.seek(0)

f.truncate()

writer=csv.writer(f)

writer.writerows(rows)

def course\_graph():

color\_lst=['#C70039','#9BB1F2','#FFC300','#FF5733','#DAAFB1','#86B7C8']

fig, ax = plt.subplots()

legend\_properties = {'weight':'heavy'}

ax.set\_facecolor("Black")

ax.tick\_params(axis="both", colors="white")

fig.set\_facecolor("Black")

ax.set\_xlabel('Grades--------->', color="white")

ax.set\_ylabel('No. of Students--------->', color="white")

ax.spines["bottom"].set\_color("white")

ax.spines["left"].set\_color("white")

ax.xaxis.label.set\_weight("heavy")

ax.yaxis.label.set\_weight("heavy")

count=0

with open(f'{path}/Course.csv','r')as f:

script= csv.reader(f)

rows=[row for row in script]

req=[]

for i in range(len(rows)):

if i==0:

pass

else:

req+=[rows[i][2]]

lst=[['Python',(req[0].split('-'))[0:-1]],

['Math',(req[1].split('-'))[0:-1]],

['Physics',(req[2].split('-'))[0:-1]],

['Chemistry',(req[3].split('-'))[0:-1]],

['Biology',(req[4].split('-'))[0:-1]],

['English',(req[5].split('-'))[0:-1]]]

for i in range(len(lst)):

for j in range(len(lst[i][1])):

try:

lst[i][1][j]=grade(int((lst[i][1][j].split(':'))[-1]))[-2]

except:

lst[i][1][j]=''

for k in range(6):

a=lst[k][1].count('A')

b=lst[k][1].count('B')

c=lst[k][1].count('C')

d=lst[k][1].count('D')

e=lst[k][1].count('E')

f=lst[k][1].count('F')

lst[k][1]={'A':a,'B':b,'C':c,'D':d,'E':e,'F':f}

for j in lst:

x=list(j[1].keys())

y=list(j[1].values())

ax.plot(x, y,marker=",",color=color\_lst[count],label=j[0],linewidth=3)

leg=plt.legend(fontsize=10,loc="upper right", facecolor="Black",edgecolor="Black",prop=legend\_properties)

count+=1

for text in leg.get\_texts():

text.set\_color('White')

plt.show()

def batch\_graph(arg):

with open(f'{path}/Batch.csv','r') as f:

reader=csv.reader(f)

req=''

rows=[row for row in reader]

for i in range(len(rows)):

if arg==rows[i][0]:

req=rows[i][4]

break

req\_lst=req.split(':')

with open(f'{path}/Course.csv','r') as f:

reader=csv.reader(f)

rows=[row for row in reader]

column=[]

for i in range(len(rows)):

if i==0:

pass

else:

column+=[rows[i][2]]

new\_column=[]

for j in range(len(column)):

new\_column+=(column[j].split('-'))[0:-1]

new\_req\_lst=[]

temp=[]

for i in req\_lst:

for j in range(len(new\_column)):

if i in new\_column[j]:

temp+=[(new\_column[j].split(':'))[-1]]

new\_req\_lst+=[[[i]]+[temp]]

temp=[]

lst=[]

temp=0

grade\_lst=[]

for i in range(len(new\_req\_lst)):

for j in range(6):

try:

temp+=int(new\_req\_lst[i][1][j])

except:

pass

lst+=[new\_req\_lst[i][0]+[temp]]

temp=0

for i in range(len(lst)):

if lst[i][0][:3]=='CSE':

grade\_lst+=[grade((lst[i][1]\*100)//600)[-2]]

lst[i][1]=grade((lst[i][1]\*100)//600)[-2]

else:

grade\_lst+=[grade((lst[i][1]\*100)//600)[-2]]

lst[i][1]=grade((lst[i][1]\*100)//600)[-2]

