Question 8 - Extend the application developed in (7) to support relative grading which uses the class average (mean) and standard deviation to compute the cutoffs for various grades as opposed to fixing them statically; you can refer the sample grader (excel sheet) attached to understand the formulas for fixing the cutoffs; the grader would involve, mean, standard deviation, max mark, passed students data mean, etc. Understand the excel grader thoroughly before you try mimicking such an application in your development platform. Formulas Required for Relative Grading: Passing Minimum: 50% of class average. (Minimum marks for passing) X= Passing Students' Mean- Passing Minimum. S_cutoff = Max_Mark – 0.1 *(Max_Mark-Passing Students Mean) Y= S_cutoff – Passing Students Mean A_cutoff = Passing Students Mean + Y* (5/8) B_cutoff = Passing Students Mean + Y *(2/8) C_cutoff = Passing Students Mean - X* (2/8) D_cutoff = Passing Students Mean - X * (5/8) E_cutoff = Passing Minimum Maroon shows failure ( " U grade "); similar to a heatmap...cold to warm. The color scheme is automatic and the least grade ends up with red-maroon shade

In [ ]:

```python
import random
from random import randint
import numpy as np
import pandas as pd
```

In [ ]:

```python
MidSem = [random.randint(0, 30) for i in range(20)]
EndSem = [random.randint(0, 50) for i in range(20)]
Assignments = [random.randint(0, 20) for i in range(20)]
TotalMarks = np.array([MidSem[i] + EndSem[i] + Assignments[i] for i in range(2
Grades = ['S', 'A', 'B', 'C', 'D', 'E', 'U']
```

In [ ]:

```python
Mean = TotalMarks.mean()
PassingMinimum = Mean/2
PassingMarks = TotalMarks[TotalMarks > PassingMinimum]
PassingMean = PassingMarks.mean()
MaximumMarks = TotalMarks.max()
```

In [ ]:

```python
Mean
```

Out[ ]:

```
48.0
```

In [ ]:

```python
PassingMinimum
```

Out[ ]:

```
24.0
```

In [ ]:

```python
PassingMarks
```

Out[ ]:

```
array([80, 67, 33, 42, 30, 42, 48, 59, 30, 53, 39, 82, 59, 32, 65, 67, 88])
```

In [ ]:

```python
PassingMean
```

```
Out[ ]: 53.88235294117647
```

```
In [ ]: MaximumMarks
```

```
Out[ ]: 88
```

```
In [ ]: X = PassingMean - PassingMinimum
        S_cutoff = MaximumMarks - 0.1*(MaximumMarks - PassingMean)
        Y = S_cutoff - PassingMean
        A_cutoff = PassingMean + Y*(5/8)
        B_cutoff = PassingMean + Y*(2/8)
        C_cutoff = PassingMean - X*(2/8)
        D_cutoff = PassingMean - X*(5/8)
        E_cutoff = PassingMinimum
```

```
In [ ]: S_cutoff
```

```
Out[ ]: 84.58823529411765
```

```
In [ ]: A_cutoff
```

```
Out[ ]: 73.07352941176471
```

```
In [ ]: B_cutoff
```

```
Out[ ]: 61.55882352941177
```

```
In [ ]: C_cutoff
```

```
Out[ ]: 46.411764705882355
```

```
In [ ]: D_cutoff
```

```
Out[ ]: 35.205882352941174
```

```
In [ ]: E_cutoff
```

```
Out[ ]: 24.0
```

In [ ]:

```
def RelativeGrading(TotalMarks):
    if(TotalMarks >= S_cutoff):
        return('S')
    elif(TotalMarks >= A_cutoff):
        return('A')
    elif(TotalMarks >= B_cutoff):
        return('B')
    elif(TotalMarks >= C_cutoff):
        return('C')
    elif(TotalMarks >= D_cutoff):
        return('D')
    elif(TotalMarks >= E_cutoff):
        return('E')
    else:
        return('U')

Grade = []
for i in range(20):
    Grade.append(RelativeGrading(TotalMarks[i]))
```

In [ ]:

```
frequency = {}
frequency['S'] = 0
frequency['A'] = 0
frequency['B'] = 0
frequency['C'] = 0
frequency['D'] = 0
frequency['E'] = 0
frequency['U'] = 0

for i in Grade:
    if i in Grades:
        frequency[str(i)] += 1
```

In [ ]:

```
frequency
```

Out[ ]:

```
{'S': 1, 'A': 2, 'B': 3, 'C': 4, 'D': 3, 'E': 4, 'U': 3}
```

In [ ]:

```
data = pd.DataFrame(frequency.items(), columns = ['Grade', 'Frequency'])
data
```

Out[ ]:

| | Grade | Frequency |
|---|---|---|
| **0** | S | 1 |
| **1** | A | 2 |
| **2** | B | 3 |
| **3** | C | 4 |
| **4** | D | 3 |
| **5** | E | 4 |
| **6** | U | 3 |