Luego de crear las funciones, cargue el módulo `miningscience` como `msc` e imprima documentación de la **función**.

In [1]:

```
# Escriba aquí su código para el ejercicio 1
import miningscience1 as msc
help(msc.dowload_pubmed)
help(msc.science_plots)
```

Ejercicio 2 [2 puntos]

Utilice dos veces la función `download_pubmed` para:

- Descargar la data, utilizando los keyword de su preferencia.
- Guardar el archivo descargado en la carpeta `data`.

Para cada corrida, imprima lo siguiente:

```
'El número artículos para KEYWORD es: XX' # Que se cargue con inserción de texto o valor que correspondea KEYWORD y XX
```

In [2]:

```
# Escriba aquí su código para el ejercicio 2
import miningscience-gol as msc
importre
myc_dato2 = msc.downbcd_pobmed("Buerger[TiTule/Abstract]")
conter = len(my_dato2)
print("El numero de articulos parc lc keyword (Buerger) es:", conter)
with open("data/Buerger.txt", "w") as txt:
        txt.write(my_dato2)
my_dato3 = msc.dowlo ed_pub med("Gastritis[Title/Abstract]")
    conter = len(my_data3)
print("El numer de articlos parcto keyword (Gastritis) es: ", conta)
with open("data/Gastritis.txt", "w") as txt:
    txt.write(my_dato3)
```

Ejercicio 3 [1.5 puntos]

```
data_Frame = msc.science_plots(my_dol2)
dolo-Frame1 - dola_ frone.sort_values(by=['Nomeo de autores'], oscending=False
dola_Frame2 =dalo_fronel.iloc [0:5]
dola_fione2
```

```
data_frone_odi = msc.science_plots(my_dotas)
dota_frone_odiL = data_frone_odi.sort_vdus (by=["Numero de autores"]
ascending = Folse)
dota_frone_odi2 = dolo_frone_odil.iloc [0:5]
dota_frone_odi2-
```

```
import matplotlib.pyplot as plt.
labels = 'Japon', 'France', 'USA', Turkey', 'Italy'
Sizes = [81, 25,24,21,19]
fig1, x1 = plt.subplots()
x1.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=10)
plt.title("Keyword-Buerge")
plt.savefig("img/autores_Buerger.jpg")
```

```
import matplotlib.pyplot as plt.
labels = 'china', 'Japon', 'Italy', 'USA', 'Germany'
    sizes = [3127, 1290, 905, 850, 252]
fig1, x1 = plt.subplots()
x1.pie(sizes, labels = labels, autopct='%1.1f%%', startangle=10)
plt.title("Keyword-Geotritis")
plt.savefig("img/autores_Geotritis.jpg").
```

④.

Grafico de la basqueda Keyword con Buerger se presenta que la mayor cantidad de autores son de Japon con un porcentaje de 47.6%. seguido de francia con un porcentaje de 14.7%.

Grafico de la busqueda Keyword con Gastritis se obtuvo que la mayor parte de autores son de China con un porcentaje de 48,7%. seguido de Japon con 20.1%. Por lo que se puede Concluir que japón es uno de las paises que posee una gran cantidad de autores que han estudiado enfermedades como la gastritis y la enfermedad de Buerger (KEYWORD).

```
from Bio.Phylo.TreeConstruction import DistanceTreeConstructor
from Bio.Phylo.TreeConstruction import DistanceCalculator
from Bio.Align.Applications import ClustalwCommandline.
from Bio import AlignIO
from Bio import Phylo
from Bio import Entrez
from Bio import SeqIO
import Bio
import warnings
warnings.filterwarnings('ignore')
import os
import matplotlib
import matplotlib.pyplot as plt
with open("data/sequence_enolasa.seq", "r") as f:
    seq_enolasa = f.read()
    seq_enolasaList = seq_enolasa.split('\n')
seq_enolasaListT = []
cont = 0
for i in seq_enolasaList:
    if cont < 15:
        seq_enolasaListT.append(i[:])
        cont += 1
Entrez.email = "qualaparo.moises@gmail.com"
ofile = open("data/sequence_enolasa.gb", 'w')
with Entrez.efetch(db="nucleotide", rettype="gb", retmode="text", id=seq_enolasaListT
    as handle:
    for seq_record in SeqIO.parse(handle, "gb"):
        ofile.write(">"+str(seq_record.description[:50])+'\n')
        ofile.write(str(seq_record.id)+'\n')
        ofile.write('\n')
ffile = open("data/sequence_enolasa.fasta", "w")
with open("data/sequence_enolasa.gb", 'r') as genbank:
    c = genbank.read()
    for line in c:
        ffile.write(str(line))

clustalw_exe = r"C:\Program Files (x86)\Clustal W2\clustalw2.exe"
clustal_cline = ClustalwCommandline(clustalw_exe, infile="data/sequence_enolasa.fasta")
assert os.path.isfile(clustalw_exe), "Clustal_W executable is missing or not found"
stout, stderr = Clustalw_cline()
ClustalAlign = AlignIO.read("data/sequence_enolasa.aln", "clustal")
```

```
calculator = DistanceCalculator ('identify')
distance_matrix = calculator.get_distance (ClustalAlign)
constructor = DistanceTreeConstructor (calculator)
Data_tree = constructor. buid_tree (ClustalAlign)
Data_tree.rooted = True
Phylo.write (Data_tree, "data/sequence_enolasa.xml", "phyloxml")

fig=plt.figure (figsize=(10,25), dpi=200)
matplotlib.rc ('font', size=12)
matplotlib.rc ('xtick', labelsize=10)
matplotlib.rc ('ytick', labelsize=10)
axes = fig.add_subplot (1, 1, 1)
Phylo.draw (Data_tree, axes=axes)
print ("\t\t\t\t\t\tFig 4. Árbol Filogenético enzima enolasa\n")

fig.savefig ("img/sequence_enolasa.jpg")
```

