BIG DATA ANALYTICS
PROJECT
HADOOP-SPARK-BIOPYTHON

Dataset: GRCh38_latest_genomic.fna
Format: FASTA
Details: Reference Genomic Sequencing of Human being chromosome 1 which contains the longest sequence of human DNA.
Size: 3.11 GB

Downloaded Github code from Big Data Europe.
File name: "docker-hadoop-spark-workbench-master"

Docker-compose file has been taken from "Big Data Europe" and a "jupypter/pyspark-notebook" container has been added by me in that docker-compose file for working in jupyter notebook.

Services include:

1. Hadoop Name Node
2. Hadoop Data Node
3. Spark Master
4. Spark Worker
5. Spark Notebook
6. Jupyter/pyspark-notebook
7. Hue-hdfs file browser

Libraries Used:

- pip install Bio
- import Bio
- from Bio import SeqIO
- import pyspark
- import pandas as pd
- import numpy as np
- import pyspark.pandas as ps
- from pyspark.sql import SparkSession
- from pyspark.sql.dataframe import DataFrame
- pip install matplotlib
- import pandas as pd
- from pyspark.sql import SQLContext, DataFrameWriter
- from pyspark.sql.types import *
- from pyspark import SparkContext
- from Bio.SeqUtils import molecular_weight

- from Bio.SeqUtils import GC
- import matplotlib.pyplot as plt
- from collections import Counter

Not all of these libraries were actually used. Some were tested and could not be used on my data set.

```
C:\Namra\IBA-Masters\BigDataAnalytics\Project\docker-hadoop-spark-workbench-master>docker-compose -f docker-compose.yml up -d
Creating network "docker-hadoop-spark-workbench-master_default" with the default driver
Pulling namenode (bde2020/hadoop-namenode:1.1.0-hadoop2.8-java8)...
1.1.0-hadoop2.8-java8: Pulling from bde2020/hadoop-namenode
6d827a3ef358: Pull complete
2726297beaf1: Pull complete
7d27bd3d7fec: Pull complete
e61641c845ed: Pull complete
cce4cca5b76b: Pull complete
6826227500b0: Pull complete
c03b117ffd91: Pull complete
821a1547b435: Pull complete
62c556046302: Pull complete
7356cfd2097b: Pull complete
1278a4edba62: Pull complete
90e77cddaa38: Pull complete
33f61e5733ff: Pull complete
ed6bab5146cb: Pull complete
dc277f34c12c: Pull complete
9baa02979410: Pull complete
78e8b8819e1b: Pull complete
08d1331fdbf7: Pull complete
97ad76e49975: Pull complete
7ab15f3d404d: Pull complete
281c3d016950: Pull complete
Digest: sha256:c0714fd74f589e44c1b9d9c800cb9a1379186fcbb65d31aedf3b80cd8a4db285
Status: Downloaded newer image for bde2020/hadoop-namenode:1.1.0-hadoop2.8-java8
Pulling datanode (bde2020/hadoop-datanode:1.1.0-hadoop2.8-java8)...
1.1.0-hadoop2.8-java8: Pulling from bde2020/hadoop-datanode
6d827a3ef358: Already exists
2726297beaf1: Already exists
7d27bd3d7fec: Already exists
e61641c845ed: Already exists
Digest: sha256:e6d0827f4e3c32d5cb10584963f3aafb5fd8ba4588a7c01e9bd75c834449d850
Status: Downloaded newer image for bde2020/hadoop-datanode:1.1.0-hadoop2.8-java8
Pulling spark-master (bde2020/spark-master:2.1.0-hadoop2.8-hive-java8)...
2.1.0-hadoop2.8-hive-java8: Pulling from bde2020/spark-master
6d827a3ef358: Already exists
2726297beaf1: Already exists
7d27bd3d7fec: Already exists
e61641c845ed: Already exists
cce4cca5b76b: Already exists
6826227500b0: Already exists
c03b117ffd91: Already exists
821a1547b435: Already exists
53d8cd7022f9: Pull complete
adeb8ae151f4: Pull complete
48d9dad952fc: Pull complete
f55b5ba1a0f9: Pull complete
dfe31f13fb6a: Pull complete
30a71e496ee1: Pull complete
7fa335f1d6a7: Pull complete
8f8cf691c243: Pull complete
017d225c721b: Pull complete
df1d318013a5: Pull complete
bb3983f9034f: Pull complete
b7cd338b1182: Pull complete
08c280769fd2: Pull complete
a385459110fc: Pull complete
ba75e978d523: Pull complete
e7d32f7da80d: Pull complete
217ae643f4e3: Pull complete
dc7010934196: Pull complete
Digest: sha256:57f81f2ae69b5875296eb1d70eb71f3138051783ff5f67f5a74b79dfd8fd20fe
Status: Downloaded newer image for bde2020/spark-master:2.1.0-hadoop2.8-hive-java8
Pulling spark-worker (bde2020/spark-worker:2.1.0-hadoop2.8-hive-java8)...
2.1.0-hadoop2.8-hive-java8: Pulling from bde2020/spark-worker
6d827a3ef358: Already exists
2726297beaf1: Already exists
7d27bd3d7fec: Already exists
e61641c845ed: Already exists
cce4cca5b76b: Already exists
6826227500b0: Already exists
c03b117ffd91: Already exists
```

```
838b53057985: Pull complete
Digest: sha256:68dc13aa09ff067b623d4d5b23a9466b8260140f8c920559cff9f4166ff51f0a
Status: Downloaded newer image for bde2020/spark-worker:2.1.0-hadoop2.8-hive-java8
Pulling spark-notebook (bde2020/spark-notebook:2.1.0-hadoop2.8-hive)...
2.1.0-hadoop2.8-hive: Pulling from bde2020/spark-notebook
6d827a3ef358: Already exists
2726297beaf1: Already exists
7d27bd3d7fec: Already exists
e61641c845ed: Already exists
cce4cca5b76b: Already exists
6826227500b0: Already exists
c03b117ffd91: Already exists
821a1547b435: Already exists
53d8cd7022f9: Already exists
adeb8ae151f4: Already exists
48d9dad952fc: Already exists
f55b5ba1a0f9: Already exists
dfe31f13fb6a: Already exists
30a71e496ee1: Already exists
7fa335f1d6a7: Already exists
8f8cf691c243: Already exists
017d225c721b: Already exists
df1d318013a5: Already exists
5cc06129b962: Pull complete
80446fdc2644: Pull complete
94cb32313e11: Pull complete
beef8a48f3c1: Pull complete
891d0ee4e2dc: Pull complete
6b63a2e308f8: Pull complete
6a4d0d5cf7df: Pull complete
Digest: sha256:cb8d641b42744af8cca691e7e5c07af401457666ec870b8d9c8445228cbed46e
Status: Downloaded newer image for bde2020/spark-notebook:2.1.0-hadoop2.8-hive
Pulling hue (bde2020/hdfs-filebrowser:3.11)...
3.11: Pulling from bde2020/hdfs-filebrowser
f25e451100bc: Pull complete
2dbe4abf311d: Pull complete
7ae2bc99836a: Pull complete
a3ed95caeb02: Pull complete
f1a0a565f855: Pull complete
```

```
Pulling jupyter-pyspark-notebook (jupyter/pyspark-notebook:)...
latest: Pulling from jupyter/pyspark-notebook
d5fd17ec1767: Pull complete
9bcd929937a5: Pull complete
409ab282c108: Pull complete
4f4fb700ef54: Pull complete
7a2cc705730a: Pull complete
b4e0a265eba2: Pull complete
96626bb5d880: Pull complete
8255dd6a1550: Pull complete
33f43eda0f10: Pull complete
d52d2c99758d: Pull complete
14a752070a26: Pull complete
e26a3aca7dfe: Pull complete
991ac8860886: Pull complete
cd8464a3d1a0: Pull complete
b5b05a9056e0: Pull complete
aa6e80757b73: Pull complete
6623e3d96572: Pull complete
4a9ca90ae021: Pull complete
9b6b2f739a51: Pull complete
```

