|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| A1 | G1 | 24 | 20 | 7 | 4 | 0 | 55 |
| G2 | 8 | 28 | 11 | 9 | 0 | 56 |
| G3 | 5 | 18 | 14 | 2 | 1 | 40 |

A1- mean Student A

G1, G2, and G3 means students be given grades

**Step 1**

1. Student G1 give the student A grade 24 for question 1.
2. Student G2 give the student A grade 8 for question 1.
3. Student G3 give the student A grade 5 for question 1 .

And So on for the other columns

1. Student G1 give the student A grade 20 for question 2.
2. Student G2 give the student A grade 28 for question 2.
3. Student G3 give the student A grade 18for question 2 .

Etc.

**Step 2**

1. Your system will take the excel sheet and start to analyze the data as written above.
2. Any grade empties the system will fill the cell with value 0 but not in the excel sheet should be in reading data in your system.
3. Your system will sort this values from less to high for Example .

* Student G3 give the student A grade 5 for question 1.
* Student G2 give the student A grade 8 for question 1.
* Student G1 give the student A grade 24 for question 1 .

And So on for the other columns

* Student G3 give the student A grade 18 for question 2.
* Student G1 give the student A grade 20 for question 2.
* Student G2 give the student A grade 28 for question 2.

**Step 3**

**We will depend on the middle number.**

**For Example .**

* Student G3 give the student A grade 5 for question 1.
* Student G2 give the student A grade 8 for question 1.
* Student G1 give the student A grade 24 for question 1.

Then apply the below code but this code may have some syntax errors because I’m not expert in Python.

temp.sort()  
# if temp[0]<temp[1]/1.5:  
# G1 = temp[1]/1.5 - temp[0]  
# if temp[2]>temp[1]\*1.5:  
# G2 = temp[2]-temp[1]\*1.5  
# else:  
#  
# if G1>G2 and G1>1:  
# ot = temp[0]  
# elif G2>G1 and G2>1:  
# ot = temp[2]  
#   
# # for i in range(3):  
# # if ot==val[i]:  
# # return i

And So on for the other columns

* Student G3 give the student A grade 18 for question 2.
* Student G1 give the student A grade 20 for question 2.
* Student G2 give the student A grade 28 for question 2.

Then apply the below code but this code may have some syntax errors because I’m not expert in Python.

temp.sort()  
# if temp[0]<temp[1]/1.5:  
# G1 = temp[1]/1.5 - temp[0]  
# if temp[2]>temp[1]\*1.5:  
# G2 = temp[2]-temp[1]\*1.5  
# else:  
#  
# if G1>G2 and G1>1:  
# ot = temp[0]  
# elif G2>G1 and G2>1:  
# ot = temp[2]  
#   
# # for i in range(3):  
# # if ot==val[i]:  
# # return i

**And So on for the whole grades.**

* Then Create New Excel with the correct values and fixed wrong cells and highlight the fixed cells.

**You can see below the whole code but it still have some issues and need some on expert to fix it .**

# import pandas as pd  
# df = pd.read\_excel ('book1.xlsx')  
# #print(df)  
# # show it in html file  
# html = df.to\_html()  
# #write html to file  
# text\_file = open("source.html", "w")  
# text\_file.write(html)  
# text\_file.close()  
#  
# #define function to find out the outlier  
# def find\_outlier(val):  
# temp=[]  
# for i in range(3):  
# temp.append(val[i])  
# 3 G1,G2=(0,0)  
# temp.sort()  
# if temp[0]<temp[1]/1.5:  
# G1 = temp[1]/1.5 - temp[0]  
# if temp[2]>temp[1]\*1.5:  
# G2 = temp[2]-temp[1]\*1.5  
# else:  
#  
# if G1>G2 and G1>1:  
# ot = temp[0]  
# elif G2>G1 and G2>1:  
# ot = temp[2]  
#   
# # for i in range(3):  
# # if ot==val[i]:  
# # return i  
  
#  
# # create a function to return the average of outlying set  
# def Find\_Avg(val,pos):  
# #print(sum(val))  
# # return (sum(val) - val[pos])/2  
# # to find new total grades after removing outliers  
# def Set\_Total(val):  
# return int(sum(val)/3)  
#  
# #create a blank dataframe  
#  
# i = 0  
# col\_header = ['Ans\_ID','CR11','CR12','CR13','CR14','CR15','CR21','CR22','CR23','CR24','CR25','CR31','CR32','CR33','CR34','CR35','TOTAL']  
# NewData = pd.DataFrame()  
# for i in col\_header:  
# NewData[i]=[]  
# start=0  
# record=[]  
# ans\_code=''  
# i=0  
# counter = 0  
# Data\_with\_outlier = pd.DataFrame(NewData)  
#  
# for start in range(start,len(df),3):  
# temp1=[]  
# ans\_code=df.iloc[start][0]  
# temp = pd.DataFrame(df.iloc[start:start+3,2:7])  
# i=0  
# #copy 3 records/rows in 1 row  
# total = 0  
# for i in range(0,3):  
# for j in range(0,5):  
# temp1.append(temp.iloc[i][j])  
# total = total + temp.iloc[i][j]  
# temp1.insert(0,ans\_code)  
# total = int(total/3)  
# temp1.insert(len(temp1), total)  
# # print(temp1)  
# NewData\_len = len(NewData)  
# NewData.loc[NewData\_len] = temp1  
# temp1=[]  
# #print(NewData)  
# # Write this data to html file source.html  
# #write html to file  
# html = NewData.to\_html()  
# text\_file = open("source\_New.html", "w")  
# text\_file.write(html)  
# text\_file.close()  
# #print the CR groups  
# i=1  
# temp1=[]  
# j=1  
#  
# for row\_count in range(NewData\_len):  
# for i in range(1,6):  
# #print("Set "+str(i))  
# counter = i  
# test\_data = []  
# testdata\_index = []  
# for j in range(0,3):  
# test\_data.append(NewData.iloc[row\_count,counter])  
# testdata\_index.append(counter)  
# counter = counter +5  
# #call function outlier  
# #print(testdata\_index)  
# #print(test\_data)  
# ot = find\_outlier(test\_data)  
# #print("Outlier position: " + str(ot))  
# if ot<3:  
# #print("The original value:")  
# #print(NewData.iloc[row\_count,testdata\_index[ot]])  
# NewVal = Find\_Avg(test\_data,ot)  
# #print("New Value:", str(NewVal))  
# NewData.iloc[row\_count,testdata\_index[ot]] = NewVal  
# #NewData.iloc[row\_count,testdata\_index[ot]] = str(NewVal) + "\*"  
# #print("The value with outlier:")  
# #print(NewData.iloc[row\_count,testdata\_index[ot]])  
# New\_Total = Set\_Total(NewData.iloc[row\_count,1:15])  
# #print("New Total:" + str(New\_Total))  
# NewData.iloc[row\_count,16] = New\_Total  
# print(NewData)  
# html = NewData.to\_html()  
# text\_file = open("source\_Outlier.html", "w")  
# text\_file.write(html)  
# text\_file.close()  
#  
# # This function is to find out the outlier  
#  
#