First exercise Bioinformatics

Python code:

#1. Finding palindromes in a sequence using recursion

#Palindrome sequences are (as their greek name suggests) sequences that are read identically from both directions. Write a python script that will:

#a. define if a sequence contains a palindrome subsequence

#b. return the longest existing palindrome p given a longer sequence S

def make\_complement\_strand(DNA): #First a function must be developed for the definition of complement DNA strand

complement\_strand=[]

rules\_for\_complement\_strand={"A":"T","T":"A","C":"G","G":"C"} # Using dictionary function for the definition of complement DNA

for letter in DNA: # For loop

complement\_strand.append(rules\_for\_complement\_strand[letter]) # Using append function

return(complement\_strand)

make\_complement\_strand("ATCGATCTGC") # example

def is\_this\_a\_palindrome(DNA): # Second a function to define if a specific DNA sequence is a palindrome sequence

DNA=list(DNA)

if DNA!=(make\_complement\_strand(DNA)[::-1]): # By using the previous function we verify using an if statement if the sequence we have is a palindrome

print("false")

return False

else:

print("true")

return True

is\_this\_a\_palindrome("TATA") # Example 1

def find\_palindromes\_in\_seq(DNA): # Third function to search if there are any palindrome sequences in a strand

DNA=list(DNA)

i=0

palindrome = list(DNA[0: 5])"

i < len(DNA) # My problem to define this function is I have problem at defining the i in the list

for every i in DNA:

while i < len(DNA)

if\_this\_a\_palindrome(DNA[palindrome]) is True:

print(palindrome)

find\_palindromes\_seq("AGCTCGTCTAGC")