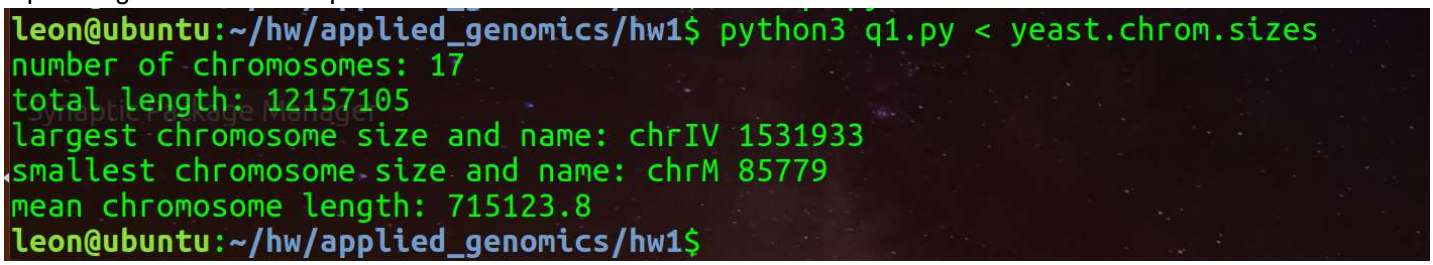
Hw1

Question 1: Chromosome structures:

Solutions:

After reading the gene file we want to process, we could store the data in a dictionary with key contains the name of chromosomes and value contains the length of sequence. Then we could use simple commands (e.g. max(), min(), len()) to process data for Q1. After inputting the genes we want to process, we could get the ansewers as shown below.
Input the genes we want to process:



Answers:

	TAIR10	zm4	ecoli	dm6	hg38	rice	ce10	yeast
Total genome size	119146348	2106338117	4639211	137547960	3088269832	373245519	100286070	12157105
Number of chromosomes	5	10	1	7	24	12	7	17
Largest chromosome size and name	Chr1: 30427671	1: 307041717	Ecoli: 4639211	chr3R: 32079331	chr1: 248956422	Chr1: 43270923	chrV: 20924149	chrIV: 1531933
Smallest chromosome size and name	Chr4: 18585056	10: 150982314	Ecoli: 4639211	chr4: 1348131	chr21: 46709983	Chr9: 23012720	chrM: 13794	chrM: 85779
Mean chromosome length	23829269.6	210633811.7	4639211.0	19649708.6	128677909.7	31103793.2	14326581.4	715123.8

Codes used are shown here:

```
#!/usr/bin/env python3
import sys
f = sys.stdin
line = f.readline()
dic = {}
while line != '':
    line = line.strip().rstrip('\n').split()
    dic[line[0]]=int(line[1])
     ###int is really really really important!!!!
    line = f.readline()
f.close()
print('number of chromosomes:',len(dic))

n = 0
for i in dic:
    n += int(dic[i])
print('total length:',n)

b = max(dic.values())   # largest size
c = list(dic.keys())[list(dic.values()).index(b)]   #coresponding name
print('largest chromosome size and name: %s %s'%(c,b))

b = min(dic.values())   # smallest size
c = list(dic.keys())[list(dic.values()).index(b)]       #coresponding name
print('smallest chromosome size and name: %s %s'%(c,b))

print('mean chromosome length:',format(n/len(dic),'.1f'))
```

Question 2: Sequence content

All the codes used are shown at the end of answers

Answers:

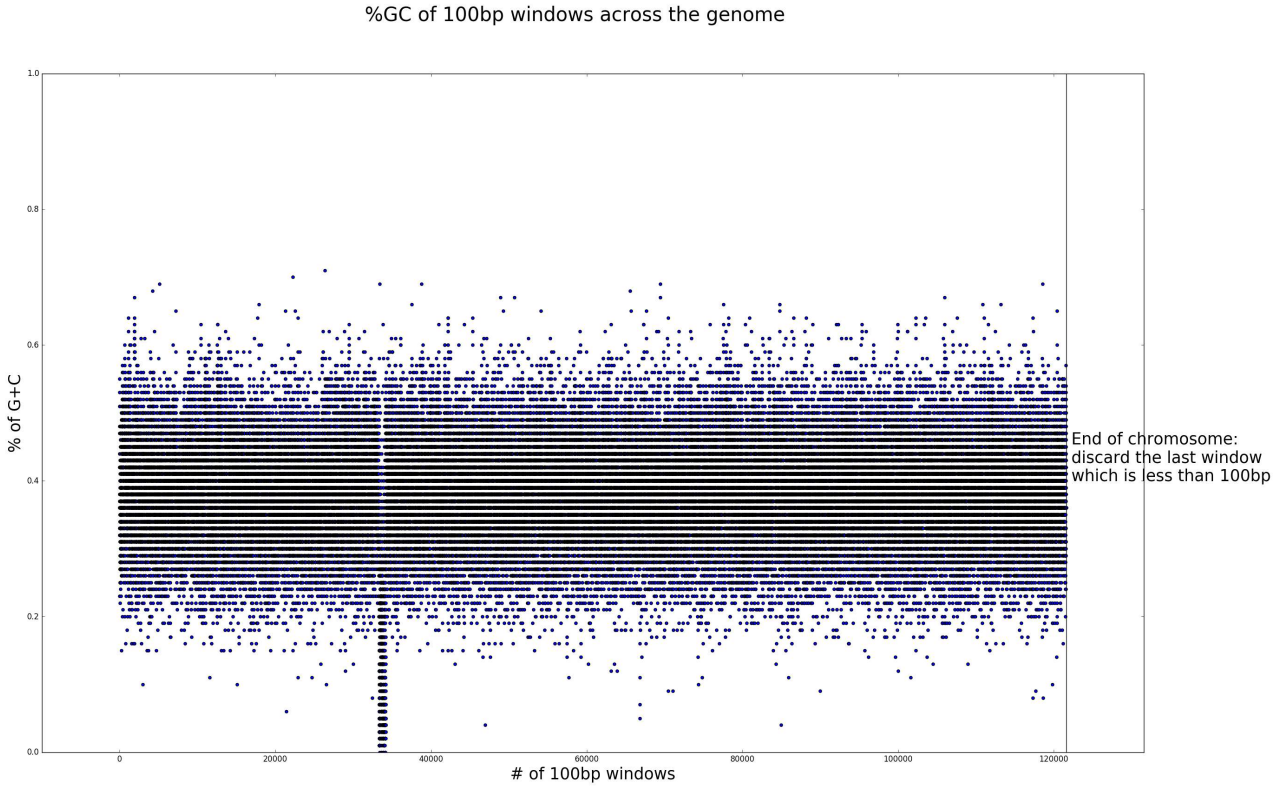
Question 2.1.

Silimar solutions in Q1, count the number of As, Cs, Gs, Ts in the entire genome using str.count()

```
leon@ubuntu:~/hw/applied_genomics/hw1$ python3 q2.py < yeast.fa
numeber of "A"s: 3766349
numeber of "T"s: 3753080
numeber of "G"s: 2317100
numeber of "C"s: 2320576
```

