

Analysis and Systems of Big Data Practise Lab - 1

Sreepathy Jayanand
CED17IO38

Q1:

Given the following setup {Class,Tally score, Frequency}, develop an application that generates the table shown ; (you can populate the relevant data ; minimum data size :50 records). The table is only an illustration for a data of color scores, you are free to test the application over any data set with the application generating the tally and frequency scores.

Logic:

Given "n", to create tally, Create "||||\" - (n / 5) times... Then followed by "|" - (n % 5) times.

50 such colours and corresponding n values are read from the input file

Packages used:

Re - Extracting words from the input file (Regular expressions)

Tabulate - For creating the table.

Code snippets:

```
import re
from tabulate import tabulate
def calctal(x):
    count = int(x)
    t1 = "||||\"
    s = ""
    for i in range(count//5):
        s = s + t1
    for i in range(count % 5):
        s = s + "|"
    s = s + " "
    return s
f = open("input1.txt", "r")
s = f.read()
f.close()
res = re.findall(r'\w+', s)
print(res)
n = len(res)
a = []
b = []
for i in range(0, n, 2):
    a.append(res[i])
    b.append(res[i + 1])
print(a)
print(b)
n = n // 2
ans = []
for i in range(n):
    tal = calctal(b[i])
    temp = [a[i], tal, b[i]]
    ans.append(temp)
print(tabulate(ans, headers = ['Score', 'Tally', 'Frequency']))
```

Input / Output:

Colour1	1
Colour2	2
Colour3	3
Colour4	4
Colour5	5
Colour6	6
Colour7	7
Colour8	8
Colour9	9
Colour10	10
Colour11	11
Colour12	12
Colour13	13
Colour14	14
Colour15	15
Colour16	16
Colour17	17
Colour18	18
Colour19	19
Colour20	20
Colour21	21
Colour22	22
Colour23	23
Colour24	24
Colour25	25
Colour26	26
Colour27	27
Colour28	28
Colour29	29
Colour30	30
Colour31	31
Colour32	32
Colour33	33
Colour34	34
Colour35	35
Colour36	36
Colour37	37
Colour38	38
Colour39	39
Colour40	40
Colour41	41
Colour42	42
Colour43	43
Colour44	44
Colour45	45
Colour46	46
Colour47	47
Colour48	48
Colour49	49
Colour50	50

```

Lab1 -- -bash -- 111x52
Score  Tally  Frequency
-----
Colour1  | 1
Colour2  || 2
Colour3  ||| 3
Colour4  |||| 4
Colour5  ||||\ 5
Colour6  ||||\ | 6
Colour7  ||||\ || 7
Colour8  ||||\ ||| 8
Colour9  ||||\ |||| 9
Colour10 ||||\ ||||\ 10
Colour11 ||||\ ||||\ | 11
Colour12 ||||\ ||||\ || 12
Colour13 ||||\ ||||\ ||| 13
Colour14 ||||\ ||||\ |||| 14
Colour15 ||||\ ||||\ ||||\ 15
Colour16 ||||\ ||||\ ||||\ | 16
Colour17 ||||\ ||||\ ||||\ || 17
Colour18 ||||\ ||||\ ||||\ ||| 18
Colour19 ||||\ ||||\ ||||\ |||| 19
Colour20 ||||\ ||||\ ||||\ ||||\ 20
Colour21 ||||\ ||||\ ||||\ ||||\ | 21
Colour22 ||||\ ||||\ ||||\ ||||\ || 22
Colour23 ||||\ ||||\ ||||\ ||||\ ||| 23
Colour24 ||||\ ||||\ ||||\ ||||\ |||| 24
Colour25 ||||\ ||||\ ||||\ ||||\ ||||\ 25
Colour26 ||||\ ||||\ ||||\ ||||\ ||||\ | 26
Colour27 ||||\ ||||\ ||||\ ||||\ ||||\ || 27
Colour28 ||||\ ||||\ ||||\ ||||\ ||||\ ||| 28
Colour29 ||||\ ||||\ ||||\ ||||\ ||||\ |||| 29
Colour30 ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ 30
Colour31 ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ | 31
Colour32 ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ || 32
Colour33 ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||| 33
Colour34 ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ |||| 34
Colour35 ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ 35
Colour36 ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ | 36
Colour37 ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ || 37
Colour38 ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||| 38
Colour39 ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ |||| 39
Colour40 ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ 40
Colour41 ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ | 41
Colour42 ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ || 42
Colour43 ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||| 43
Colour44 ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ |||| 44
Colour45 ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ 45
Colour46 ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ | 46
Colour47 ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ || 47
Colour48 ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||| 48
Colour49 ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ |||| 49
Colour50 ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ ||||\ 50

```

Q2:

In a class of 18 students, assume marks distribution in an exam are as follows. Let the roll numbers start with CSE20D01 and all the odd roll numbers secure marks as follows: $25 + ((i+7) \% 10)$ and even roll numbers : $25 + ((i+8) \% 10)$. Develop an application that sets up the data and calculate the mean and median for the marks obtained using the platform support.