grade\_no\_lst={'A':grade\_lst.count('A'),'B':grade\_lst.count('B'),'C':grade\_lst.count('C'),'D':grade\_lst.count('D'),'E':grade\_lst.count('E'),'F':grade\_lst.count('F')}

labels = list(grade\_no\_lst.keys())

sizes = list(grade\_no\_lst.values())

color\_lst=['#C70039','#9BB1F2','#FFC300','#FF5733','#DAAFB1','#86B7C8']

explode = (0.01,0.1,0.02,0.05,0.03,0.1)

new\_labels=[]

for i in range(len(labels)):

new\_labels+=[f'{labels[i]} : {str(sizes[i])}']

fig,ax = plt.subplots()

ax.set\_facecolor("Black")

fig.set\_facecolor("Black")

plt.rcParams['font.weight'] = 'heavy'

#plt.rcParams['font.size'] = '1'

patches, texts=ax.pie(sizes, labels=new\_labels, colors=color\_lst,explode=explode,shadow=True,startangle= -90,textprops={'fontsize': 0})

centre\_circle = plt.Circle((0,0),0.60,fc='black')

fig = plt.gcf()

fig.gca().add\_artist(centre\_circle)

legend\_properties = {'weight':'heavy'}

leg=plt.legend(fontsize=10,loc="center", facecolor="Black",edgecolor="Black",prop=legend\_properties)

for text in leg.get\_texts():

text.set\_color('white')

plt.title('Overall Grades vs No. of Students',color='White',weight='heavy')

plt.axis('equal')

plt.show()

def department\_graph():

need={}

with open(f'{path}/Batch.csv','r') as f:

reader=csv.reader(f)

batch=[batch[0] for batch in reader]

batch=batch[1:]

for arg in batch:

avg=0

with open(f'{path}/Batch.csv','r') as f:

reader=csv.reader(f)

req=''

rows=[row for row in reader]

for i in range(len(rows)):

if arg==rows[i][0]:

req=rows[i][4]

break

req\_lst=req.split(':')

with open(f'{path}/Course.csv','r') as f:

reader=csv.reader(f)

rows=[row for row in reader]

column=[]

for i in range(len(rows)):

if i==0:

pass

else:

column+=[rows[i][2]]

new\_column=[]

for j in range(len(column)):

new\_column+=(column[j].split('-'))[0:-1]

new\_req\_lst=[]

temp=[]

for i in req\_lst:

for j in range(len(new\_column)):

if i in new\_column[j]:

temp+=[(new\_column[j].split(':'))[-1]]

new\_req\_lst+=[[[i]]+[temp]]

temp=[]

lst=[]

temp=0

grade\_lst=[]

for i in range(len(new\_req\_lst)):

for j in range(6):

try:

temp+=int(new\_req\_lst[i][1][j])

except:

pass

lst+=[new\_req\_lst[i][0]+[temp]]

temp=0

for i in range(len(lst)):

if lst[i][0][:3]=='CSE':

lst[i][1]=(lst[i][1]\*100)/600

else:

lst[i][1]=(lst[i][1]\*100)/600

for i in range(len(lst)):

avg+=lst[i][1]

avg=int(avg//len(lst))

need[arg]=avg

xdata = list(need.keys())

ydata = list(need.values())

color\_lst=['#C70039','#9BB1F2','#FFC300','#FF5733','#DAAFB1','#86B7C8']

fig,ax = plt.subplots()

ax.set\_facecolor("Black")

fig.set\_facecolor("Black")

ax.set\_xlabel("X axis", color="white")

ax.set\_ylabel("Y axis", color="white")

ax.spines["bottom"].set\_color("white")

ax.spines["left"].set\_color("white")

ax.spines['bottom'].set\_linewidth(2)

ax.spines['left'].set\_linewidth(2)

ax.xaxis.label.set\_weight("heavy")

ax.yaxis.label.set\_weight("heavy")

ax.tick\_params(axis='x', labelcolor='white', labelsize=10,color='white',width=2)

ax.tick\_params(axis='y', labelcolor='white', labelsize=10,color='white',width=2)

plt.barh(xdata,ydata,color=color\_lst,height=0.3,align='center')

plt.title('Histogram of Average of Students vs Batch',color='white',pad=17,fontweight='bold')

plt.xlabel('Average----------------->')

plt.ylabel('Batch----------------->', labelpad=15)

plt.show()

