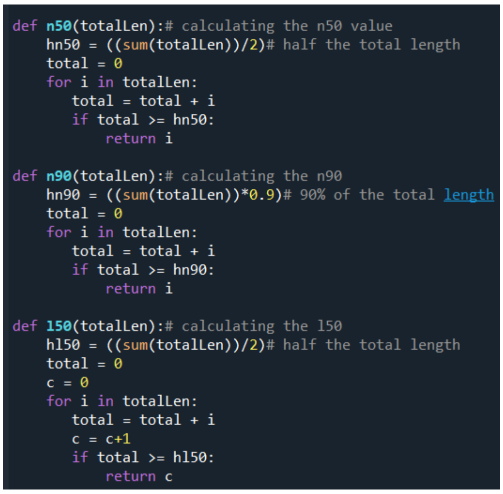
CS31420 Computational Bioinformatics tob31 assignment Part 1

Calculating the metrics N50 N90 L50 GC percentage

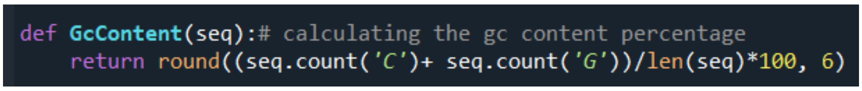


```
def n50(totalLen):# calculating the n50 value
    hn50 = ((sum(totalLen))/2)# half the total length
    total = 0
    for i in totalLen:
       total = total + i
       if total >= hn50:
           return i

def n90(totalLen):# calculating the n90
    hn90 = ((sum(totalLen))*0.9)# 90% of the total length
    total = 0
    for i in totalLen:
       total = total + i
       if total >= hn90:
           return i

def l50(totalLen):# calculating the l50
    hl50 = ((sum(totalLen))/2)# half the total length
    total = 0
    c = 0
    for i in totalLen:
       total = total + i
       c = c+1
       if total >= hl50:
           return c
```

to calculate n50 the length of the sequence is halved and from the sorted list largest to smallest the last contig length value that is added on to make up half /or more than half the length of the sequence is displayed.

The same is down for n90 but instead of half you will do 90% of the full sequence length and display the contig length that makes that.

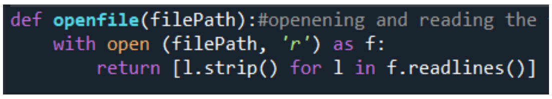
The same is done for L50 however instead of displaying the contig length you display the count of how many contigs it is along from the start of the list.



```
def GcContent(seq):# calculating the gc content percentage
    return round((seq.count('C')+ seq.count('G'))/len(seq)*100, 6)
```

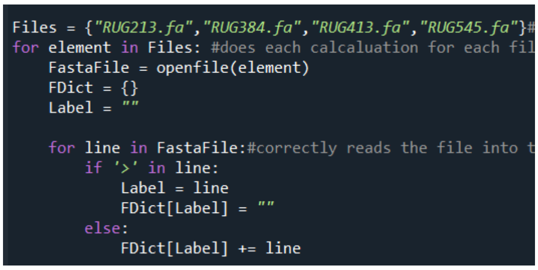
The GC content is calculated by counted the g & c from the sequence dividing them by the length of the sequence and multiplying by 100 to calculate the percentage of the sequence that is g & c.

Opening, Reading and Displaying Multiple Fasta files and their calculated information.



```
def openfile(filePath):#openening and reading the
    with open (filePath, 'r') as f:
        return [l.strip() for l in f.readlines()]
```

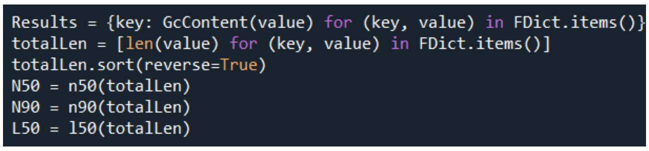
This function is used to open the file provided as read and strips any leading or ending spaces



```
Files = {"RUG213.fa","RUG384.fa","RUG413.fa","RUG545.fa"}#
for element in Files: #does each calcaluation for each fil
    FastaFile = openfile(element)
    FDict = {}
    Label = ""

    for line in FastaFile:#correctly reads the file into t
        if '>' in line:
            Label = line
            FDict[Label] = ""
        else:
            FDict[Label] += line
```

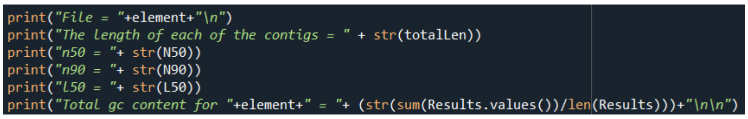
The files names provided are stored in a list that a for loop iterated through to display the correct information from each genome file separately.

Another for loop then correctly stores the data from the files into a python dictionary. With a key and value from the scaffolds. The key starts with '>'



```
Results = {key: GcContent(value) for (key, value) in FDict.items()}
totalLen = [len(value) for (key, value) in FDict.items()]
totalLen.sort(reverse=True)
N50 = n50(totalLen)
N90 = n90(totalLen)
L50 = l50(totalLen)
```

The functions to calculate the desired values are called here and put into lists /or dictionaries. The list of contigs lengths is also sorted by reverse order.



```
print("File = "+element+"\n")
print("The length of each of the contigs = " + str(totalLen))
print("n50 = "+ str(N50))
print("n90 = "+ str(N90))
print("L50 = "+ str(L50))
print("Total gc content for "+element+" = "+ (str(sum(Results.values())/len(Results)))+"\n\n")
```

Here is where the information is displayed to the screen.