Logic:

Inside a for loop based on parity of loop variable the appropriate value is generated and stored. Using `statistics.mean()`, `statistics.median()` the mean and median are calculated.

Packages used:

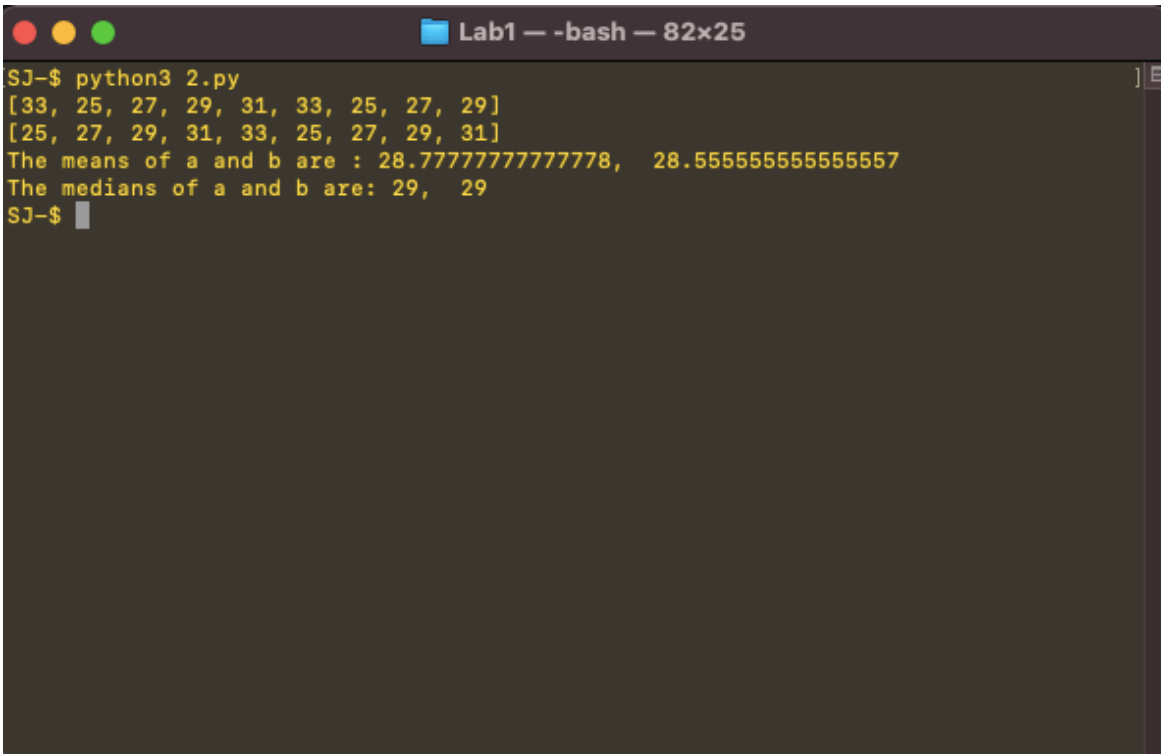
Statistics - For `mean()` and `median()`

Code Snippet:

```
import statistics
a = []
b = []
for i in range(1, 19):
    if i % 2 == 1:
        a.append(25 + ((i + 7) % 10))
    else:
        b.append(25 + ((i + 8) % 10))
print(a)
print(b)
print("The means of a and b are :", statistics.mean(a), "\b, ", statistics.mean(b))
print("The medians of a and b are:", statistics.median(a), "\b, ", statistics.median(b))
```

Input / Output:

The input is generated using the code.