#Creation of Folder and all the Modules recquired...

try:

os.makedirs(f'{path}/ReportCards')

message=True

except:

message=False

while message:

createfile('Batch.csv',['Batch ID','Batch Name','Department Name','List of Courses','List of Students'])

createfile('Course.csv',['Course ID','Course Name','Marks Obtained'])

with open(f'{path}/Course.csv','a',newline='')as f:

script= csv.writer(f)

script.writerow(['C001','Python Programming'])

script.writerow(['C002','Math'])

script.writerow(['C003','Physics'])

script.writerow(['C004','Chemistry'])

script.writerow(['C005','Biology'])

script.writerow(['C006','English'])

createfile('Department.csv',['Department ID','Department Name','List of Batches'])

with open(f'{path}/Department.csv','a',newline='')as f:

script= csv.writer(f)

script.writerow(['CSE','Computer Sience and Engineering'])

script.writerow(['CSEAI','Computer Sience and Engineering and Artificial Intelligence'])

script.writerow(['CSEAIML','Computer Sience and Engineering and Artificial Intelligence and Machine Learning'])

script.writerow(['CSEIOTCSBS','Computer Sience and Engineering and Internet of Things and Business Studies'])

script.writerow(['IT','Information Technology'])

script.writerow(['ECE','Electrical and Communications Engineering'])

script.writerow(['ME','Mechanical Engineering'])

script.writerow(['EE','Electrical Engineering'])

createfile('Student.csv',['Student ID','Name','Class Roll Number','Batch ID'])

createfile('Examination.csv',['Course Name','Student ID','Marks'])

break

print('\n','Computer Sience and Engineering : CSE','\n',

'Computer Sience and Engineering and Artificial Intelligence : CSEAI','\n',

'Computer Sience and Engineering and Artificial Intelligence and Machine Learning : CSEAIML','\n',

'Computer Sience and Engineering and Internet of Things and Business Studies : CSEIOTCSBS','\n',

'Information Technology : IT','\n',

'Electrical and Communications Engineering : ECE','\n',

'Mechanical Engineering : ME','\n',

'Electrical Engineering : EE','\n')

print("Please write all the stream name in short form as mentioned above and in capital letters only!!!")

print()

student\_no=int(input("Enter the no. of students whose data you want to input : "))

print()

print('-'\*50)

for i in range(student\_no):

name=input("Enter Student's Name : ")

batch=input("Which batch they are in (e.g. 2022-26) : ")

stream=input("Which Stream are you in (e.g. CSE) : ")

roll=input("What is your Class Roll Number : ")

batch\_id=stream+batch[2:4]

student\_id=batch\_id+roll

batch\_name=stream+batch

if duplicate('Student.csv',student\_id,0):

print("the student is already present in the directory")

print(f"You can find your report card here : {path}/ReportCards/{student\_id}\_{name}.txt")

else:

print()

print("The subjects are [Python,Math,Physics,Chemistry,Biology,English]")

print('please enter the subjects marks in the above mentioned order in a list type and if you dont have a particular subject write there "null" (e.g. [100,100,"null",75,69,85])')

print('Each Subject is ot of 100 marks')

print()

marks\_lst=eval(input("Enter the Marks list : "))

total\_marks=add(marks\_lst)

print()

with open(f"{path}/ReportCards/{student\_id}\_{''.join(name.split())}.txt",'w') as f:

f.writelines([f'Name of the student : {name} \n',

f'Class Roll of the student : {roll} \n',

f'Stream of the student : {stream} \n',

f'Your Student ID is : {student\_id}\n',

'\n',

f'Marks obtained in Math is : {marks\_lst[1]} \n',

f'Marks obtained in Python is : {marks\_lst[0]} \n',

f'Marks obtained in Physics is : {marks\_lst[2]} \n',

f'Marks obtained in Chemistry is : {marks\_lst[3]} \n',

f'Marks obtained in Biology is : {marks\_lst[4]} \n',

f'Marks obtained in English is : {marks\_lst[5]} \n'])

f.write('\n')

f.write(f'You have got {total\_marks} in total with {percent(total\_marks)}%\n')

f.write(grade(total\_marks/count(marks\_lst)))

createfile('Student.csv',[student\_id,name,roll,batch\_id])

print(f"You can find your report card here : {path}/ReportCards/{student\_id}\_{''.join(name.split())}.txt")

openpath=f"{path}/ReportCards/{student\_id}\_{''.join(name.split())}.txt"

subprocess.run(['start',openpath], shell=True)

ask=input("Do you want to remove this name from database now is the time (Y/N) : ")

if ask.lower()=='n':

if duplicate('Batch.csv',batch\_id,0):

with open(f'{path}/Batch.csv','r+',newline='') as f:

script=csv.reader(f)

rows=[row for row in script]

for i in rows:

if batch\_id==i[0]:

rows[rows.index(i)][4]+=f':{student\_id}'

f.seek(0)

f.truncate()

writer=csv.writer(f)

writer.writerows(rows)

print("Batch.csv has been updated")

else:

createfile('Batch.csv',[batch\_id,batch\_name,stream,choice(stream),student\_id])

with open(f'{path}/Course.csv','r+',newline='') as f:

script=csv.reader(f)

rows=[row for row in script]

for i in range(len(rows)):

if i==0:

pass

else:

try:

rows[i][2]+=f'{student\_id}:{marks\_lst[i-1]}-'

except:

rows[i].append(f'{student\_id}:{marks\_lst[i-1]}-')

f.seek(0)

f.truncate()

writer=csv.writer(f)

writer.writerows(rows)

else:

remove(student\_id)

subprocess.call("TASKKILL /F /IM notepad.exe", shell=True)

os.remove(openpath)

print('Your details have been successfully removed from the directory')

print('-'\*50)

print()

try:

with open(f'{path}/Department.csv','r+',newline='') as f:

script=csv.reader(f)

rows=[row for row in script]

lst=get\_batch()

for i in lst:

for j in rows:

if i[0:-2]==j[0]:

try:

if i in j[2]:

pass

else:

rows[rows.index(j)][2]+=f'{i}:'

except:

rows[rows.index(j)].append(f'{i}:')

break

f.seek(0)

f.truncate()

writer=csv.writer(f)

writer.writerows(rows)

except:

print("Nothing to add in Department.csv")

#Creation of the Graphs...

print()

print("Give the details Below to see the Batchwise percent Graph")

batch=input("Which batch they are in (e.g. 2022-26) : ")

stream=input("Which Stream are they in (e.g. CSE) : ")

print('Please Close the Figure window after viewing to continue')

batch\_id=stream+batch[2:4]

with open(f'{path}/Batch.csv','r') as f:

reader=csv.reader(f)

batch=[batch[0] for batch in reader]

batch=batch[1:]

while True:

if batch\_id in batch:

batch\_graph(batch\_id)

break

else:

print(f'details with {batch\_id} this Batch ID is not in the directory')

ask=input("Do you want to continue (y/n) : ")

if ask.lower()=='y':

batch=input("Which batch they are in (e.g. 2022-26) : ")

stream=input("Which Stream are they in (e.g. CSE) : ")

batch\_id=stream+batch[2:4]

continue

else:

print('OK')

break

print()

print('The overall Course graph will come now')

print('Please Close the Figure window after viewing to continue')

loading\_screen()

course\_graph()

print()

print()

print("The overall Department wise average graph will come now")

print('Please Close the Figure window after viewing to continue')

loading\_screen()

department\_graph()