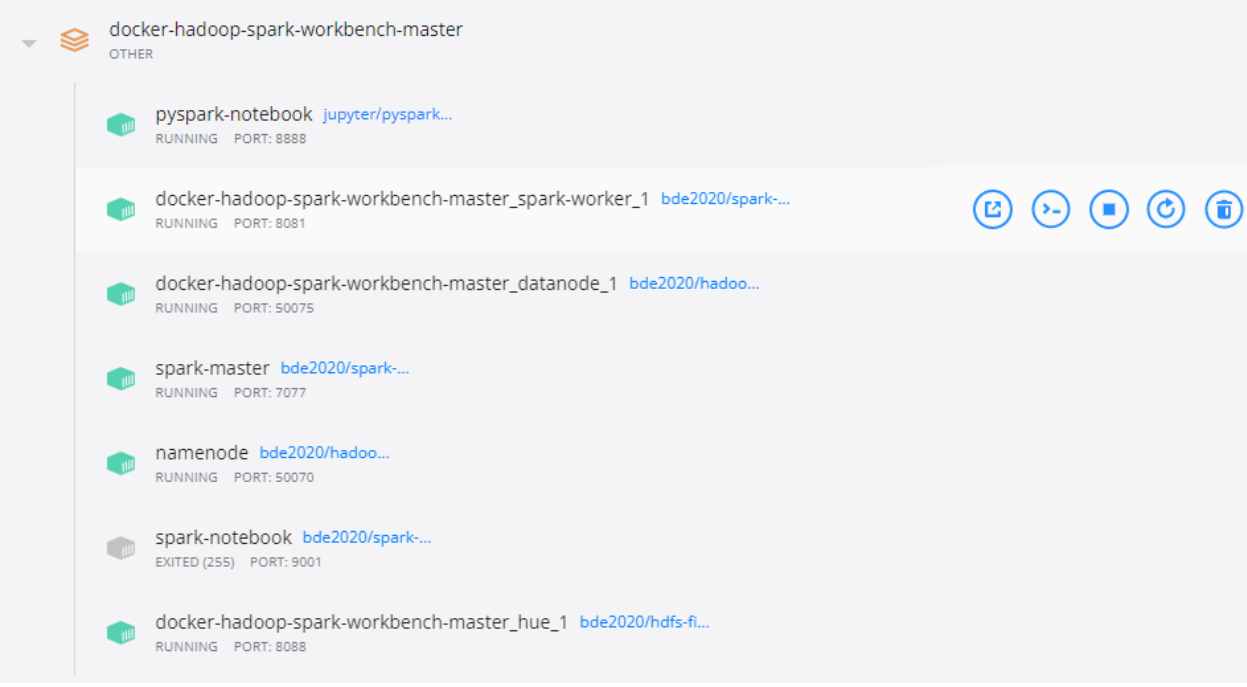
```
Status: Downloaded newer image for bde2020/hdfs-filebrowser:3.11
Creating spark-master                                  ... done
Creating namenode                                      ... done
Creating spark-notebook                                ... done
Creating docker-hadoop-spark-workbench-master_hue_1    ... done
Creating pyspark-notebook                              ... done
Creating docker-hadoop-spark-workbench-master_datanode_1      ... done
Creating docker-hadoop-spark-workbench-master_spark-worker_1  ... done

C:\Namra\IBA-Masters\BigDataAnalytics\Project\docker-hadoop-spark-workbench-master>
```

After launching docker-compose file, checking whether all containers are running:

```
C:\Namra\IBA-Masters\BigDataAnalytics\Project\docker-hadoop-spark-workbench-master>docker ps
CONTAINER ID   IMAGE                                      COMMAND                  CREATED         STATUS                   PORTS
               NAMES
4b57b6a304be   bde2020/spark-worker:2.1.0-hadoop2.8-hive-java8   "entrypoint.sh /bin/…"   5 minutes ago   Up 5 minutes (healthy)   0.0.0.0:8081->8081/tcp
               docker-hadoop-spark-workbench-master_spark-worker_1
d84b787814a6   bde2020/hadoop-datanode:1.1.0-hadoop2.8-java8     "/entrypoint.sh /run…"   5 minutes ago   Up 5 minutes (healthy)   0.0.0.0:50075->50075/tcp
               docker-hadoop-spark-workbench-master_datanode_1
fb8fff14ce5d   bde2020/hadoop-namenode:1.1.0-hadoop2.8-java8     "/entrypoint.sh /run…"   6 minutes ago   Up 5 minutes (healthy)   0.0.0.0:50070->50070/tcp
               namenode
f34535520e53   bde2020/spark-notebook:2.1.0-hadoop2.8-hive       "/entrypoint.sh /run…"   6 minutes ago   Up 5 minutes             0.0.0.0:9001->9001/tcp
               spark-notebook
7f6a0427b4ef   bde2020/hdfs-filebrowser:3.11                     "/entrypoint.sh buil…"   6 minutes ago   Up 5 minutes             0.0.0.0:8088->8088/tcp
               docker-hadoop-spark-workbench-master_hue_1
e2e24ff634ff   bde2020/spark-master:2.1.0-hadoop2.8-hive-java8   "entrypoint.sh /bin/…"   6 minutes ago   Up 5 minutes (healthy)   0.0.0.0:7077->7077/tcp, 6066/tcp, 0.0
.0.0:8080->8080/tcp   spark-master
dae335a5d008   ubuntu:latest                              "bash"                   3 hours ago     Up 3 hours
               ecstatic_almeida
```

Below is the docker interface which shows containers running except sprak-notebook. It was exiting itself despite running in detach mode:



Using docker cp command to copy data to Hadoop name node:

```
C:\Namra\IBA-Masters\BigDataAnalytics\Project\data>docker cp C:/Namra/IBA-Masters/BigDataAnalytics/Project/data fb8fff14ce5d:/hadoop/dfs/name
```

Bashing into Hadoop name node to check whether gene sequencing file was successfully copied into the container:

```
C:\Namra\IBA-Masters\BigDataAnalytics\Project\docker-hadoop-spark-workbench-master>docker exec -it fb8fff14ce5d bash
root@fb8fff14ce5d:/# ls
bin   dev           etc     hadoop-data  lib    media  opt   root  run.sh  srv  tmp  var
boot  entrypoint.sh  hadoop  home         lib64  mnt    proc  run   sbin    sys  usr
root@fb8fff14ce5d:/# cd hadoop
root@fb8fff14ce5d:/hadoop# ls
dfs
```

Hadoop -> dfs -> name -> data -> GRCh38_latest_genomic.fna:

```
root@fb8fff14ce5d:/hadoop# cd dfs
root@fb8fff14ce5d:/hadoop/dfs# ls
name
root@fb8fff14ce5d:/hadoop/dfs# cd name
root@fb8fff14ce5d:/hadoop/dfs/name# ls
current  data  in_use.lock
root@fb8fff14ce5d:/hadoop/dfs/name# cd data
root@fb8fff14ce5d:/hadoop/dfs/name/data# ls
GRCh38_latest_genomic.fna
```

Copied data to spark-master docker container in home directory:

```
C:\Namra\IBA-Masters\BigDataAnalytics\Project\docker-hadoop-spark-workbench-master>docker cp C:/Namra/IBA-Masters/BigDataAnalytics/Project/data e2e24ff634ff:/home
```

Bashing into spark-master and going into data directory to see if dataset has been successfully copied:

```
C:\Namra\IBA-Masters\BigDataAnalytics\Project\docker-hadoop-spark-workbench-master>docker exec -it spark-master /bin/bash
root@e2e24ff634ff:/# ls
bin   entrypoint.sh    finish-step.sh  lib        media  proc  sbin   sys  var
boot  etc              hadoop-data     lib64      mnt    root  spark  tmp  wait-for-step.sh
dev   execute-step.sh  home            master.sh  opt    run   srv    usr
root@e2e24ff634ff:/# cd home
root@e2e24ff634ff:/home# ls
data
```

We see that dataset file is in there:

```
root@e2e24ff634ff:/home# cd data
root@e2e24ff634ff:/home/data# ls
GRCh38_latest_genomic.fna
```

Enabling Spark Session:

```
root@e2e24ff634ff:/home/data# cd /
root@e2e24ff634ff:/# /spark/bin/spark-shell --master=local spark://spark-master:7077
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLeve
22/05/26 16:21:21 WARN util.NativeCodeLoader: Unable to load native-hadoop library for yo
22/05/26 16:21:33 WARN metastore.ObjectStore: Version information not found in metastore.
ion 1.2.0
22/05/26 16:21:33 WARN metastore.ObjectStore: Failed to get database default, returning N
22/05/26 16:21:35 WARN metastore.ObjectStore: Failed to get database global_temp, returni
Spark context Web UI available at http://172.19.0.5:4040
Spark context available as 'sc' (master = local, app id = local-1653582082772).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 2.1.2-SNAPSHOT
      /_/

Using Scala version 2.11.8 (OpenJDK 64-Bit Server VM, Java 1.8.0_121)
Type in expressions to have them evaluated.
Type :help for more information.
```

Creating a network bridge between Hadoop and spark:

```
C:\Namra\IBA-Masters\BigDataAnalytics\Project\docker-hadoop-spark-workbench-master>docker network create -d bridge hadoopspark
47344ae4403fd9039841e8f38a2e5f0b55077857043444b9be92c5c577ff6605
```

Inspecting bridge network which is empty:

```
C:\Namra\IBA-Masters\BigDataAnalytics\Project\docker-hadoop-spark-workbench-master>docker inspect hadoopspark
[
    {
        "Name": "hadoopspark",
        "Id": "47344ae4403fd9039841e8f38a2e5f0b55077857043444b9be92c5c577ff6605",
        "Created": "2022-05-26T20:00:30.3841525Z",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "172.20.0.0/16",
                    "Gateway": "172.20.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {},
        "Options": {},
        "Labels": {}
    }
]
```

I added jupyter notebook service to spark docker-compose file as well.

```
C:\Namra\IBA-Masters\BigDataAnalytics\Project\docker-hadoop-spark-workbench-master>docker-compose -f docker-compose-hive.yml up -d spark-master spark-worker spark-noteb
ook hue
Recreating spark-master ...
Recreating spark-master   ... done
Recreating spark-notebook ... done
Recreating docker-hadoop-spark-workbench-master_spark-worker_1 ... done
```

Transfer data to HDFS:

Bashing into Hadoop name node. Inside directory “Hadoop-2.8.0” -> bin: With command:

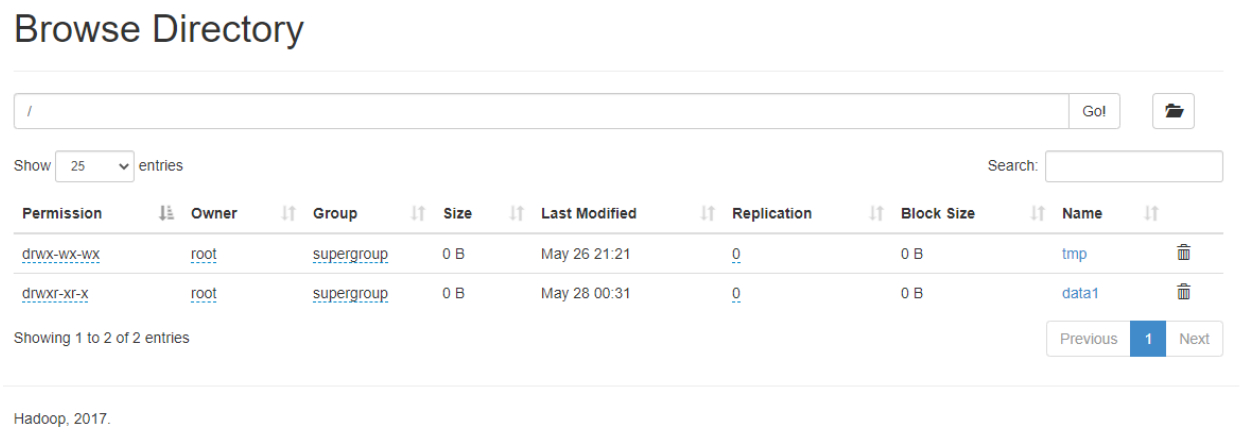
hadoop dfs -mkdir /data1: Directory name in hdfs is data1.

```
root@84adf7260ac4:/hadoop# cd ..
root@84adf7260ac4:/# cd opt
root@84adf7260ac4:/opt# ls
hadoop-2.8.0
root@84adf7260ac4:/opt# cd hadoop-2.8.0
root@84adf7260ac4:/opt/hadoop-2.8.0# ls
LICENSE.txt  NOTICE.txt  README.txt  bin  etc  include  lib  libexec  logs  sbin  share
root@84adf7260ac4:/opt/hadoop-2.8.0# cd bin
root@84adf7260ac4:/opt/hadoop-2.8.0/bin# ls
container-executor  hadoop.cmd  hdfs.cmd  mapred.cmd  test-container-executor  yarn.cmd
hadoop              hdfs        mapred    rcc         yarn
root@84adf7260ac4:/opt/hadoop-2.8.0/bin# hadoop dfs -mkdir /data1
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.
```