A terminal window titled "Lab1 - -bash - 82x25" with a dark background and light green text. The window shows the execution of a Python script. The output consists of two lists of numbers, followed by the means and medians of these lists. The first list is [33, 25, 27, 29, 31, 33, 25, 27, 29] and the second is [25, 27, 29, 31, 33, 25, 27, 29, 31]. The means are calculated as 28.77777777777778 and 28.555555555555557. The medians are both 29.

```
SJ-$ python3 2.py
[33, 25, 27, 29, 31, 33, 25, 27, 29]
[25, 27, 29, 31, 33, 25, 27, 29, 31]
The means of a and b are : 28.77777777777778,  28.555555555555557
The medians of a and b are: 29,  29
SJ-$
```

Q3:

For a sample space of 20 elements, the values are fitted to the line $Y=2X+3$, $X>5$. Develop an application that sets up the data and computes the standard deviation of this sample space. (use random number generator supported in your development platform to generate values of X).

Logic:

Inside a for loop based 20 random integers are generated (x) and $2 * x + 3$ is calculated and stored. Using `statistics.stdev()` the standard deviation is calculated.

Packages used:

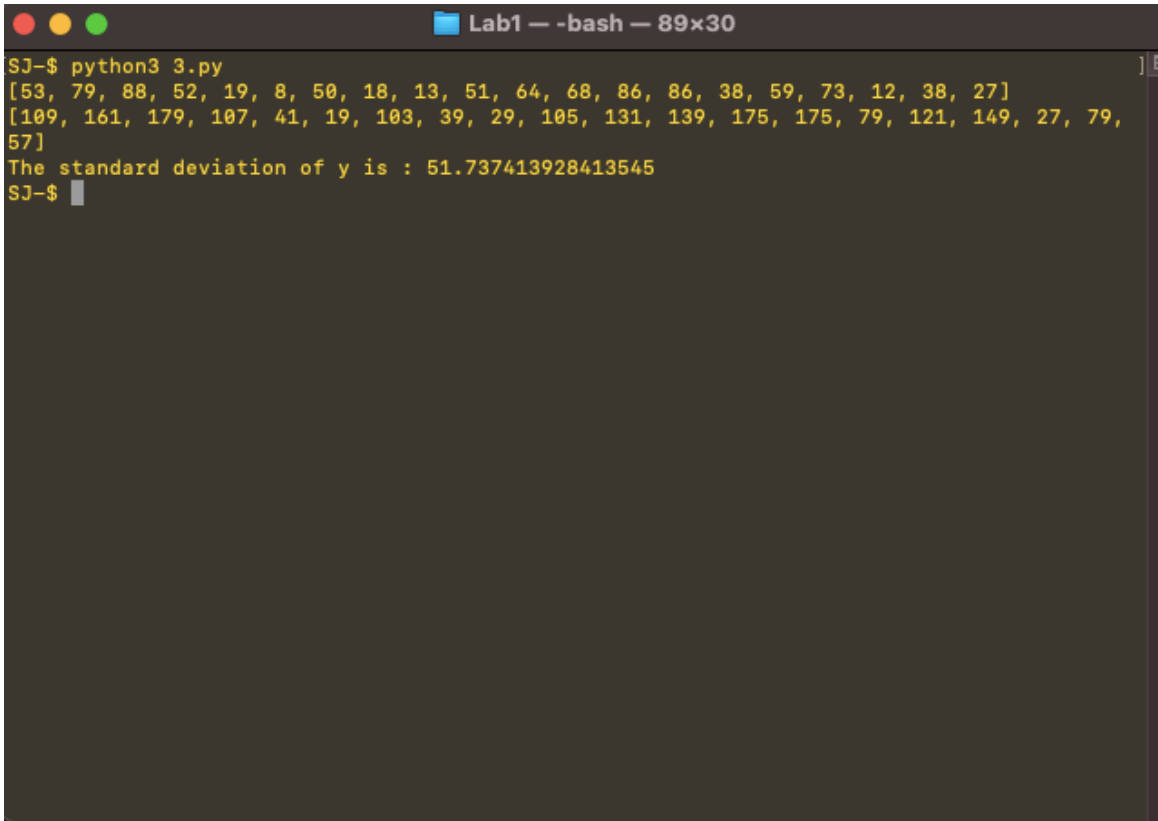
Statistics - For standard deviation (`stdev()`)

Code Snippet:

```
import random
import statistics
y = []
x = []
for i in range(20):
    p = random.randint(5, 100)
    x.append(p)
    y.append(2 * p + 3)
print(x)
print(y)
print("The standard deviation of y is :", statistics.stdev(y))
```

Input / Output:

Input is generated using the code.