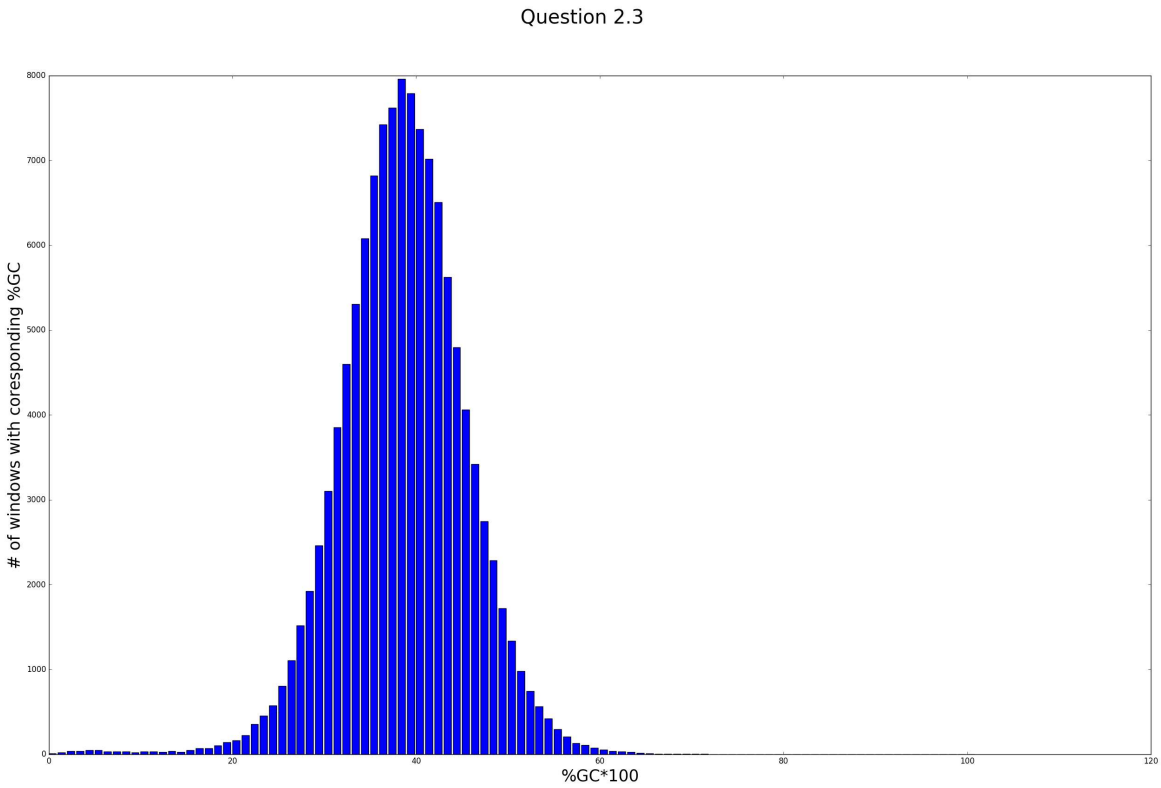
Question 2.2.

Using a dictionary whith key equals # of window, value equals %GC and draw this distionary:



Question 2.3.

Count the number of multiple values responding to one key and draw the bar:



Question 2.4.

From the results shown in Q2.2, we could expect windows(100bp) arround #32000, #33000 would perform poorly.

Codes used are shown here:

```
#!/usr/bin/env python3
########################################Question 2.1#########################################
import sys, matplotlib.pyplot as plt
f = sys.stdin
line = sys.stdin.readline() #skip header
line = sys.stdin.readline()
seq = ''
while line != '':
    seq += line.strip().rstrip('\n')
    line = f.readline()
f.close()
print('numeber of "'"A"'"s:', seq.count('A'))
print('numeber of "'"T"'"s:', seq.count('T'))
print('numeber of "'"G"'"s:', seq.count('G'))
print('numeber of "'"C"'"s:', seq.count('C'))
########################################Question 2.2#############################################
n = len(seq)//100    #number of bins, discard the last window
dic ={}
for i in range(n):
    b = ''
    b += seq[100*i:100*(i+1)] #divide the sequence by a window of 100
    for x in b:     #for each bin, count #GC
        gc = ''
        gc += str(b.count('G')+b.count('C'))
    dic[i+1]= int(gc)/100   # get a dictionary: key is the number of   \window, value is %GC
lists = sorted(dic.items())
x,y = zip(*lists)
plt.scatter(x,y)
plt.suptitle('%GC of 100bp windows across the genome',fontsize = 30)
plt.xlabel('# of 100bp windows',fontsize = 25)
plt.ylabel('% of G+C',fontsize = 25)
plt.xlim(0-10000,max(x)+10000)
plt.ylim(0,1)
plt.vlines(max(x),0,1)
plt.annotate(' End of chromosome:\n discard the last window \n which is less than 100bp',(max
(x),0.4),fontsize = 25)
plt.show()
#############################################Q2.3#####################################
for i in range(n):
    b = ''
    b += seq[100*i:100*(i+1)]
    for x in b:
        gc = ''
        gc += str(b.count('G')+b.count('C'))
    dic[i+1]= int(gc)
dic1={}
for n in range(101):
    dic1[n]=sum(value == n for value in dic.values())
#print(dic1)
plt.suptitle('Question 2.3',fontsize = 30)
lists = sorted(dic1.items())
x,y = zip(*lists)
plt.bar(x,y)
```

```
plt.xlabel('%GC*100',fontsize = 25)
plt.ylabel('# of windows with coresponding %GC',fontsize = 25)
plt.show()
```