```
def mean():
    sepalLen = generateList(dataSet, "SepalLengthCm")
    sepalWid = generateList(dataSet, "SepalWidthCm")
    petalLen = generateList(dataSet, "PetalLengthCm")
    petalWid = generateList(dataSet, "PetalWidthCm")

    sepalLenMean = 0
    sepalWidMean = 0
    petalLenMean = 0
    petalWidMean = 0

    for i in sepalLen:
        sepalLenMean += i
    sepalLenMean = sepalLenMean / len(sepalLen)

    for i in sepalWid:
        sepalWidMean += i
    sepalWidMean = sepalWidMean / len(sepalWid)

    for i in petalLen:
        petalLenMean += i
    petalLenMean = petalLenMean / len(petalLen)

    for i in petalWid:
        petalWidMean += i
    petalWidMean = petalWidMean / len(petalWid)
    print(f'SepalLen Mean: {sepalLenMean} sepalWid mean: {sepalWidMean}\npetalLen mean: {petalLenMean} petalWid mean:
    return sepalLenMean, sepalWidMean, petalLenMean, petalWidMean
```

3. Median

```
def median():
    sepalLen = sorted(generateList(dataSet, "SepalLengthCm"))
    sepalWid = sorted(generateList(dataSet, "SepalWidthCm"))
    petalLen = sorted(generateList(dataSet, "PetalLengthCm"))
    petalWid = sorted(generateList(dataSet, "PetalWidthCm"))

    sepalLenMeadian = 0
    sepalWidMeadian = 0
    petalLenMeadian = 0
    petalWidMeadian = 0

    if len(sepalLen) % 2 == 0:
        sepalLenMeadian = sepalLen[(len(sepalLen) - 1) // 2]
    else:
        sepalLenMeadian = sepalLen[(len(sepalLen) - 1) // 2] + sepalLen[((len(sepalLen) - 1) // 2) + 1] / 2.0

    if len(sepalWid) % 2 == 0:
        sepalWidMeadian = sepalWid[(len(sepalWid) - 1) // 2]
    else:
        sepalWidMeadian = sepalWid[(len(sepalWid) - 1) // 2] + sepalWid[((len(sepalWid) - 1) // 2) + 1] / 2.0

    if len(petalLen) % 2 == 0:
        petalLenMeadian = petalLen[(len(petalLen) - 1) // 2]
    else:
        petalLenMeadian = petalLen[(len(petalLen) - 1) // 2] + petalWid[((len(petalLen) - 1) // 2) + 1] / 2.0

    if len(petalWid) % 2 == 0:
        petalWidMeadian = petalWid[(len(petalWid) - 1) // 2]
    else:
        petalWidMeadian = petalWid[(len(petalWid) - 1) // 2] + petalWid[((len(petalWid) - 1) // 2) + 1] / 2.0

    return f'Sepal Length Meadian: {sepalLenMeadian} Sepal Width meadian: {sepalWidMeadian}\nPetal Length meadian: {petalLenMeadian} Peta
```

```
Sepal Length Meadian: 5.8 Sepal Width meadian: 3.0
Petal Length meadian: 4.3 Petal Width meadian: 1.3
```

3. Mode

```
def abstractMax(_list):
    return max(set(_list), key=_list.count)

def mode():
    sepalLen = generateList(dataSet, "SepalLengthCm")
    sepalWid = generateList(dataSet, "SepalWidthCm")
    petalLen = generateList(dataSet, "PetalLengthCm")
    petalWid = generateList(dataSet, "PetalWidthCm")

    return f'Sepal Length Mode: {abstractMax(sepalLen)} Sepal Width Mode: {abstractMax(sepalWid)}\nPe
```

```
Sepal Length Mode: 5.0 Sepal Width Mode: 3.0
Petal Length: 1.5 Petal Width: 0.2
```