A terminal window titled "Lab1 — -bash — 89x30" with standard macOS window controls (red, yellow, green buttons). The terminal shows the execution of a Python script. The prompt is "SJ-\$". The command "python3 3.py" is entered. The output consists of two lines of numbers in brackets, followed by a line stating the standard deviation of y. The prompt "SJ-\$" is shown again with a cursor.

```
SJ-$ python3 3.py
[53, 79, 88, 52, 19, 8, 50, 18, 13, 51, 64, 68, 86, 86, 38, 59, 73, 12, 38, 27]
[109, 161, 179, 107, 41, 19, 103, 39, 29, 105, 131, 139, 175, 175, 79, 121, 149, 27, 79,
57]
The standard deviation of y is : 51.737413928413545
SJ-$
```

Q4:

For a given data of heights of a class, the heights of 15 students are recorded as 167.65, 167, 172, 175, 165, 167, 168, 167, 167.3, 170, 167.5, 170, 167, 169, and 172. Develop an application that computes; explore if there are any packages supported in your platform that depicts these measures / their calculation of central tendency in a visual form for ease of understanding.

- Mean height of the student
- Median and Mode of the sample space
- Standard deviation
- Measure of skewness. $[(\text{Mean}-\text{Mode})/\text{standard deviation}]$

Logic:

Using the statistics.mean(), median(), mode(), stdev() all the values are calculated.

Then skewness is calculated as $[(\text{Mean}-\text{Mode})/\text{standard deviation}]$

Packages used:

Statistics - For mean(), median(), stdev()

Matplotlib - For visualisation

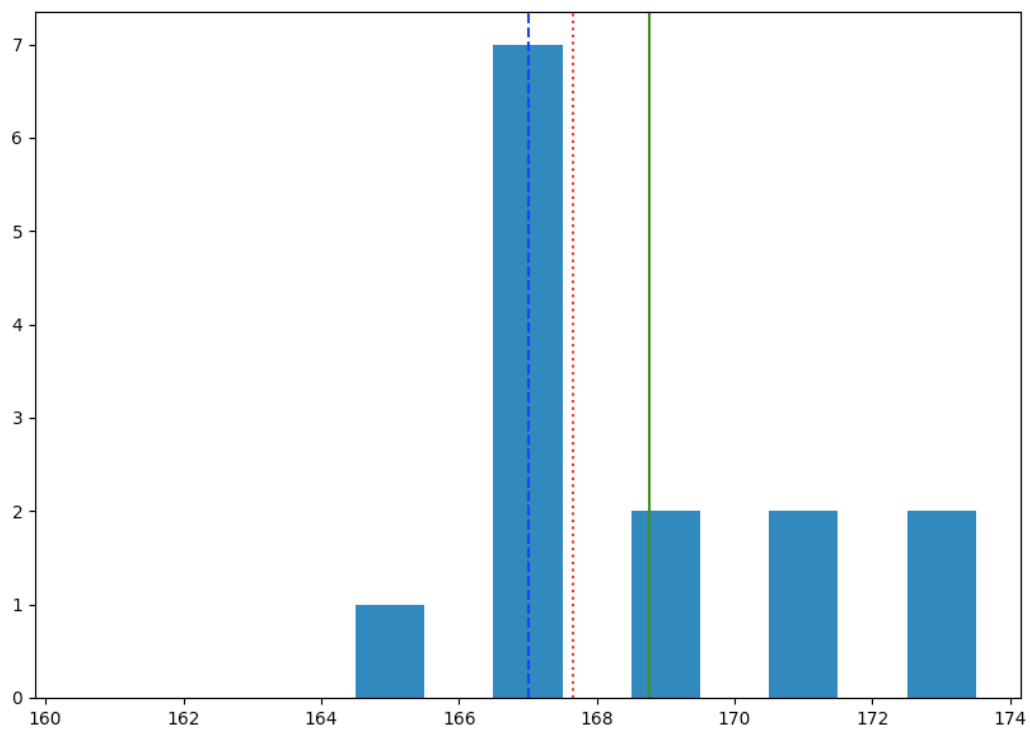
Code Snippet:

```
import statistics
from matplotlib import pyplot as plt
x = [167.65, 167, 172, 175, 165, 167, 168, 167, 167.3, 170, 167.5, 170, 167, 169, 172]
x_mean = statistics.mean(x)
x_median = statistics.median(x)
x_mode = statistics.mode(x)
x_stdev = statistics.stdev(x)
print("The mean is", x_mean)
print("The median and mode is", x_median, "\b, ", x_mode)
print("The standard deviation is", x_stdev)
print("The measure of skewness is", (x_mean - x_mode) / x_stdev)
fig, ax = plt.subplots(figsize=(10, 7))
ax.hist(x, bins = [160, 162, 164, 166, 168, 170, 172, 174], histtype = 'bar', rwidth = .5)
plt.axvline(x_median, color = 'r', linestyle = ':')
plt.axvline(x_mode, color = 'b', linestyle = '--')
plt.axvline(x_mean, color = 'g', linestyle = '-')
plt.show()
```