Putting data in hdfs using command: Hadoop dfs -put /Hadoop/dfs/name/GRCh38_latest_genomic.fna inside hdfs directory data1:

```
root@84adf7260ac4:/opt/hadoop-2.8.0/bin# hadoop dfs -put /hadoop/dfs/name/data/GRCh38_latest_genomic.fna /data1
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.
```

We can see “data1” as directory in hdfs on local host:



Inside datat1 directory, we can see the data set file having size 3.11 GB:

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	root	supergroup	3.11 GB	May 28 00:31	3	128 MB	GRCh38_latest_genomic.fna

Showing 1 to 1 of 1 entries

Previous 1 Next

We can see Data node information:

Datanode Information

In service Down Decommissioned Decommissioned & dead

In operation

Show 25 entries Search:

Node	Http Address	Last contact	Capacity	Blocks	Block pool used	Version
c69a983b5ee9:50010 (172.19.0.8:50010)	c69a983b5ee9:50075	2s	237.87 GB	25	3.13 GB (1.32%)	2.8.0

Showing 1 to 1 of 1 entries

Previous 1 Next

Inside “Overview”, we can see information of hdfs memory used with block pool used: 3.11 GB:

Configured Capacity:	237.87 GB
DFS Used:	3.13 GB (1.32%)
Non DFS Used:	128.08 GB
DFS Remaining:	106.66 GB (44.84%)
Block Pool Used:	3.13 GB (1.32%)
DataNodes usages% (Min/Median/Max/stdDev):	1.32% / 1.32% / 1.32% / 0.00%
Live Nodes	1 (Decommissioned: 0)
Dead Nodes	0 (Decommissioned: 0)
Decommissioning Nodes	0
Total Datanode Volume Failures	0 (0 B)
Number of Under-Replicated Blocks	25
Number of Blocks Pending Deletion	0
Block Deletion Start Time	Fri May 27 20:57:28 +0500 2022
Last Checkpoint Time	Fri May 27 20:57:31 +0500 2022

This is containerized jupyter notebook for pyspark. I loaded my data set here using SeqIO module of Biopython package: It took a lot of time to figure out how to load and parse the data set. The format was fasta and after multiple tries, I finally figured out that we had to use work directory that was present in jupyter notebook container:

```
count = 0
for seq_record in SeqIO.parse("/work/GRCh38_latest_genomic.fna", "fasta"):
    count = count + 1
    print(seq_record.seq)
print(count)
```

```
IOPub data rate exceeded.
The Jupyter server will temporarily stop sending output
to the client in order to avoid crashing it.
To change this limit, set the config variable
`--ServerApp.iopub_data_rate_limit`.

Current values:
ServerApp.iopub_data_rate_limit=1000000.0 (bytes/sec)
ServerApp.rate_limit_window=3.0 (secs)
```

```
GAATTCACTCAGCAGTCAGAAGAAAAGGAAGCCTCTTGTTGGAATTCATACTAAGAAGATCAGAAAGTGTAGTGGAAAAGAGAGGACTAGAACTGCAGGTGCAAGTAGAAGTTGGTGTCCAAAAAGA
AGTGTATTTGTATTGTTCATTTGACCTCATTTATTTTAGTATGCAAGCGtgtgcttaaaaatatttcagcctTCTTTTTACTATATATGCAACTCATCATCATTTATTGAGAGATTATATATCCATC
ATTACTTCAATGATCTCATTTttagctttcttcttctttattaatggATGCAGTAACTTCTGGCATTCCCCTGAGAATATTAATTCTACTTATTTGGAAGTACTCTTTTTATTGCTCTCTGctgtct
ttcttttgtattttatacatctttcatttgctttatgttattttaaatcttaCCCTATGCACTTACAATGATGGCACTTCTCTGAATGActgttgattttcttcctttctttctttttttcatgttg
tctacatttttttttcaaggtgagTTTCTGCTAGACTGTTATTACTGAGCAATCCTTTGGTAGAGAAGGAGGTTCATGCTGCTACAGGCCACAATTTGCCTCATTTGCAGTCAGATCAGGCTTTTTG
TCAGAGAGAGGACATGCCCATTAGGCAATATGTATTCAGATATTTCCAACTGATCgctgccttttctttctgattcaaatTCCCATTCTCAATATATTTACCTAATGTATGCCTCCCTTTTGCCTTT
TTAAGCAAGACTAAAAAATCAATACCCTtgaaaaaaattgattaatttgGCTAACTTAAATTTAAAGTCTTCACTTCTAGCAAAATAGTTTTACTACATGTTTTTTACTATGAGAAAATATGTAAAC
TATGTATACCAAAgcaaatgtataaataatttagaaaaataaaaatatttcaatagaaaagtgggaaaatgtaaaaataggtaCTTACTGGAAGAATCAAGAGATTAGTTCTAAAGaccttaattat
ttatttaaactaatttaattaaaatttttaaaaataattaaatttttcaaTAATCAAGTAAAACAAATTAAACTTAAACAATAgaacctttttattttgagatgcaaaAGCATGAATCAAAGACATCTG
TAAAGCTTAAATGTATTCAGATTGTAAAGCTAACCCATGTAAACCAAACTAGGCCAGGAATTAGACTATATTAGTGCCCCAGACTCCCCCAAGTGTCCCTTCCTAAAAGAAGCTCCTCTCACCCTaa
gttttatgtataatttttttgctGTGCTTCAAAAAGGTATAGCATGTATGCATCTCTGAACAACATACCTTAGTTTTTTTTGGCCTGCTTTGATAATTACTATGAAATcctctttctgtattatttt
gtgCCACACTCCTTTTGTTCAACATATGTTTATGAGTGACAGCCCTGTTGTTGCAAGCggttgtaatttatttattttcactgctcTATAATATTTGATTGTTTATATAGAtgagaattttctattt
tattccttatatacatttgtgttattttaaagtttGGGTGCTTAGGAGCAAAGTCAATATGAAAATACTTGGACATATGCTCTCAGAATTATATGCATAAGTTTATCTACAACATATAACTATCCAA
```

