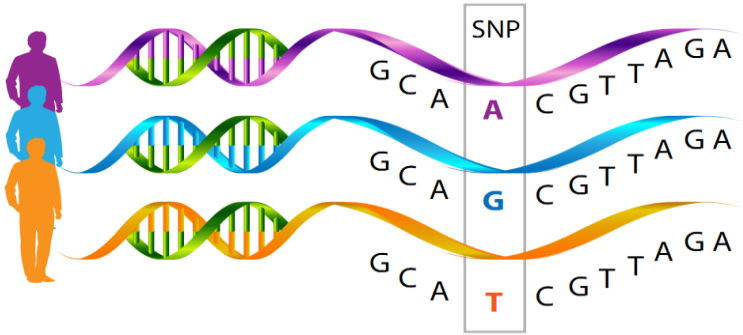
Python Final Project:

“Single nucleotide polymorphisms (SNP’s) extraction tool”

Ramon Cierco Jiménez

1. Introduction

Single Nucleotide Polymorphism (SNP) is a variation at a single position in DNA sequence (>1% population).



SNP's are related with several diseases like sickle-cell anemia, β-thalassemia and cystic fibrosis.

1. Introduction

- **Script name:** “get_snps_tool.py” 

- **Python version:** Python 3.5

- **OS:** Ubuntu 16.04

2. Program workflow

List of UniProt codes:
["P30953",...]

1

2

SNP:
[“dbSNP:rs1735011”, “129”, ”P”, ”L”]

Sequence:
“MMGQNQTSIS...”

3

4

Data frame:

- Rows: SNP’s

- Columns: information about them.

Index	UniProt	SNP	Original_aa	Changed_aa	Position	Sequence
0	Q8NHA8	NA	NA	NA	NA	MEGKNQTNISE…
1	O43749	dbSNP:rs183…	F	S	75	MSGTNQSSVSE…
2	O43749	dbSNP:rs804…	V	I	305	MSGTNQSSVSE…
3	P30953	dbSNP:rs173…	P	L	129	MMGQNQTSISD…
4	P30953	dbSNP:rs150…	A	T	143	MMGQNQTSISD…
5	P30953	dbSNP:rs379…	I	V	221	MMGQNQTSISD…
6	Q8NGI9	dbSNP:rs171…	F	L	103	MAVGRNNTIVT…
7	Q8NGI9	dbSNP:rs145…	P	L	172	MAVGRNNTIVT…
8	Q9Y585	dbSNP:rs560…	L	F	244	MKKENQSFNLD…
9	Q9Y585	dbSNP:rs224…	G	C	256	MKKENQSFNLD…
10	Q9Y585	dbSNP:rs246…	R	C	260	MKKENQSFNLD…
11	Q9Y585	dbSNP:rs121…	W	C	293	MKKENQSFNLD…
12	P47881	dbSNP:rs169…	S	G	78	MQPESGANGTV…
13	P47881	dbSNP:rs703…	A	D	120	MQPESGANGTV…

5

Save as .xlsx file:

6

Load from .xlsx file

3. Program structure

1

```
import os
import pandas as pd
import urllib
```

3 imported modules
3 created functions

2

```
def get_snps_info(code):
    """
    This function needs an UniProt code as input (variable code). Then, the function give as ouput a lists of lists with all
    the related SNP's, following this structure: [SNP name, Initial aa, Chaged aa, Position].

    Author: Ramon Cierco Jiménez
    Contact: ramoncierco7@gmail.com
    """
    data = str(urllib.request.urlopen("http://www.uniprot.org/uniprot/" + code + ".txt").read())
    if data.find('VARIANT      ')!=-1:
        p1 = data.find('VARIANT      ')
        data = data[p1:]
        p2 = data.find('SEQUENCE')
        data = data[:p2]    #between p1 and p2 we have all the information about SNP's, so now we will convert from str format to list
        data = data.split() #to filter the information
        pos_variant = []
        #here you add the intial position for each SNP using the position that matches with the string object 'VARIANT'
        for x in range(0, len(data)):
            tar = data[x]
            if tar.find('VARIANT')!=-1:
                pos_variant.append(x)
            else:
                continue
            del(x) ;del(tar)
        #Now for each position we know where the SNP information starts and we are able to extract it using the following loop
        raw_data_list = []
        for y in range(0,len(pos_variant)): #Here we are storing the information about each SNP (noise included)
            if y!=len(pos_variant)-1:       #as a list of lists (raw_data_list)
                tar_data = data[pos_variant[y]:pos_variant[y+1]]
                raw_data_list.append(tar_data)
            if y==len(pos_variant)-1:
                tar_data = data[pos_variant[y]:]
                raw_data_list.append(tar_data)
        raw_data_list; del(y); del(tar_data)
        final_list_snp = [] #Now we will keep the information we need to use: aminoacid change, their position and the SNP datbase name
        for z in range(0,len(raw_data_list)):
            tar_list = raw_data_list[z]
            tar_cor_list = []
            for b in range(0,len(tar_list)):
                if b<len(tar_list)-1 and tar_list[b]==tar_list[b+1]: # The aa change position is repeated twice in the HTML code
                    POS = tar_list[b]
                if tar_list[b]=='->':                               # The '->' separates the two aa (initial left, changed right)
                    AA1 = tar_list[b-1]
                    AA2 = tar_list[b+1]
                if tar_list[b].find('dbSNP')!=-1:                   # Where you find the dbSNP string you can asume that is the SNP db name
                    SNP = tar_list[b].replace(').\\nFT', '')
            # Finally you append the list of the target SNP to the final list and obtain the desired list of lists
            tar_cor_list.append(SNP);tar_cor_list.append(AA1);tar_cor_list.append(AA2);tar_cor_list.append(POS)
            final_list_snp.append(tar_cor_list); del(b); del(POS); del(AA1); del(AA2); del(SNP); del(tar_cor_list); del(tar_list)
        return(final_list_snp)
    else:
        return("NA") # some UniProt proteins have no SNP's, when it's the case the function will return the 'NA' string
```