Input / Output:

Input is given as a part of the code

```
Lab1 — -bash — 82x25
SJ-$ python3 4.py
The mean is 168.76333333333332
The median and mode is 167.65, 167
The standard deviation is 2.606617037647145
The measure of skewness is 0.6764834679838465
SJ-$
```



Q5:

In Analytics and Systems of Bigdata course, for a class of 100 students, around 31 students secured 'S' grade, 29 secured 'B' grade, 25 'C' grades, and rest of them secured 'D' grades. If the range of each grade is 15 marks. (S for 85 to 100 marks, A for 70 to 85 ...). Develop an application that represents the above data : using Pie and Bar graphs.

Logic:

Using the library matplotlib the pie chart and bar graph is drawn.

Packages used:

Matplotlib - For Pie chart and bar graph

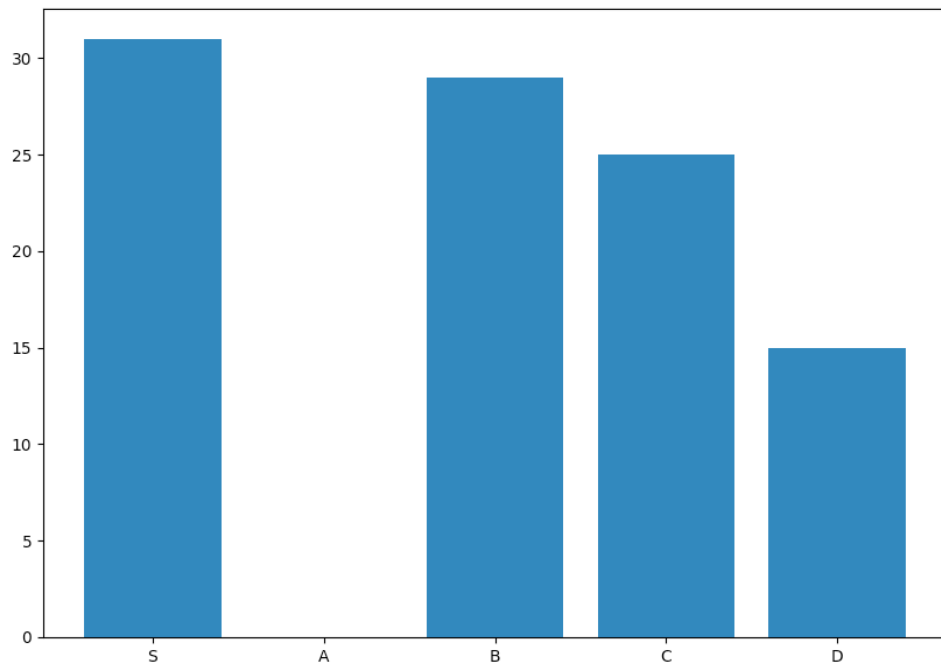
Code Snippet:

```
from matplotlib import pyplot as plt
x = ['S', 'A', 'B', 'C', 'D']
y = [31, 0, 29, 25, 15]
fig = plt.figure(figsize = (10, 7))
# plt.pie(y, labels = x)
plt.bar(x, y)
plt.show()
```

Input / Output:

The input is a part of the code.

Figure 1



Q6:

On a given day (average basis), a student is observed to spend 33% of time in studying, 30% in sleeping, 18% in playing, 5% for hobby activities, and rest for spending with friends and family. Plot a pie chart showing his daily activities.

Logic:

Using the library matplotlib the pie chart and bar graph is drawn.

