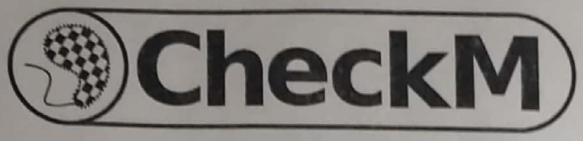

EXAMEN PARCIAL PYTHON

GBI6-2021II: BIOINFORMÁTICA

Apellidos, Nombres <--- CAMBIE POR LOS QUE CORRESPONDA A SUS DATOS
Angiely Camacho.
03-08-2022

Color de texto

REQUERIMIENTOS PARA EL EXAMEN

Utilice de preferencia `Jupyter` de Anaconda, dado que tienen que hacer un control de cambios en cada pregunta.

Para este examen se requiere dos documentos:

1. Archivo `miningscience.py` donde tendrá dos funciones:
2. Archivo `2022I_GBI6_ExamenPython` donde se llamará las funciones y se obtendrá resultados.

Ejercicio 0 [0.5 puntos] ✓

Realice cambios al cuaderno de jupyter:

- Agregue el logo de la Universidad
- Coloque sus datos personales
- Escriba una **tabla** con las características de su computador

Ejercicio 1 [2 puntos]

Cree el archivo `miningscience.py` con las siguientes dos funciones:

i. `download_pubmed` : para descargar la data de PubMed utilizando el **ENTREZ** de Biopython. El parámetro de entrada para la función es el `keyword` . -> Guardar

ii. `science_plots` : la función debe

- utilizar como argumento de entrada la data descargada por `download_pubmed`
- ordenar los conteos de autores por país en orden ascedente y
- seleccionar los cinco más abundantes. Con esta selección debe graficar un `pie_plot` . Como guía para el conteo por países puede usar el ejemplo de MapOfScience (https://github.com/CSB-book/CSB/blob/master/regex/solutions/MapOfScience_solution.ipynb).

iii *Cree un docstring para cada función.* ✓

Luego de crear las funciones, cargue el módulo `miningscience` como `msc` e **imprima docstring de** [illegible] **función.**

In [1]:

```
# Escriba aquí su código para el ejercicio 1
import miningscience
  "          "
help (msc.download_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 os
Import re
a= msc.download_pubmed("dogs")
b = len(a)
print ('El número artículo para KeyWORD es:',b)
with open ("Data/dogs.txt","w") as txt:
    txt.write(a)
```

Ejercicio 3 [1.5 puntos]

Utilice dos veces la función `science_plots` para:

- Visualizar un pie_plot para cada data descargada en el ejercicio 2.
- Guardar los pie_plot en la carpeta `img`

[4]:

```
# Escriba aquí su código para el ejercicio 3
import matplotlib.pyplot as plt
cats = msc.science_plots (a).
with open ("img/gatos.png", "w") as png

dogs = msc.science_plots (v)
plt.savefig ('img/perros.png', dpi = 500)
plt.show()
```

Ejercicio 4 [1 punto]

Interprete los resultados de las figuras del **ejercicio 3**

Escriba la respuesta del ejercicio 5 .

. Se realizo un arbol filogenético con los Accession list.

.

Ejercicio 5 [2 puntos]

Para algún **gen de las enzimas que intervienen en la ruta metabolica de la gluconeogenesis** (Lista de genes por tipología (https://www.genome.jp/pathway/map00010+C00068)), realice lo siguiente: ✓

1. Una búsqueda en la página del NCBI nucleotide (https://www.ncbi.nlm.nih.gov/nucleotide/). ✓
2. Descargue el `Accession List` de su búsqueda y guarde en la carpeta `data` .
3. Cargue el `Accession List` en este notebook y haga una descarga de las secuencias de los **quince primeros** IDs de la accesión.
4. Arme un árbol filogenético para los resultados del paso 3.
5. Guarde su arbol filogénetico en la carpeta `img`
6. Interprete el árbol del paso 4.

entrar a cuadritos con #

numero de accesion.

cargar notebook

usar ese #

In [3]:

```
# Escriba aquí su código para el ejercicio 6
with open ('Data/sequence.seq') as file:
    text = file.read()
text = text.split ('\n')
text = ','.join(text[:10])
handle = Entrez.efetch(db="nucleotide", rettype="gb", retmode="text", id=text)
(print(handle.url)
records = SeqIO.parse("Data/sequence.gb", genbank")
count = SeqIO.write(records, "Data/sequence.fasta", "fasta")
clustalw_exe= r"C:\Program file (x86)\Clustal W2\clustalw2.exe"
clustalw_cline= ClustalwCamandline (clustalw_exe, infile="Data/sequence.fasta")
assert os.path.isfile(clustalw_exe), "Clustal W executable is missing or not
found).
```

Escriba aquí la interpretación del árbol

Ejercicio 6 [1 punto]

1. Cree en GitHub un repositorio de nombre `GBI6_ExamenPython` .
2. Cree un archivo `Readme.md` que debe tener lo siguiente:

- Datos personales
- Características del computador
- Versión de Python/Anaconda y de cada uno de los módulos/paquetes y utilizados
- Explicación de la data utilizada
- Un diagrama de procesos del módulo `miningscience`

3. Asegurarse que su repositorio tiene las carpetas `data` e `img` con los archivos que ha ido guardando en las preguntas anteriores.
4. Realice al menos 1 control de la versión (commits) por cada ejercicio (del 1 al 5), con un mensaje que inicie como:

```
Carlitos Alimaña ha realizado el ejercicio 1

Carlitos Alimaña ha realizado el ejercicio 2

...
```

In [ ]:

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

Ikiam
Universidad Regional Amazónica

Nombre [Apellido, Nombre]:

Construya las funciones del módulo miningscience.py

```
def download_pubmed( Key word.                    ):
    """
    Este script permite boscar articulos en pubmed por medio
    de palabras claves. "

Entrez.email = 'guadapuro.moises@gmail.com'
busq = Entrez.read (Entrez.esearch (db = "pubmed",
                                    term = Keyword,
                                    usehistory = 'y"))
    """

webenv = busq ["WebEnv"]
query_key = busq [QueryKey]
handle = Entrez.efetch (db="pubmed",
                        rettype= "medline"
                        retmode = "text",
                        restart = 0,
                        retmax = 543, webenv=webenv, query_key = query_key)

data = hadle.read()
dataexp.re.sub (r'\n\s{6}', '', data)
return data exp.
```

Ikiam
Universidad Regional Amazónica

Nombre [Apellido, Nombre]:

```
def science_plots(
                                                    ):
    """

email = re.sub(r'\s[\w._%+-]+@[\w.-]+\.[a-zA-Z]{1,4}',",file)
    puntos = re.sub(r'\..\d.\,',',',email)
    numb = re.sub(r'\..\d.',",puntos)
    x=numb[1:].split('PMID-')

    Countries_A=[]
    for PMID in x:
        q=PMID.split('\n')
        for fila in q:
            w=fila.split(' ')
            if w[0] == 'AD':
                e=fila.split(',')
                Countries_A.append(e[-1])

    a=0
    Countries_B =[0]*len(Countries_A)
    for lis in Countries_A:
        bytes(lis,encoding="utf8")
        if lis != '':
            w=lis
            if w[0] == ' ':
                w = re.sub (r'^\s',",w)
            if w[-1] == '.':
                w = re.sub (r'\.$',",w)
            w = re.sub (r'\.$',",w)
            w = re.sub (r'\s$',",w)
        Countries_B[a]=w
        a=a+1

    Contries_all=[...]
email = re.sub(r'\s[\w._%+-]+@[\w.-]+\.[a-zA-Z]{1,4}',",file)
    puntos = re.sub(r'\..\d.\,',',',email)
    numb = re.sub(r'\..\d.',",puntos)
    x=numb[1:].split('PMID-')

    Countries_A=[]
    for PMID in x:
        q=PMID.split('\n')
        for fila in q:
            w=fila.split(' ')
            if w[0] == 'AD':
                e=fila.split(',')
                Countries_A.append(e[-1])
 a=0
    Countries_B =[0]*len(Countries_A)
    for lis in Countries_A:
        bytes(lis,encoding="utf8")
        if lis != '':
            w=lis
            if w[0] == ' ':
                w = re.sub (r'^\s',",w)
            if w[-1] == '.':
                w = re.sub (r'\.$',",w)
            w = re.sub (r'\.$',",w)
            w = re.sub (r'\s$',",w)
        Countries_B[a]=w
        a=a+1
```