GBI6 – BIOINFORMÁTICA [2022I]
Examen Final [Python]

Universidad Regional Amazónica

Nombre [Apellido, Nombre]:

Construya las funciones del módulo miningscience.PY

```
def download_pubmed(Keyword ): from Bio import Entrez.
    """                        import re.
    Con esta función se buscara en pubmed los
    articulos asociados con la palabra que se colocara como Keyword, siendo esta la
    referencia al tema del que se hara la busqueda de los articulos.

    """
Entrez.email= "gualapuro.moises@gmail.com"
        handle = Entrez.esearch (db= "pubmed",
                                 term= Keyword,
                                 usehistory= "y"
        record = Entrez.read (handle)
        id_list = record ["IdList"]
        webenv = record ["WebEnv"]
        query_key = record ["QueryKey"]
        handle = Entrez.efetch( db= "pubmed"
                                rettype= "medline",
                                retmode= "text",
                                retstart = 0,
                                retmax= 1500,
                                webenv = webenv,
                                query_Key = query_Key)

        my_data = handle.read()
        my_data2= re.sub(r'\n\s{6}', ' ', my_data)

        return(my_data2)
```

GBI6 – BIOINFORMÁTICA [2022I]
Examen Final [Python]

Universidad Regional Amazónica

Nombre [Apellido, Nombre]:

```
import pandas as pd
import csv
import itertools.

def science_plots(my_data2                    ):
    """
```

Con esta funcion se va a leer la fuente que se ha empleado en el keyword para la busqueda en Pubmed, por medio deluso de expresiones regulares para delimitar cada posible variante de lectura, con el fin de que tome el pais y el autor de cada registro. Además genera un dataframe relacionando los coordenados y el pais con el uso de la data Coordenadas.txt / recalcando que luego se tendra que hacer un arreglo paraque se grafique solo los rs más abundantes

```
    """
    AD=[]
    paises1=[]
    paises2=[]
    paises3=[]
    paises4=[]
    paises5=[]
    paises6=[]
    paises7=[]
    paises8=[]
    paises9=[]

    paisesT=[]

for line in my_data2.splitlines():
    if line.startswith("AD -"):
        AD.append(line[:])
    for line in my_data2.splitlines():
        if line.startswith("AD -"):
            AD = line[:]

pais1 = re.findall(r'\,\s(\w{2,16})\.', AD)
    paises1.append(pais1)
    pais2 = re.findall(r'\,\s(\w{2,16}[^0-9],]\s\w{2,16}[^0-9])\.', AD)
    paises2.append(pais2)
```

```
pais3=re.findall(r'\,\s(\w{3,16}[^0-9\,]\s\w{2,3}[^0-9\,]\s\w
{3,16}[^0-9\,])\.', AD)
paises3.append(pais3)
pais4=re.findall(r'\,\s(\w{2,16})\.\s[a-z0-9_\.-]+@[\da-z\.-]+\.[a-z\.]{
2,6}', AD)
paises4.append(pais4)
pais5=re.findall(r'\j\s(\w{2,16}[^0-9\,]\s\w{2,16}[^0-9])\.\s[a-z0-9_\.-]
+@[\da-z\.-]+\.[a-z\.]{2,6}', AD)
paises5.append(pais5)
pais6=re.findall(r'\,\s(\w{3,16}[^0-9\,]\s\w{2,3}[^0-9\,]\s\w{3,16}[^0-9\,])
\.\s[a-z0-9_\.-]+@[\da-z\.-]+\.[a-z\.]{2,6}', AD)
paises6.append(pais6)
pais7=re.findall(r'\,\s(w{2,16})\.Electronic addres:\s[a-z0-9_\.-]+@[\da-z\.-]+
\.[a-z\.]{2,6}\.', AD)
paises7.append(pais7)
pais8=re.findall(r'\,\s(\w{2,16}[^0-9\,]\s\w{2,16}[^0-9])\.Electronic
adrdress:\s[a-z0-9_\.-]+@[\da-z\.-]+\.[a-z\.]{2,6}\.', AD)
paises8.append(pais8)
pais9=re.findall(r'\,\s(\w{3,16}[^0-9\,]\s\w{2,3}[^0-9\.]\s\w{3,16}[0-9\,]).
Electronic.addres:\s[a-z0-9-\.-]+@[\da-z\.-]+\.[a-z\.-]{2,6}\.', AD)
paises9.append(pais9)

paisesT=paises1+paises2+pabes3+paises4+paises5+paises6+paises7+paises8+paises9
paisesT=list(itertools.chain.from_Herable(pabesT))
len(paisesT)
unique_paisesT=list(set(paisesT))
unique_paisesT.sort()
len(unique_paisesT)

Impor csv
coordenadas={}
with open('coordenadas.txt')af:
    csvr=csv.DictReader(f)
for row in csvr:
    coordenadas[row['Nome']]=[row["latitud"],row
    ['longitud']]
pais=[]
lon=[]
lat=[]
c=[]
for z in unique_paisesT:
if z in coordenadas.key():
    pais.append(z)
    lat.append(float(coordenadas[z][0]))
    lon.append(float(coordenadas[z][1]))
```

```
C.oppend (paises T.count(z))
 tablaPaises T.count(z))
 tabla Paises = pd.DataFrame()

 tablaPaises ["pais"]=pais

 tablaPaises ["Numero de autores"]=C

 return(tablaPaises).
```