Packages used:

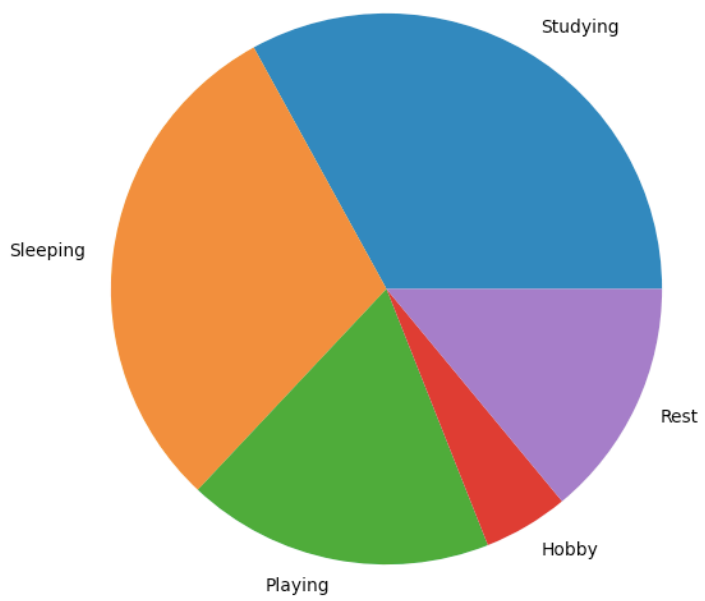
Matplotlib - For Pie chart and bar graph

Code Snippet:

```
from matplotlib import pyplot as plt
x = ['Studying', 'Sleeping', 'Playing', 'Hobby', 'Rest']
y = [33, 30, 18, 5, 14]
fig = plt.figure(figsize = (10, 7))
plt.pie(y, labels = x)
# plt.bar(x, y)
plt.show()
```

Input / Output:

The input is a part of the code.



Q7:

Develop an application (absolute grader) that accepts marks scored by 20 students in ASBD course (as a split up of three : Mid Sem (30), End Sem(50) and Assignments(20). Compute the total and use it to grade the students following absolute grading : ≥ 90 – S ; ≥ 80 – A and so on till D. Compute the Class average for total marks in the course and 50% of class average would be fixed as the cut off for E. Generate a frequency table for the grades as well (Table displaying the grades and counts of them).

Logic:

The input is read as a txt file and stores. Individual entries are separated using the regular expressions library. The sum of marks for each student based on their individual scores are calculated. Then using their total marks the average is calculated. Based on the conditions for the grades if/elif conditions are made and the count of grades are calculated by incrementing a counter for each grades.

Packages used:

Re - Regular expressions for tokenizing the input.

Statistics - To calculate the mean()

Code Snippet:

After reading the input from the file,

```

entries = []
grades = dict()
grades['S'] = 0
grades['A'] = 0
grades['B'] = 0
grades['C'] = 0
grades['D'] = 0
grades['E'] = 0
grades['F'] = 0
for i in range(0, len(res) - 3, 4):
    entry_1 = int(res[i])
    entry_2 = int(res[i + 1])
    entry_3 = int(res[i + 2])
    entry_4 = int(res[i + 3])
    entries.append([entry_1, entry_2, entry_3, entry_4, entry_2 + entry_3 + entry_4])
# print(entries)
average = 0
n = len(res) // 4
for i in range(n):
    average = average + entries[i][4]
average = average / 20
print("The average is", average)
final_grades = [None] * 20
for i in range(n):
    if (entries[i][4] >= 90):
        final_grades[i] = 'S'
        grades['S'] = grades['S'] + 1
    elif (entries[i][4] >= 80):
        final_grades[i] = 'A'
        grades['A'] = grades['A'] + 1
    elif (entries[i][4] >= 70):
        final_grades[i] = 'B'
        grades['B'] = grades['B'] + 1
    elif (entries[i][4] >= 60):
        final_grades[i] = 'C'
        grades['C'] = grades['C'] + 1
    elif (entries[i][4] >= 50):
        final_grades[i] = 'D'
        grades['D'] = grades['D'] + 1
    elif (entries[i][4] >= average / 2):
        final_grades[i] = 'E'
        grades['E'] = grades['E'] + 1
    else:
        final_grades[i] = 'F'
        grades['F'] = grades['F'] + 1
    print("Student" + str(i) + " : Marks = ", entries[i][1], ",", entries[i][2], ",", entries[i][3], ",", "Total : ", entries[i][4])
print("Frequency distribution", grades)

```

Input / Output:

Input:

```

1 2 13 4
2 29 36 5
3 26 5 20
4 0 10 14
5 24 49 5
6 23 50 17
7 26 0 15
8 12 31 0
9 16 11 8
10 24 7 11
11 1 22 11
12 26 17 12
13 10 44 17
14 13 18 16
15 23 13 12
16 12 21 13
17 24 23 17
18 20 25 10
19 17 10 12
20 9 43 0

```

Output:

```
Lab1 — -bash — 89x30
[SJ-$ python3 7.py
The average is 50.2
Student0 : Marks = 2 , 13 , 4 , Total : 19 Grades = F
Student1 : Marks = 29 , 36 , 5 , Total : 70 Grades = B
Student2 : Marks = 26 , 5 , 20 , Total : 51 Grades = D
Student3 : Marks = 0 , 10 , 14 , Total : 24 Grades = F
Student4 : Marks = 24 , 49 , 5 , Total : 78 Grades = B
Student5 : Marks = 23 , 50 , 17 , Total : 90 Grades = S
Student6 : Marks = 26 , 0 , 15 , Total : 41 Grades = E
Student7 : Marks = 12 , 31 , 0 , Total : 43 Grades = E
Student8 : Marks = 16 , 11 , 8 , Total : 35 Grades = E
Student9 : Marks = 24 , 7 , 11 , Total : 42 Grades = E
Student10 : Marks = 1 , 22 , 11 , Total : 34 Grades = E
Student11 : Marks = 26 , 17 , 12 , Total : 55 Grades = D
Student12 : Marks = 10 , 44 , 17 , Total : 71 Grades = B
Student13 : Marks = 13 , 18 , 16 , Total : 47 Grades = E
Student14 : Marks = 23 , 13 , 12 , Total : 48 Grades = E
Student15 : Marks = 12 , 21 , 13 , Total : 46 Grades = E
Student16 : Marks = 24 , 23 , 17 , Total : 64 Grades = C
Student17 : Marks = 20 , 25 , 10 , Total : 55 Grades = D
Student18 : Marks = 17 , 10 , 12 , Total : 39 Grades = E
Student19 : Marks = 9 , 43 , 0 , Total : 52 Grades = D
Frequency distribution {'S': 1, 'A': 0, 'B': 3, 'C': 1, 'D': 4, 'E': 9, 'F': 2}
SJ-$
```


Q8:

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.

Logic:

The input is read as a txt file and stores. Individual entries are separated using the regular expressions library. The sum of marks for each student based on their individual scores are calculated. Then using their total marks the average is calculated. The standard deviation is also calculated using the statistics library.

Mean - μ , Standard Deviation - s

S grade - $\geq \mu + 2s$

A grade - $\geq \mu + s$

B grade - $\geq \mu +$

C grade - $\geq \mu - s$

D grade - $\geq \mu - 2s$

E grade - $\geq \mu / 2$

F grade - Else

Packages used:

Statistics - For mean(), median() and stdev()

Re - To tokenize the input

Code Snippet:

```
for i in range(0, len(res) - 3, 4):
    entry_1 = int(res[i])
    entry_2 = int(res[i + 1])
    entry_3 = int(res[i + 2])
    entry_4 = int(res[i + 3])
    entries.append([entry_1, entry_2, entry_3, entry_4, entry_2 + entry_3 + entry_4])
# print(entries)
average = 0
n = len(res) // 4
marks = []
for i in range(n):
    average = average + entries[i][4]
    marks.append(entries[i][4])
mean_marks = statistics.mean(marks)
stddev = statistics.stdev(marks)
average = average / 20
print("The average is", mean_marks)
print("The standard deviation is", stddev)
final_grades = [None] * 20
for i in range(n):
    if (entries[i][4] >= mean_marks + 2 * stddev):
        final_grades[i] = 'S'
        grades['S'] = grades['S'] + 1
    elif (entries[i][4] >= mean_marks + stddev):
        final_grades[i] = 'A'
        grades['A'] = grades['A'] + 1
    elif (entries[i][4] >= mean_marks):
        final_grades[i] = 'B'
        grades['B'] = grades['B'] + 1
    elif (entries[i][4] >= mean_marks - stddev):
        final_grades[i] = 'C'
        grades['C'] = grades['C'] + 1
    elif (entries[i][4] >= mean_marks - 2 * stddev):
        final_grades[i] = 'D'
        grades['D'] = grades['D'] + 1
    elif (entries[i][4] >= average / 2):
        final_grades[i] = 'E'
        grades['E'] = grades['E'] + 1
    else:
        final_grades[i] = 'F'
        grades['F'] = grades['F'] + 1
    print("Student" + str(i) + " : Marks = ", entries[i][1], ",", entries[i][2], ",", entries[i][3], ",", "Total : ", entries[i][4])
print("Frequency distribution", grades)
```