In order to download the dataset directly into jupyter notebook, I tried wget command:

```
pip install wget
```

```
Requirement already satisfied: wget in /opt/conda/lib/python3.9/site-packages (3.2)
Note: you may need to restart the kernel to use updated packages.
```

```
!python -m wget https://ftp.ncbi.nlm.nih.gov/refseq/H_sapiens/annotation/GRCh38_latest/refseq_identifiers/GRCh38_latest_genomic
```

```
 60% [.............................                    ] 588906496 / 972898531Traceback (most recent call last):
  File "/opt/conda/lib/python3.9/runpy.py", line 197, in _run_module_as_main
    return _run_code(code, main_globals, None,
  File "/opt/conda/lib/python3.9/runpy.py", line 87, in _run_code
    exec(code, run_globals)
  File "/opt/conda/lib/python3.9/site-packages/wget.py", line 568, in <module>
    filename = download(args[0], out=options.output)
  File "/opt/conda/lib/python3.9/site-packages/wget.py", line 526, in download
    (tmpfile, headers) = ulib.urlretrieve(binurl, tmpfile, callback)
  File "/opt/conda/lib/python3.9/urllib/request.py", line 278, in urlretrieve
    raise ContentTooShortError(
urllib.error.ContentTooShortError: <urlopen error retrieval incomplete: got only 588906496 out of 972898531 bytes>
```

I tested wget command inside terminal in jupyter notebook container to load dataset from internet. I successfully downloaded the data set here in terminal but wasn't able to figure out how to load it in jupyter notebook container:

```
GRCh38_latest_genomic.fna.gz  100%[++++++++++++++++++++++++++++++++++++++++++++++++++>] 927.83M   647KB/s    in 16m 10s

2022-05-29 09:39:11 (431 KB/s) - 'GRCh38_latest_genomic.fna.gz' saved [972898531/972898531]

(base) jovyan@220baabef7ed:~/work$ ls
GRCh38_latest_genomic.fna.gz  GRCh38_latest_genomic.fna.gz7ius4ncd.tmp  index.shtml  Untitled.ipynb
```

```
(base) jovyan@220baabef7ed:~/work$ wget https://ftp.ncbi.nlm.nih.gov/refseq/H_sapiens/annotation/GRCh38_latest/refseq_id
entifiers/GRCh38_latest_genomic.fna.gz
--2022-05-29 08:31:36--  https://ftp.ncbi.nlm.nih.gov/refseq/H_sapiens/annotation/GRCh38_latest/refseq_identifiers/GRCh3
8_latest_genomic.fna.gz
```

This is the command to convert the dataset into list so that I can use ‘ids’ of every sequence:

```
# Biopython parses the file into a list of SeqRecord objects
seqs = list(SeqIO.parse("/work/GRCh38_latest_genomic.fna", "fasta"))
print(seqs)
```

```
[SeqRecord(seq=Seq('NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN...NNN'), id='NC_000001.11', name='NC_000001.11', des
cription='NC_000001.11 Homo sapiens chromosome 1, GRCh38.p14 Primary Assembly', dbxrefs=[]), SeqRecord(seq=Seq('GAATTCAGCTGAGAA
GAACAGGCAAGGACTTAGGAAATATTCCTTATTTGAAGG...TTC'), id='NT_187361.1', name='NT_187361.1', description='NT_187361.1 Homo sapiens ch
romosome 1 unlocalized genomic scaffold, GRCh38.p14 Primary Assembly HSCHR1_CTG1_UNLOCALIZED', dbxrefs=[]), SeqRecord(seq=Seq
('AGGGGTCTGCTTAGAGAGGGTCTCATTAGTGGGGTCTAGTAGTGGGGTTTTGGT...TTC'), id='NT_187362.1', name='NT_187362.1', description='NT_187362.
1 Homo sapiens chromosome 1 unlocalized genomic scaffold, GRCh38.p14 Primary Assembly HSCHR1_CTG2_UNLOCALIZED', dbxrefs=[]), Se
qRecord(seq=Seq('gatcgagactatcctggctaacacggtgaaaccccgtctctactaaaaatacaa...ATC'), id='NT_187363.1', name='NT_187363.1', descript
ion='NT_187363.1 Homo sapiens chromosome 1 unlocalized genomic scaffold, GRCh38.p14 Primary Assembly HSCHR1_CTG3_UNLOCALIZED',
dbxrefs=[]), SeqRecord(seq=Seq('gaattcattcgatgacgattccattcaattccgttcaatgattccattagattc...TTC'), id='NT_187364.1', name='NT_1873
64.1', description='NT_187364.1 Homo sapiens chromosome 1 unlocalized genomic scaffold, GRCh38.p14 Primary Assembly HSCHR1_CTG4
_UNLOCALIZED', dbxrefs=[]), SeqRecord(seq=Seq('AAACTGtgcagcctctatggaaaacagtaccgtggttcctcaaaaaaattaaga...ATC'), id='NT_187365.
1', name='NT_187365.1', description='NT_187365.1 Homo sapiens chromosome 1 unlocalized genomic scaffold, GRCh38.p14 Primary Ass
embly HSCHR1_CTG5_UNLOCALIZED', dbxrefs=[]), SeqRecord(seq=Seq('tgtgtatgtgtatgtgtatgtatgtgtatgtgtatgtgtatatgtatatgtata...GCT'),
id='NT_187366.1', name='NT_187366.1', description='NT_187366.1 Homo sapiens chromosome 1 unlocalized genomic scaffold, GRCh38.p
14 Primary Assembly HSCHR1_CTG6_UNLOCALIZED', dbxrefs=[]), SeqRecord(seq=Seq('GAATTCTCAGAACTTTCAAAACACAAATCTTCATCCGCAGGGATGTTCA
GGAGG...TTC'), id='NT_187367.1', name='NT_187367.1', description='NT_187367.1 Homo sapiens chromosome 1 unlocalized genomic sca
ffold, GRCh38.p14 Primary Assembly HSCHR1_CTG7_UNLOCALIZED', dbxrefs=[]), SeqRecord(seq=Seq('GAGCATTCTCTAGTAGGAAGACCTTTTTAGAGTA
CAAAGGCCAGATCCTGCCCT...aac'), id='NT_187368.1', name='NT_187368.1', description='NT_187368.1 Homo sapiens chromosome 1 unlocali
zed genomic scaffold, GRCh38.p14 Primary Assembly HSCHR1_CTG8_UNLOCALIZED', dbxrefs=[]), SeqRecord(seq=Seq('CTAAACACAAAAagtttag
ctgggcatgatggtgcatgcctgtaatctgagcta...GGA'), id='NT_187369.1', name='NT_187369.1', description='NT_187369.1 Homo sapiens chromo
some 1 unlocalized genomic scaffold, GRCh38.p14 Primary Assembly HSCHR1_CTG9_UNLOCALIZED', dbxrefs=[]), SeqRecord(seq=Seq('NNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN...NNN'), id='NC_000002.12', name='NC_000002.12', description='NC_000002.12 H
omo sapiens chromosome 2, GRCh38.p14 Primary Assembly', dbxrefs=[]), SeqRecord(seq=Seq('GAATTCTCAGAAaccagtttgtgatgtgtgtactcaact
aacagggttgaacct...ttc'), id='NT_187370.1', name='NT_187370.1', description='NT_187370.1 Homo sapiens chromosome 2 unlocalized g
```