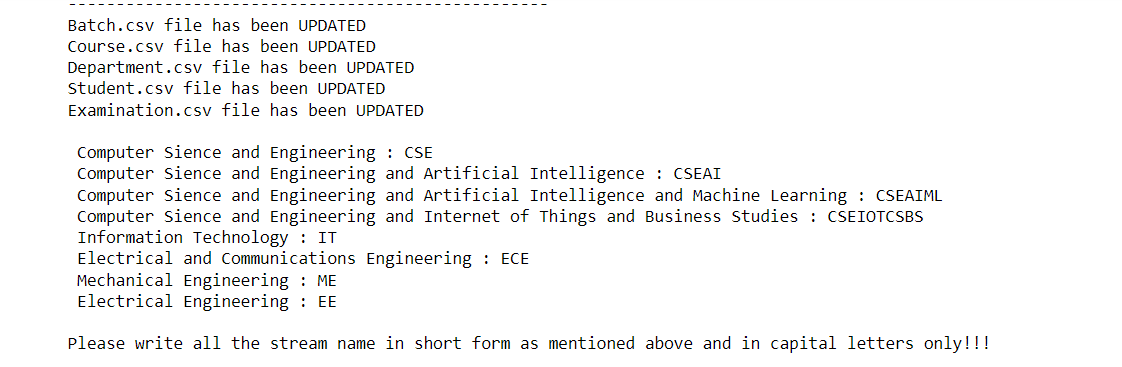
print()

print()

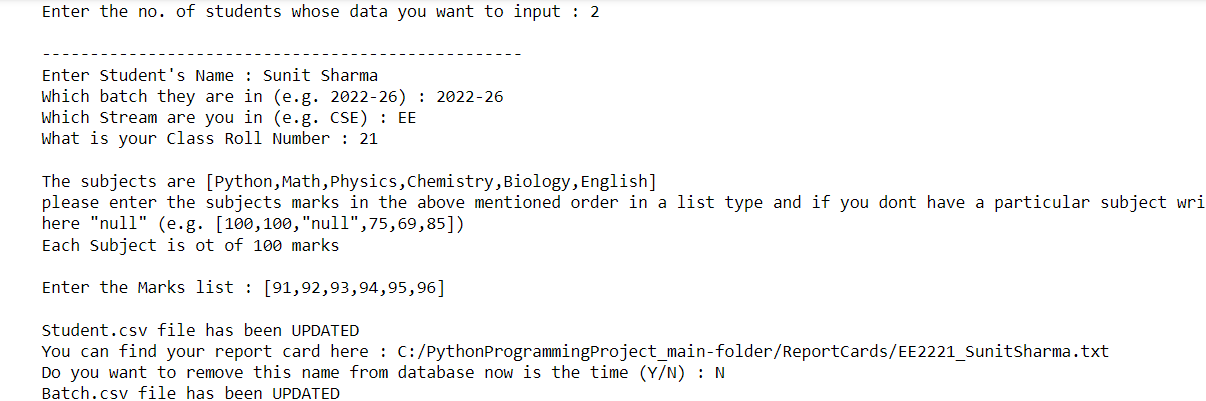
last=input("Press Enter to exit")

subprocess.call("TASKKILL /F /IM notepad.exe", shell=True)

