Afrin Rukdar
64-D
2/1/2023

Date
Page
Stalker

```
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:/
Python Programming Project_main-fol
der'
print('_'*50)

#All the functions used throughout the code
def loading_screen():
  for i in range(10):
    sys.stdout.write("\r Loading" + "" "." *i)
    sys.stdout.flush()
    time.sleep(0.5)
sys.stdout.write("\r Loading 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
```

Date
Page

```
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':
 num = (num*100)//500
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 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 the exam with grade E.")
else:
return("Extreamly 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':
    return
("C002:C003:C004:C005:C006")

def get_batch():
with open(f'C:/
Python Programming Project_main_fol
der/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'C:/
Python programming project_main_fol der/studen
.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()
write = csv.write(f)
write.writerows(rows)

def course_graph():
colour_lst = ['#C70039','#9BB1F2','#FFC300','#FF5733','#DAAFB1','#86B7C8']
fig, ax = plt.subplots()
legend_properties = {'weight':'heavy'}
ax.set_facecolour("Black")
ax.tick_params(axis="both", colours="white")
fig.set_facecolour("Black")
ax.set_xlabel('Grades ---->', colour="white")
ax.set_ylabel('No of students ---->', colour="white")

ax.spines["buttom"].set_colour("white")
ax.spines["left"].set_colour("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])
  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 = 'o', colour = colour_lst[count],
 lab el = j[0], linewidth = 3)

leg = plt.legend(fontsize = 10, loc = "upper right",
facecolour = "Black", edgecolour = "Black", prop = legend
_properties)
count += 1
```

```
for text in dig.get_texts():
    text.set_colour('white')
plt.show()
def batch_graph(req):
    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 req==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)//500)[-2]]
        lst[i][1] = grade((lst[i][1]*100//5000)[-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())

Colour_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_facecolour("Black")
fig.set_facecolour("Black")
plt.rcParams['font.weight'] = 'heavy'
#plt.rcParams['font.size'] = 1

patches, texts = ax.pie(sizes, labels = new_labels,
colours = colour_list, 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",
facecolour = "Black", edgecolour = "Black",
prop = legend_properties)
for text in leg.get_texts():
text.set_colour('white')

plt.title('overall grades vs No. of students', colour
= '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:
arg = 0
with open(f'{path}/Batch.csv','r') as f:
reader = csv.reader(f)
req = ""
rows = [rows 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][7])
except:
pass
lst+=[new_req_lst[i][0]+[temp]] tem=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)/500
for i in rang(len(lst)):
avg+=lst[i][1]
avg=int(avg//len(lst))
need[avg]=avg

x_data=list(need.keys())
y_data=list(need.values())

colour_lst=['#C70039','#9BB1F2','FFC300','#FF
5733','#DAAFB1','#86B7C8']
fig,ax=plt.subplots()
ax.set_facecolour("Black")
ax.set_facecolour("Black");
ax.set_xlabel("xaxis",colour="white")
ax.set_ylabel("yaxis",colour="white")
```

```
ax.spines["bottom"].set_colour("white")
ax.spines["left"].set_colour("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.label.set_weight("heavy")
ax.tick_params(axis='x', label colour='white',
labelsize=10, colour='white', width=2)
ax.tick_params(axis='y',
labelcolour='white',
labelsize=10, colour='white', width=2)

plt.barh(xdata, ydata, colour=colour_lst, height
=0.3, align='centre')
plt.title("Histogram of average students vs Batch',
colour='white', pad=17, fontweight='bold')
plt.xlabel('Average ---->')
plt.ylabel('Batch ---->', labelpad=15)
plt.show()

#creation of folder and all modules required
try:
os.makedirs(f'{path}/Reportcards')
message=True
except:
message=False
if message:
createfile('Batch.csv',['Batch ID','Batch Name','Departme
name','List of courses','List of students'])
createfile('course.csv',['Course ID','Course Nam
','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(['C004', 'Biology'])
    script.writerow(['C005', 'English'])
createfile('Department.csv',
['Department ID', 'Department Name', 'List of Branches'])
with open(f'{path}/Department.csv', 'a', newline='') as f:
    script = csv.writer(f)
    script.writerow(['CSE', 'Computer Science and
Engineering'])
    script.writerow(['CSEAI', 'Computer Science and
Engineering and Artificial Intelligence'])
    script.writerow(['CSEAIML', 'Computer Science and
Engineering and Artificial Intelligence and Machine
Learning'])
    script.writerow(['CSEIOTCSBS', 'Computer Science and
Engineering and Internet of Things and Business
Studies'])
    script.writerow(['IT', 'Information Technology'])
    script.writerow(['ECE', 'Electrical and Communicat-
ion Engineering'])
    script.writerow(['ME', 'Mechanical Engineering'])
Createfile('Student.csv', ['Student ID', 'Name', 'Class
Roll Number', 'Batch ID'])
Createfile('Examination.csv', ['Course Name', 'Student
ID', 'Marks'])
break
```

print('\n', 'Computer Science and Engineering : CSE', '\n', 'Computer Science and Engineering and Artificial Intelligence : CSEAI', '\n', 'Computer Science Engineering and Artificial Intelligence and Machine learning : CSEAIML', '\n', 'Computer Science and Engineering and Internet of Things and Business Studies : CSEIOTCSBS', '\n', 'Information Technology : IT', '\n', 'Electrical and Communication Engineering : ECE', '\n', 'Mechanical Engineering : ME', '\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 are you in :")

Stream = input("Which Stream are you in :")

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}/Report Cards/{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")
print('Each subject is of of 100 marks')
print()
marks_lst = eval(input("Enter the marks list: "))
total_marks = add(marks_lst)
print()

with open(f"{path}/Report cards/{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
Maths 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-list))
create_file('student.csv',[student-id,name,roll,batch
id])
print(f"you can find your report card here: {path}/
Report cards/{student-id}-{"".join(name.split())}.
txt")
openpath = f"{path}/Report cards/{student-id}-
{"".join.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:
```

Date
Page
Staller

```
createfile('Batch.csv',['batch_id','batch_name','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 notpad.exe",
shell = True)
os.remove(openpath)
print("your details has been sucessfully 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 graphs....
print()
print("Give the detail below to see the batches percent graph")
batch = input("Which batch they are in:")
stream = input("Which stream are they in:")
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:")
stream = input("which stream they are in:")
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 ~~exist~~ exit"))
subprocess.call("~~TASKKILL~~ TASKKILL /F /IM
notepad.exe", shell = True)