Printed ids of all sequences:

```
 # Loop over sequences view important attributes, which can be converted to strings
for record in seqs:
    print("Record id:", record.id)
```

```
Record id: NC_000001.11
Record id: NT_187361.1
Record id: NT_187362.1
Record id: NT_187363.1
Record id: NT_187364.1
Record id: NT_187365.1
Record id: NT_187366.1
Record id: NT_187367.1
Record id: NT_187368.1
Record id: NT_187369.1
Record id: NC_000002.12
Record id: NT_187370.1
Record id: NT_187371.1
Record id: NC_000003.12
Record id: NT_167215.1
Record id: NC_000004.12
Record id: NT_113793.3
Record id: NC_000005.10
Record id: NT_113948.1
Record id: NC_000006.12
Record id: NC_000007.14
Record id: NC_000008.11
Record id: NC_000009.12
Record id: NT_187372.1
Record id: NT_187373.1
Record id: NT_187374.1
Record id: NT_187375.1
```

Printed lengths of all records(sequences):

```
for record in seqs:
    print("length of sequence is: ", len(record))
```

```
length of sequence is:  248956422
length of sequence is:  175055
length of sequence is:  32032
length of sequence is:  127682
length of sequence is:  66860
length of sequence is:  40176
length of sequence is:  42210
length of sequence is:  176043
length of sequence is:  40745
length of sequence is:  41717
length of sequence is:  242193529
length of sequence is:  161471
length of sequence is:  153799
length of sequence is:  198295559
length of sequence is:  155397
length of sequence is:  190214555
length of sequence is:  209709
length of sequence is:  181538259
length of sequence is:  92689
length of sequence is:  170805979
length of sequence is:  159345973
length of sequence is:  145138636
length of sequence is:  138394717
length of sequence is:  40062
length of sequence is:  38054
length of sequence is:  176845
length of sequence is:  39050
length of sequence is:  133797422
```

This is how we can retrieve ‘id’ of a particular sequences by iterating over ‘seqs’ list:

```
print(seqs[0].id)  # first record
print(seqs[-1].id)  # last record
```

```
NC_000001.11
NC_012920.1
```

To get 1st sequence information:

```
print(seqs[0]) #1st sequence
```

```
ID: NC_000001.11
Name: NC_000001.11
Description: NC_000001.11 Homo sapiens chromosome 1, GRCh38.p14 Primary Assembly
Number of features: 0
Seq('NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN...NNN')
```

100th sequence information:

```
print(seqs[100]) #100th sequence
```

```
ID: NT_187432.1
Name: NT_187432.1
Description: NT_187432.1 Homo sapiens unplaced genomic scaffold, GRCh38.p14 Primary Assembly HSCHRUN_RANDOM_136
Number of features: 0
Seq('AGGGCCTCAAAgcacgccaaatatccacttgcagatcctatgaaaagagtgttc...GAG')
```

We can print different sequences:

```
print(seqs[90], seqs[45], seqs[120])
```

```
ID: NT_187422.1
Name: NT_187422.1
Description: NT_187422.1 Homo sapiens unplaced genomic scaffold, GRCh38.p14 Primary Assembly HSCHRUN_RANDOM_126
Number of features: 0
Seq('ggaatgttcaactctatgagttgaatgcaaacatcacaaagaaattctgagaat...att') ID: NT_113930.2
Name: NT_113930.2
Description: NT_113930.2 Homo sapiens chromosome 17 unlocalized genomic scaffold, GRCh38.p14 Primary Assembly HSCHR17_RANDOM_CT
G3
Number of features: 0
Seq('gaattctatgtgagggagaaacactcagaacccagcagcagtgttctggaatcc...TGT') ID: NT_187452.1
Name: NT_187452.1
Description: NT_187452.1 Homo sapiens unplaced genomic scaffold, GRCh38.p14 Primary Assembly HSCHRUN_RANDOM_160
Number of features: 0
Seq('cttaaagtataataaaaaaagaaagtttcaaaactgctgtatcacaagataggt...TCC')
```

I tried coverting ‘seqs’ list into a data frame of python so that further relevant analysis could be done on it and spark sql could be used but it gave memory error:

```
df = pd.DataFrame(genes, columns=['sequence','ID','Name','Description'])
```

```
---------------------------------------------------------------------------
MemoryError                               Traceback (most recent call last)
Input In [8], in <cell line: 1>()
----> 1 df = pd.DataFrame(genes, columns=['sequence','ID','Name','Description'])

MemoryError: Unable to allocate 9.27 GiB for an array with shape (5, 248956422) and data type object
```

It was a real deal to get the list converted to a data frame. I tried as above as well as through pyspark too to convert it into a spark sql’s data frame. My idea was to create tables from that data frame and run sql

queries on it. But since my file was in fasta format, I couldn't find any command that could load fasta dataset file in spark.

Note: I created spark context too as well as spark sql context, but it was difficult to integrate spark sql, python and my data set file. My data set format was in 'fasta', one that is not easily manageable and required 'Biopython' package to be parsed. Hence, combining 'Biopython', spark and that too within a container required in depth knowledge. Although I spent 3 days just trying to parse this file and integrate all of the above, but eventually figured out that I can't work with spark while I am working with Biopython without guidance.

Cutting short, I decided to work with just Biopython because this package contains modules relevant to my need.

Genome Sequence Analysis

We can print a sequence and store it in a variable:

```
dna_1 = seqs[90]
dna_1
```

```
SeqRecord(seq=Seq('ggaatgttcaactctatgagttgaatgcaaacatcacaaagaaattctgagaat...att'), id='NT_187422.1', name='NT_187422.1', descri
ption='NT_187422.1 Homo sapiens unplaced genomic scaffold, GRCh38.p14 Primary Assembly HSCHRUN_RANDOM_126', dbxrefs=[])
```

Type can be known using 'type' method. We can see that sequences were parsed using 'SeqRecord' class of Biopython:

```
type(dna_1)
```

```
Bio.SeqRecord.SeqRecord
```

We can also use the following command to assign record to a variable. 'break' will stop the loop after 1st iteration which means 1st record gets assigned to the mentioned variable:

```
for record in seqs:
    dna_2 = record
    break
```

Printing dna_2 record:

```
dna_2
```

```
SeqRecord(seq=Seq('NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN...NNN'), id='NC_000001.11', name='NC_000001.11', desc
ription='NC_000001.11 Homo sapiens chromosome 1, GRCh38.p14 Primary Assembly', dbxrefs=[])
```

Printing id and just the sequence like this:

```
dna_2.id
```

```
'NC_000001.11'
```

```
dna_2.seq
```

```
Seq('NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN...NNN')
```

We can print the sequence record using indexes:

Furthermore, *'.translate()'* method converts the dna sequence into a protein sequence. We can see in 2nd command that sequence at index=39 has been converted into a protein sequence and hence, the sequence record does not only contain the letters "A,C,T,G", which are part of a dna (gene) sequence.

```
seqs[39]
```

```
SeqRecord(seq=Seq('GCCTTCAGAGTTACAAGGCTATATCAGTTTTCCACGATTGTTTCCCCTTTTTGT...CAG'), id='NT_187381.1', name='NT_187381.1', descri
ption='NT_187381.1 Homo sapiens chromosome 14 unlocalized genomic scaffold, GRCh38.p14 Primary Assembly HSCHR14_CTG8_UNLOCALIZE
D', dbxrefs=[])
```

```
seqs[39].translate()
```

```
/opt/conda/lib/python3.9/site-packages/Bio/Seq.py:2979: BiopythonWarning: Partial codon, len(sequence) not a multiple of three.
Explicitly trim the sequence or add trailing N before translation. This may become an error in future.
  warnings.warn(
```

```
SeqRecord(seq=Seq('AFRVTRLYQFSTIVSPFCCWLFPPLFYMFYFSFPLFSLLMGCKTSQHLKYR*K*...RLG'), id='<unknown id>', name='<unknown name>', de
scription='<unknown description>', dbxrefs=[])
```

Checking sequence at index=700 which is in lower case:

```
seqs[700].seq
```

```
Seq('tttctttctttttttttttttgtggtgaggacccttaagatctactctcccag...TTC')
```

We can edit sequence by adding different parts of the sequence together:

```
edited = seqs[39][:10] + seqs[39][11:]
print(edited.seq)
```

```
GCCTTCAGAGTACAAGGCTATATCAGTTTTCCACGATTGTTTCCCCTTTTTGTTGTTGGTTATTTCCacctttattttatatgttttacttctctttccccctattttctttattaatgggATGC
AAGACTTCACAGCATTTGAAATATAGGTAAAAATGAGCTATCCTAACAAACTGGGACCTATTTATCAAGGAATAATCGGTCCTACCCATGAAAGATAAAACAGCACGGGAGACCAGAGACaaatt
tttttagtaaaatacattctctgaaagattttgaaaaagaagaggtggggaaagtgtgaaaggaaaataaaaacttgtgaTTTCAATTCATTATGTcatgaggggaaaaaaaaaactaaaagatg
agtcatgcaagaaactgattttcctttcattcctaagaaaatagctacagataaaaggttaaatagtttcacagatagctactatttgttcatttttgaaacaGTGCAGGAGAACGACCTAATTT
TCTATTTCCCTATGTGCTTCTTTTCCATTACAACATGTAAATTCTCATACGGTCCTTCTTTCCCCTCTAGCCAGCTTTTCCCTTTTATATATTGAAAGCTCTAAAAACCGTCCTTGGGAGAACGG
CACTGACCACACACTGTTTCTGTgattacttttacttttcttccaggCATGTCCTAACTttggcaaaattaattttaatttgattgaCATCTGTCTCAGAAATCTTTGATTTGCACTAGGAAAGA
TGCCAAATTAGGAGCCAGTTACTGTAAAACTCAGCCATACCACTGTgcgtgcatgtgtgagtgtgtgtgtgtgtacatgtgtgctttaatttttgtgggcttTAAGCCATGTCGTTCTCTCTGTG
AAGATACGATTTGGCGTGACCTTTGAATAgagaattctaaaataaataagagtcTCCTATGAATTCTGTGAAAATTTCTGGACTCACCAATGATCTTGACTGTGTCATTGCATCTGACAGTCCCA
GGGAAGAGACTCTCTGGTGGTTTCATAAAATCTGTGCTTGGGTTCCCCCTGCAGATTACTGGGTATAGTGATGCCAAATCACTGTTTAAGAGACAATTTCAAAACATAAGATGCTGCTGAAAGAG
CATTTTGAATCAGGGGACAGCCCCTTCATTGTGAGAGAGCGACATTGGGAGAATATGCTCTGTGAGCCCAAACAGCATCCTCTGCAGGGTGAGGGCAGAGCGGCAGGGCAAGCCCAGAGCCCACT
CAACACAGATGTCAGCCCTGGAGCTGCTGCAGAGGAGTCTGAGGAGAAAAATTTTCCAGCACCTGAATTACACTTATTTCAAAAcgaaaatgcaattaaaaagttaaaaaaagtaattaagacac
agtggcttacacctataatctgaGAAATTtgaggctgaggttggaagattacatgaacccaggagtttgagatcagcctgtgaAAAATAGTGaggcttcatttttatttctgaaaaaaaaattaa
cccaatgAGTCAATGAGAACcaaacttgaaaaaagaaaatttagagtTTTCATATATCTTAATAGTTGGAAGATgtagaacaaaaataattttatcaattcAATGTGTGCAAATACTGAGAGAC
ACACTCATGCCCAGAATTCAACCTGCAGAGGGCAAAACCCAAAAAAATGTAGAGGTTGTTAATGTTCCATTTGAAGGTGAGATCATTTTGAGGACCATGTCCTGTGAGAGTCTGTTTCTCTATTA
GAAGAGTTCTGTACTCATAAAGTTCTGGACATGCCAGGAGACAAATATCAGTAAACAAACATCAGAACTTGAACTTCAGCTTCCCACTGTGGCATTCTCCATGTGTCATCTCTCTAGTATTTCTC
ATGCTAGATCAGGTCTTTAGCTATGAAATATTCCTCCTAATTAATGTAAATAGCTTGAAGTCTACAgagttaaatatgtatattttctcctgtttttctcaGTTCTTCCCTCCCACAGCTCCAAT
```

I again checked if I can work with pyspark by unzipping the dataset file within the jupyter/pyspark-notebook container in cli:

```
(base) jovyan@220baabef7ed:~/work$ gunzip GRCh38_latest_genomic.fna.gz
(base) jovyan@220baabef7ed:~/work$ ls
genes.fna                                GRCh38_latest_genomic.fna.gzgqvj5s3u.tmp  Project-3.ipynb
GRCh38_latest_genomic.fna                GRCh38_latest_genomic.fna.gznft1arru.tmp
GRCh38_latest_genomic.fna.gz7ius4ncd.tmp  index.shtml
```

