from excel\_handler import ExcelHandler  
from file\_handler import \*  
from query\_handler import QueryHandler  
from graph\_handler import GraphHandler  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 try:  
  
 # <editor-fold desc="Ex1">  
  
 # reading data from file  
 threats=FileHandler('C:/final/final\_assignment/threats.txt')  
 threats\_from\_file=threats.read\_file\_content()  
  
 #creating a db handler  
 q = QueryHandler("localhost", "final\_assignment", "root", "")  
  
 #sending a query for each row of data  
 for row in threats\_from\_file:  
 row.append('0')  
 values = tuple(row)  
  
 #entering the data to the db  
 query="INSERT INTO attacks (attack\_type , date , attacker\_ip , country, duration\_sec , data\_leaked\_mb, is\_investigated) VALUES (%s, %s, %s,%s, %s, %s, %s)"  
 q.execute\_non\_fetch(query, values)  
  
 #gettint the 10 most attacking ip addresses  
 query1="SELECT attacker\_ip , COUNT(attacker\_ip) as num from attacks GROUP BY attacker\_ip ORDER BY num DESC limit %s;"  
 print("The 10 most attacking ip addresses")  
 print(q.execute\_fetch(query1, (10,)))  
  
 #readin data from blacklist file  
 blacklist = FileHandler('C:/final/final\_assignment/blacklist.txt')  
 blacklist\_from\_file=blacklist.read\_file\_content()  
  
 #updating every ip from the blacklist file  
 for ip in blacklist\_from\_file:  
 query2 = "UPDATE attacks SET is\_investigated = %s WHERE attacker\_ip = %s"  
 values\_to\_query2=('1',ip)  
 q.execute\_non\_fetch(query2, values\_to\_query2)  
 # </editor-fold>  
  
 # <editor-fold desc="Ex2">  
 print("Ex2")  
 empty\_tuple = ()  
 attack\_type\_query="SELECT attack\_type, COUNT(\*) as num FROM attacks GROUP BY attack\_type"  
  
 #getting the data from db  
 attack\_dict\_from\_db=q.execute\_fetch(attack\_type\_query, empty\_tuple)  
  
 #prepering empty lists  
 attack\_type=[]  
 amount=[]  
  
 #prepering the lists for the bar graph  
 for dict\_from\_db in attack\_dict\_from\_db:  
 attack\_type.append(dict\_from\_db['attack\_type'])  
 amount.append(dict\_from\_db['num'])  
  
  
  
 #creating bar graph to show amount for each attack type  
 GraphHandler.bar("Attack Type", "Amount", attack\_type, amount)  
  
 #getting the country data from db  
 country\_query = "SELECT country, COUNT(\*) as num FROM attacks GROUP BY country"  
 country\_dict\_from\_db = q.execute\_fetch(country\_query, empty\_tuple)  
  
 #prepering empty lists  
 country=[]  
 country\_amount=[]  
  
 #filling the list from the data  
 for country\_dict in country\_dict\_from\_db:  
 country.append(country\_dict['country'])  
 country\_amount.append(country\_dict['num'])  
  
 #sending the data to graph handler to show as pie graph  
 GraphHandler.pie(country\_amount,country, 90 )  
  
 #getting the duration/mb data from db  
 dur\_mb\_query = "SELECT duration\_sec, data\_leaked\_mb FROM attacks "  
 duration\_dict\_from\_db = q.execute\_fetch(dur\_mb\_query, empty\_tuple)  
  
 #prepering empty lists  
 dur = []  
 mb\_leaked = []  
  
 #filling the list from the data  
 for dur\_mb in duration\_dict\_from\_db:  
 dur.append(dur\_mb['duration\_sec'])  
 mb\_leaked.append(dur\_mb['data\_leaked\_mb'])  
  
 GraphHandler.scatter\_plot(dur,mb\_leaked)  
 # </editor-fold>  
  
 # <editor-fold desc="Ex3">  
  
 print("Ex3")  
 #getting the data from the activity log file  
 activity\_log = FileHandler('C:/final/final\_assignment/activity\_log.txt')  
 activity\_log\_file = activity\_log.read\_file\_content()  
  
 #creating the data to enter the new activity log file  
 susp\_activity=[]  
 for activity in activity\_log\_file:  
 if activity[len(activity)-1].strip()=='FAILURE':  
 susp\_activity.append(activity)  
  
 activity\_log.write\_log\_file('suspicious\_activity.txt',susp\_activity)  
  
 no\_duplicate=[]  
 for activity in activity\_log\_file:  
 if activity[len(activity) - 1].strip() == 'FAILURE':  
 if activity[0] not in no\_duplicate:  
 no\_duplicate.append(activity[0])  
 activity\_log.write\_log\_file('suspicious\_ips.txt', no\_duplicate)  
  
 # </editor-fold>  
  
 # <editor-fold desc="Ex4">  
  
 print("Ex4")  
 # creating an ExcelHandler for malware.xlsx  
 ex=ExcelHandler('malware.xlsx')  
  
 #reading al the types  
 malware\_type=ex.read\_specific\_columns('C')  
  
 #reading al the severities  
 severity=ex.read\_specific\_columns('D')  
  
 #prepering lists for the graphs  
 list\_type=[]  
 list\_avg=[]  
  
 #going over the types of malware  
 for i in range (len(malware\_type)):  
  
 #reseting the counter and sum  
 count=0  
 sum\_of\_sev=0  
 current\_type=malware\_type[i]  
  
 #checking if the current type was already checked  
 if current\_type not in list\_type:  
 list\_type.append(current\_type)  
  
 #going over the rest of the list  
 for j in range (i,len(malware\_type)):  
  
 #checking if the type from the rest of the list equals the current type  
 if malware\_type[j]== current\_type:  
 count+=1  
 sum\_of\_sev+=int(severity[j])  
  
 #adding the avg to the list (the count can never be zero so it is ok)  
 list\_avg.append((sum\_of\_sev/count))  
  
 #making a graph to present the Malware and their severity avg  
 GraphHandler.bar("Malware" , "Severity avg" ,list\_type, list\_avg)  
  
 #printing the number of lines in the file  
 print("The number of lines in the file is:",len(severity))  
  
 #printing the amount of the biggest severity  
  
 max\_sev=max(severity)  
 count\_sev=severity.count(max\_sev)  
  
 print("The max severity is: ",max\_sev)  
 print("And the amount is appears in the file is: ",count\_sev)  
  
  
  
  
 # </editor-fold>  
  
 except BaseException as b:  
 print("error", b)