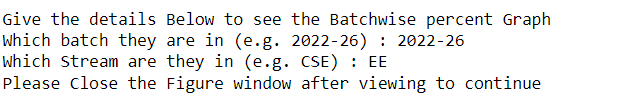
# Outputs



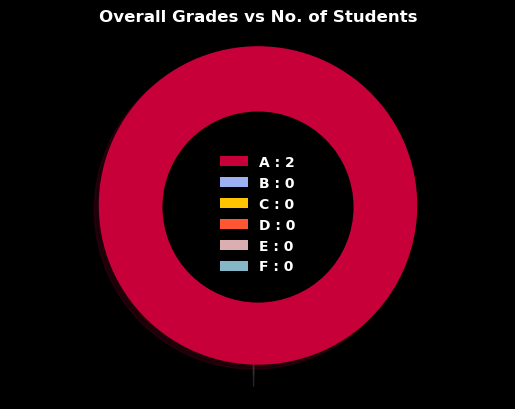
1. *Display of various batches*



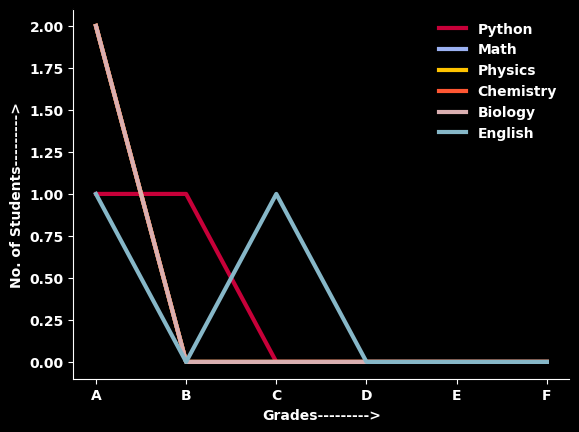
1. *Entry of data*



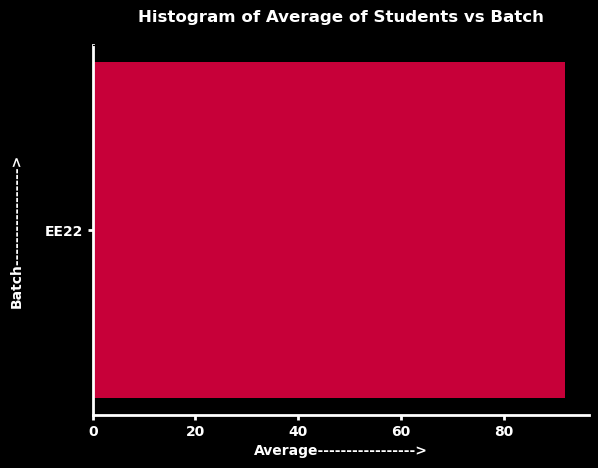
1. *Choosing a particular index for viewing graph*



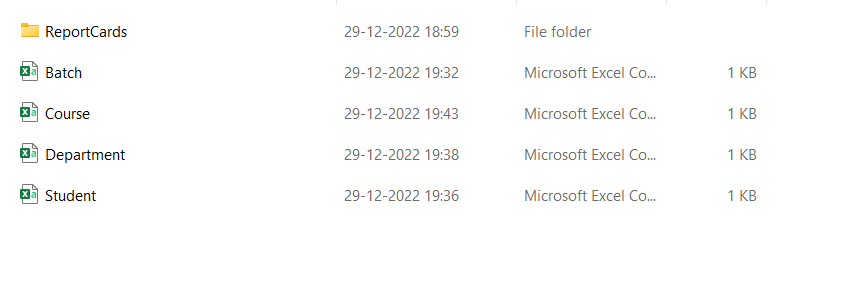
1. *Pie Chart displaying the Overall Grades of different Students*