Input / Output:

Input:

```
1 2 13 4
2 29 36 5
3 26 5 20
4 0 10 14
5 24 49 5
6 23 50 17
7 26 0 15
8 12 31 0
9 16 11 8
10 24 7 11
11 1 22 11
12 26 17 12
13 10 44 17
14 13 18 16
15 23 13 12
16 12 21 13
17 24 23 17
18 20 25 10
19 17 10 12
20 9 43 0
```

Output:

```
SJ-$ python3 8.py
The average is 50.2
The standard deviation is 17.677222930265824
Student0 : Marks = 2 , 13 , 4 , Total : 19 Grades = D
Student1 : Marks = 29 , 36 , 5 , Total : 70 Grades = A
Student2 : Marks = 26 , 5 , 20 , Total : 51 Grades = B
Student3 : Marks = 0 , 10 , 14 , Total : 24 Grades = D
Student4 : Marks = 24 , 49 , 5 , Total : 78 Grades = A
Student5 : Marks = 23 , 50 , 17 , Total : 90 Grades = S
Student6 : Marks = 26 , 0 , 15 , Total : 41 Grades = C
Student7 : Marks = 12 , 31 , 0 , Total : 43 Grades = C
Student8 : Marks = 16 , 11 , 8 , Total : 35 Grades = C
Student9 : Marks = 24 , 7 , 11 , Total : 42 Grades = C
Student10 : Marks = 1 , 22 , 11 , Total : 34 Grades = C
Student11 : Marks = 26 , 17 , 12 , Total : 55 Grades = B
Student12 : Marks = 10 , 44 , 17 , Total : 71 Grades = A
Student13 : Marks = 13 , 18 , 16 , Total : 47 Grades = C
Student14 : Marks = 23 , 13 , 12 , Total : 48 Grades = C
Student15 : Marks = 12 , 21 , 13 , Total : 46 Grades = C
Student16 : Marks = 24 , 23 , 17 , Total : 64 Grades = B
Student17 : Marks = 20 , 25 , 10 , Total : 55 Grades = B
Student18 : Marks = 17 , 10 , 12 , Total : 39 Grades = C
Student19 : Marks = 9 , 43 , 0 , Total : 52 Grades = B
Frequency distribution {'S': 1, 'A': 3, 'B': 5, 'C': 9, 'D': 2, 'E': 0, 'F': 0}
SJ-$
```

Q9:

Consider the following sample of weights for 45 individuals: 79 71 89 57 76 64 82 82 67 80 81 65 73 79 79 60 58 83 74 68 78 80 78 81 76 65 70 76 58 82 59 73 72 79 87 63 74 90 69 70 83 76 61 66 71 60 57 81 57 65 81 78 77 81 81 63 71 66 56 62 75 64 74 74 70 71 56 69 63 72 81 54 72 91 92. For the above data generates histograms and depict them using packages in your platform. Explore the different types of histograms available and test drive the types supported in your platform.

Logic:

Using matplotlib, the different types of histograms, the following data is visualised.

Packages used:

Statistics - For mean() and median()

Matplotlib - For different types of histograms

Code Snippet:

```
import statistics
from matplotlib import pyplot as plt
import re
f = open("input9.txt", "r")
s = f.read()
f.close()
res = re.findall(r'\w+', s)
print(res)
x = []
for i in range(len(res)):
    x.append(int(res[i]))
print(x)
fig, ax = plt.subplots(figsize=(10, 7))
ax.hist(x, bins = [45, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95], histtype = 'stepfilled', rwidth = 0.5)
plt.show()
```

Input / Output:

Input:

```
79 71 89 57 76 64 82 82 67 80 81 65 73 79 79 60 58 83 74 68 78 80 78 81 76
65 70 76 58 82 59 73 72 79 87 63 74 90 69 70 83 76 61 66 71 60 57
81 57 65 81 78 77 81 81 63 71 66 56 62 75 64 74 74 70 71 56 69 63 72 81 54 72 91 92
```

Output:

