* *CODE FOR READING THE MULTI FASTA FILE, EXTRACTING THE SEQS BASED ON THE ACCESSION NUMBER AND GENE IDENTIFIER, REMOVING THE EXTRACTED SEQS FROM THE PRIMARY FILE AND REWRITING TO ANOTHER FILE, REVERSE COMPLEMENTING THE EXTRACTED SEQS, INTRODUCING RANDOM 1 NUCLEOTIDE CHANGE TO THE EXTRACTED SEQS AND THEN ADDING IT BACK TO THE PRIMARY FILE*

*#open the fasta file in python*

f = open(**"genome.fasta"**, **"r"**)  
print(f.read())

*#extracting subsequences from the fasta file   
#importing module*

**from** Bio **import** SeqIO  
*#defining a function*

**def** extract\_sequences\_by\_accession(accession\_numbers):  
 *# Open the output file for writing and check if the accession number is in the list and writing it to an output file.*

**with** open(**"genome\_1.fasta"**, **'w'**) **as** output:  
**for** record **in** SeqIO.parse(**"genome.fasta"**, **'fasta'**):**if** record.id **in** accession\_numbers:SeqIO.write(record, output, **'fasta'**)  
  
accession\_numbers = [**'gi|60495|gb|X17223|Influenza'**, **'gi|58576|gb|X52226|Influenza'**, **'gi|2271190|gb|AF008733|Influenza'**]  
*#calling the function*

extract\_sequences\_by\_accession(accession\_numbers)

*#removing the extracted subsequences from the file and rewriting it to another file  
# Accession numbers to remove*num\_to\_remove = [**'gi|60495|gb|X17223|Influenza'**, **'gi|58576|gb|X52226|Influenza'**, **'gi|2271190|gb|AF008733|Influenza'**]  
  
*# Open the input and output files***with** open(**"genome.fasta"**, **'r'**) **as** input\_file, open(**"genome\_removed.fasta"**, **'w'**) **as** output\_file:  
 **for** record **in** SeqIO.parse(**"genome.fasta"**, **'fasta'**):**if** record.id **not in** num\_to\_remove:SeqIO.write(record, output\_file, **'fasta'**)

*#reverse complementing the extracted subsequences  
# Read the sequences from the fasta file*records = list(SeqIO.parse(**"genome\_1.fasta"**, **'fasta'**))  
  
*# Reverse complementing each sequence***for** record **in** records:  
 record.seq = record.seq.reverse\_complement()  
  
*# Writing the reversed complement sequences back to the same FASTA file***with** open(**"genome\_1.fasta"**, **'w'**) **as** output\_file:  
 SeqIO.write(records, output\_file, **'fasta'**)

#introducing random one nucleotide change to the extracted seq

#reading the file containimg extracted seqs ,creating an output file (processed\_seq.fasta).Generating a random index within the file and selecting random nucleotides.Replacing the nucleotide at the random index and writing this to the output file.

**import** random  
  
**with** open(**"genome\_1.fasta"**, **'r'**) **as** infile:  
 **with** open(**"processed\_seq.fasta"**, **'w'**) **as** outfile:  
 **for** line **in** infile:  
 **if** line.startswith(**'>'**): outfile.write(line)

**else**:  
 sequence = line.strip

index = random.randint(0, len(sequence) - 1)

new\_nucleotide = random.choice([**'A'**, **'T'**, **'C'**, **'G'**]) mutated\_sequence = sequence[:index] + new\_nucleotide + sequence[  
 index + 1:] outfile.write(mutated\_sequence + **'\n'**)

*##Insert the processed subsequences back to the primary sequence(multifasta).***from** Bio **import** SeqIO  
original\_records = list(SeqIO.parse(**"genome\_removed.fasta"**, **'fasta'**))  
new\_records = list(SeqIO.parse(**"processed\_seq.fasta"**, **'fasta'**))  
modified\_records = original\_records + new\_records  
**with** open(**"genome\_removed.fasta"**, **'w'**) **as** output\_file:  
 SeqIO.write(modified\_records, output\_file, **'fasta'**)

* *CODE FOR CONVERTING THE MULTI FASTA FILE INTO BED FILE, EXTRACTING THE SEQS FROM FASTA FILE BASED ON THE #CHROM ID IN THE BED FILE AND WRITING IT TO A NEW FILE, REMOVING THE EXTRACTED SEQS FROM THE FASTA FILE AND REWRITING TO ANOTHER FILE, REVERSE COMPLEMENTING THE EXTRACTED SEQS, INTRODUCING RANDOM 1 NUCLEOTIDE CHANGE TO THE EXTRACTED SEQS AND THEN ADDING IT BACK TO THE PRIMARY FILE*