1. *Graph displaying the Grades of the students in various courses*



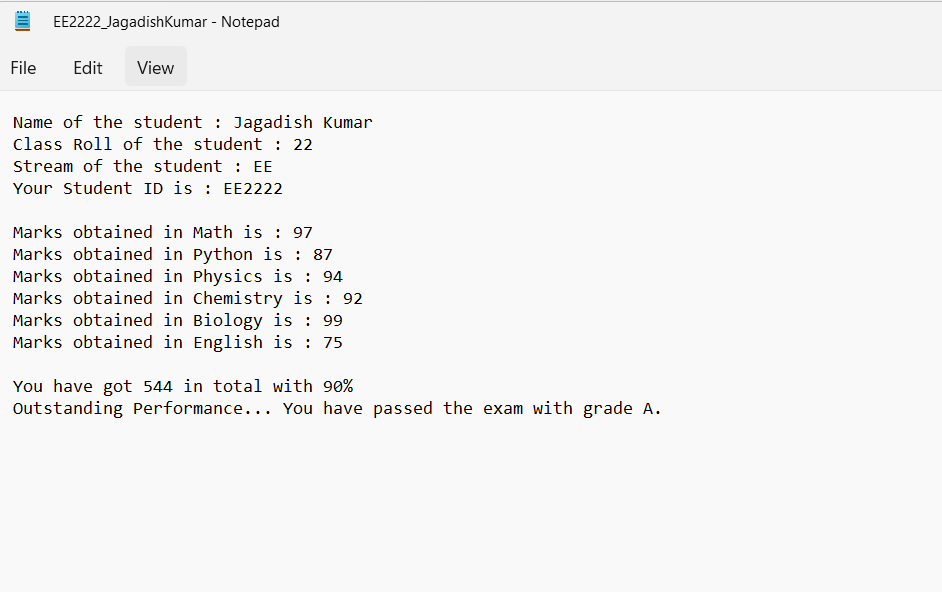
1. *Histogram displaying the average students in a batch*



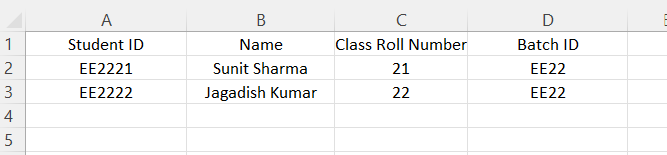
1. *Creation of a folder of Report Cards, csv files of Batch, Course, Department and Student after execution of code*



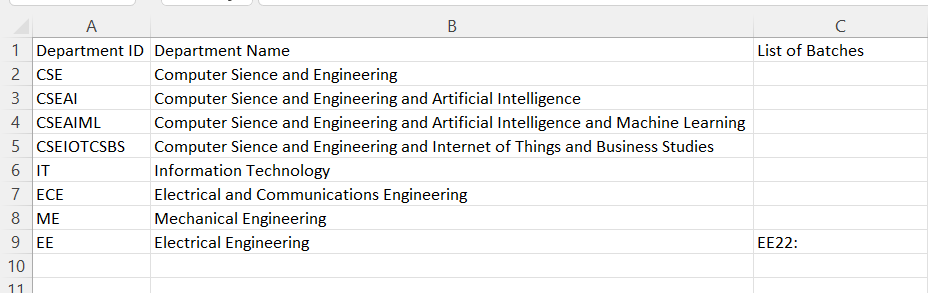
1. *Creation of Report cards as text files*



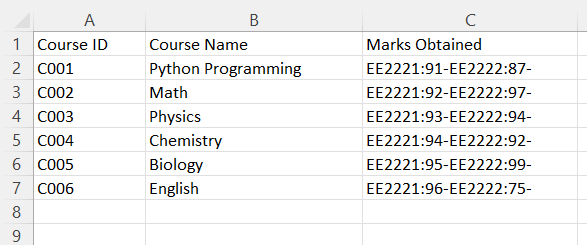
1. *Report Card text file*



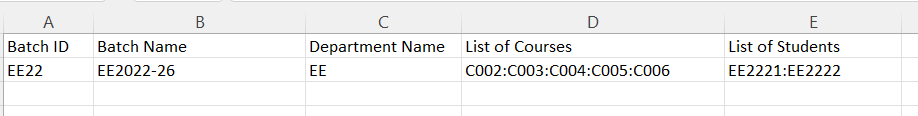
1. *Student.csv file contents*



1. *Department.csv file contents created*



1. *Course.csv file contents after creation*



1. *Batch.csv file created*