'less' with dataset file name shows the contents of the file. Since, the file was huge, I haven't captured the ouput.

```
(base) jovyan@220baabef7ed:~/work$ less GRCh38_latest_genomic.fna
(base) jovyan@220baabef7ed:~/work$
```

Back to notebook:

Printing length of a particular sequence this way to get the number of nucleotides in a sequence:

```
print(f'The genome of human chromosome 1, sequence#90, consists of {len(dna_1)} nucleotides. ')
```

```
The genome of human chromosome 1, sequence#90, consists of 1774 nucleotides.
```

We can get the molecular weight of a gene sequence using this module from Biopython: We can see that the weight of dna_1 sequence is 546583.88:

```
molecular_weight(dna_1.seq)
```

```
546583.879900005
```

Higher GC content in s gene sequence implied more stable molecule due to G and C forming triple hydrogen bonds. Here I have used GC library to get the gc content in dna_1 sequence which is 37%:

```
# GC Content - higher GC Content implies more stable molecule due to G and C forming triple hydrogen bonds
from Bio.SeqUtils import GC
GC(dna_1.seq)
```

```
37.4859075535513
```

I created a dictionary for a particular dna sequence to show the count of nucleotides separately:

```
count_nucleotides = {
    'A' : dna_1.seq.count('A'),
    'T' : dna_1.seq.count('T'),
    'C' : dna_1.seq.count('C'),
    'G' : dna_1.seq.count('G')}
```

We can see the count of all nucleotides present in dna_1 sequence as follows:

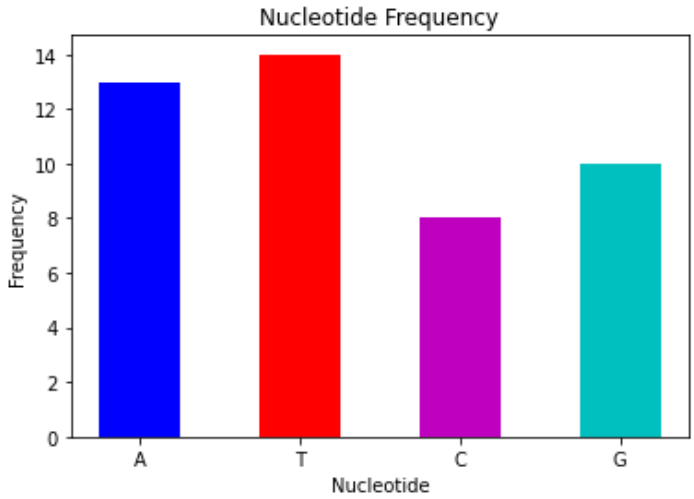
```
count_nucleotides
```

```
{'A': 13, 'T': 14, 'C': 8, 'G': 10}
```

I plotted a bar graph to show number of nucleotides as frequency on y-axis and nucleotide on x-axis:

```
import matplotlib.pyplot as plt
width = 0.5
plt.bar(count_nucleotides.keys(), count_nucleotides.values(), width, color=['b','r','m','c'])
plt.xlabel('Nucleotide')
plt.ylabel('Frequency')
plt.title('Nucleotide Frequency')
```

```
Text(0.5, 1.0, 'Nucleotide Frequency')
```



To get the protein sequence version of dna_1 sequence, using translate() method:

```
# Understanding the information stored in genomes is crucial to finding cures and vaccines.
# So let's check translation and transciption.
mrna = dna_1.translate()
mrna.seq
```

```
Seq('GMFNSMS*MQTSQRNSENAAVYLLFEFPLPTKTSKLSKYPLADSTKRVFQNCSI...GRY')
```

Here, most occurring amino acids are shown at the top from the protein sequence of dna_1. ‘S’ is occurring 65 time in this protein sequence which stands for ‘Serine’ amino acid:

```
# most common amino acids
from collections import Counter
common_amino = Counter(mrna)
common_amino.most_common(10)
```

```
[('S', 65),
 ('F', 62),
 ('L', 48),
 ('K', 42),
 ('R', 38),
 ('N', 32),
 ('Q', 32),
 ('T', 32),
 ('P', 29),
 ('I', 27)]
```

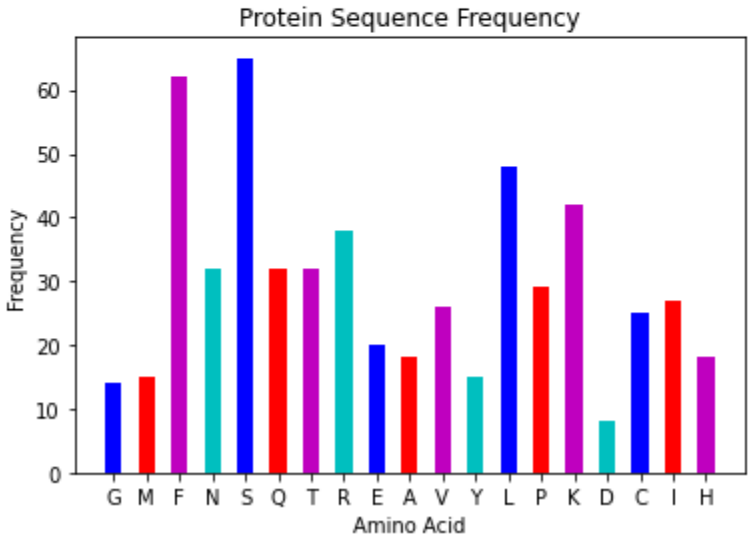
Star symbol (‘*’) is known as a stop codon in a protein sequence. In order to remove its occurrence, I am deleting this codon from a particular sequence to check frequency of amino acids:

The bar char below shows the count of amino acids in a particular protein sequence. We see that ‘S’ and ‘F’ is widely present in this sequence:

```
del common_amino['*']  # delecting stop codon

width = 0.5
plt.bar(common_amino.keys(), common_amino.values(), width, color=['b','r','m','c'])
plt.xlabel('Amino Acid')
plt.ylabel('Frequency')
plt.title('Protein Sequence Frequency')
```

```
Text(0.5, 1.0, 'Protein Sequence Frequency')
```



To get the count of all amino acids in a certain protein sequence, I ran the following command:

```
print(f'Human genome for mrna(dna_1) has {sum(common_amino.values())} amino acids')
```

```
Human genome for mrna(dna_1) has 566 amino acids
```

We can break a protein sequence to see few amino acids. Here I set amino acid reach to 5 to see 1st 5 amino acids in this protein sequence:

```
mrna[:5]
```

```
SeqRecord(seq=Seq('GMFNS'), id='<unknown id>', name='<unknown name>', description='<unknown description>', dbxrefs=[])
```

Printing count of amino acids in a particular protein sequence:

```
print(f'We have {len(mrna)} amino acids in dna_1 sequence of human genome')
```

```
We have 591 amino acids in dna_1 sequence of human genome
```