*#converting the multi-fasta file into bed file*

**from** Bio **import** SeqIO  
**with** open(**"genome.bed"**, **'w'**) **as** bed\_file:  
 *# Write BED header* bed\_file.write(**"# chrom\tstart\tend\n"**)  
 **for** record **in** SeqIO.parse(**"genome.fasta"**, **'fasta'**):  
 seq\_id = record.id  
 seq\_length = len(record.seq)  
 bed\_file.write(**f'{**seq\_id**}\t0\t{**seq\_length**}\n'**)

*#reading the BED file, extracting sequences from a corresponding FASTA file based on the specified chromosome (#chrom), and writing the extracted sequences to a new FASTA file.*

**from** Bio **import** SeqIO  
**from** Bio.SeqRecord **import** SeqRecord  
  
  
extract = [**"gi|58576|gb|X52226|Influenza"**,**"gi|59292|gb|X53029|Influenza"**,**"gi|59294|gb|X59778|Influenza"**] sequences = []  
  
*# Reading the BED file and skipping the line that starts with # which is the header***with** open(**"genome.bed"**, **'r'**) **as** bed\_file:  
 **for** line **in** bed\_file:  
 **if** line.startswith(**"#"**):  
 **continue** fields = line.strip().split(**'\t'**)  
 bed\_chrom, start, end = fields[:3]

*# Checking if the chromosome is present in the extract list and then reading the fasta file to extract the sequences correspondingly* **if** bed\_chrom **in** extract:  
**with** open(**"genome.fasta"**, **'r'**) **as** fasta\_file:  
 **for** record **in** SeqIO.parse(fasta\_file, **'fasta'**):  
 **if** record.id == bed\_chrom:  
 sequence = record.seq[int(start):int(end)]  
 seq\_record = SeqRecord(sequence, id=**f"{**bed\_chrom**}\_{**start**}\_{**end**}"**)  
 sequences.append(seq\_record)  
  
*# Writing the extracted sequences to the output FASTA file(newgene.fasta)***with** open(**"newgene.fasta"**, **'w'**) **as** output\_file:  
 SeqIO.write(sequences, output\_file, **'fasta'**)

*#removing the extracted subsequences from the file and rewriting it to another file*

*# Accession numbers to be removed from the fasta file(genome.fasta)*

to\_remove = [**"gi|58576|gb|X52226|Influenza"**,**"gi|59292|gb|X53029|Influenza"**,**"gi|59294|gb|X59778|Influenza"**]

*# Open the input and output files and check if the accession number mention in the to\_remove list is present.If present remove it and rewrite the contents to new fasta file(genome\_removed.fasta)*

**with** open(**"genome.fasta"**, **'r'**) **as** input\_file, open(**"genome\_removed.fasta"**, **'w'**) **as** output\_file:**for** record **in** SeqIO.parse(**"genome.fasta"**, **'fasta'**):**if** record.id **not in** to\_remove:SeqIO.write(record, output\_file, **'fasta'**)

*#reverse complementing the extracted subsequences  
# Reading the sequences from the fasta file containing only the extracted sequences.*records = list(SeqIO.parse(**"newgene.fasta"**, **'fasta'**))  
  
*# Reverse complementing each sequence and writing it to the same file***for** record **in** records:  
 record.seq = record.seq.reverse\_complement()  
**with** open(**"newgene.fasta"**, **'w'**) **as** output\_file:  
 SeqIO.write(records, output\_file, **'fasta'**)

#introducing random one nucleotide change to the extracted seq

#reading the file containimg extracted seqs ,creating an output file (processed\_seq.fasta).Generating a random index within the file and selecting random nucleotides.Replacing the nucleotide at the random index and writing this to the output file.

**import** random  
  
**with** open(**"newgene.fasta"**, **'r'**) **as** infile:  
 **with** open(**"processed\_seq.fasta"**, **'w'**) **as** outfile:  
 **for** line **in** infile:  
 **if** line.startswith(**'>'**): outfile.write(line) **else**:  
 sequence = line.strip() index = random.randint(0, len(sequence) - 1) new\_nucleotide = random.choice([**'A'**, **'T'**, **'C'**, **'G'**]) mutated\_sequence = sequence[:index] + new\_nucleotide + sequence[  
 index + 1:]

outfile.write(mutated\_sequence + **'\n'**)

*##Insert the processed subsequences back to the primary sequence(multifasta).***from** Bio **import** SeqIOoriginal\_records = list(SeqIO.parse(**"genome\_removed.fasta"**, **'fasta'**))  
new\_records = list(SeqIO.parse(**"processed\_seq.fasta"**, **'fasta'**))modified\_records = original\_records + new\_records  
**with** open(**"genome\_removed.fasta"**, **'w'**) **as** output\_file:  
 SeqIO.write(modified\_records, output\_file, **'fasta'**)