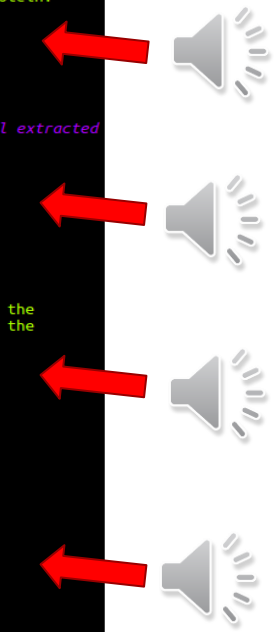
3. Program structure

3

```
def get_sequences(code):
    """
    This function needs an UniProt code as input (variable code). Then, the function give as ouput the sequence of the target protein.

    Author: Ramon Cierco Jiménez
    Contact: ramoncierco7@gmail.com
    """
    try:
        data = str(urllib.request.urlopen("http://www.uniprot.org/uniprot/" + code + ".txt").read())
        data = data[data.find('SEQUENCE   '):]
        seq_length = data[:data.find('AA;')].split()[-1] # We create this variable to control that the annotated sequence is well extracted
        data = data[data.find('\\n     '):]
        data = data.replace('\\n',''); data = data.replace('//n',''); data = data.replace(' ','')
        data = data.replace('//',''); data = data.replace("'",'')
        if int(len(data))==int(seq_length):  # Here we use the control variable that we created previously
            return(data)
    except:
        print("Something wrong, check if the UniProt code is correct or internet connection works properly")
```

4

```
def create_dataframe(dicc_of_snps, dicc_of_seqs):
    """
    This function needs two dictionaries that we should create with the functions 'get_snps_info()' and 'get_sequences()'. Then, the
    function is able to create a data frame object that contains information about the SNP's of the proteins that we targeted in the
    previous functions. The data frame follows this structure; SNP's as rows and Information as columns:

    UniProt  SNP              Original_aa   Changed_aa  Position   Sequence
    Q9Y585   dbSNP:rs12150427  W             C           293.0      MKKENQSFNLDFILLGVTSQ...

    Author: Ramon Cierco Jiménez
    Contact: ramoncierco7@gmail.com
    """
    dicc_keys = list(dicc_of_snps.keys())
    list_of_rows = [] # The pandas df function needs a list of tuples as the rows of the data frame
    for key in range(0, len(dicc_keys)): #This loop will select all the SNP's for each Protein
        target_key = dicc_keys[key]
        target_snps = dicc_of_snps[target_key]
        target_seq = dicc_of_seqs[target_key]
        if type(target_snps)==str and target_snps == 'NA': # In this case there are no SNP's
            row = (target_key, 'NA', 'NA', 'NA', 'NA', target_seq) # each row is a tuple
            list_of_rows.append(row)
            row = ()
        if type(target_snps)==list: # In this case there are at least one SNP
            for pos in range(0, len(target_snps)):
                target_snp = target_snps[pos]
                row = (target_key, target_snp[0], target_snp[1], target_snp[2], target_snp[3], target_seq) # each row is a tuple
                list_of_rows.append(row)
                row = ()
    del(pos); del(key); del(target_key); del(target_snps); del(target_snp); del(row); del(target_seq)
    list_of_labels = ['UniProt', 'SNP', 'Original_aa', 'Changed_aa', 'Position', 'Sequence'] #column labels
    dataframe = pd.DataFrame.from_records(list_of_rows, columns = list_of_labels)
    return(dataframe)
```



3. Program structure

5

6

7

8

9

```
# INTRODUCE THE TARGET LIST OF UNIPROT CODES

# Random list containing UniProt codes
list_of_uniprot_codes = ["Q8NHA8","P30953","P47881","Q8NGI9","Q9Y585","O43749","P30953"] # Here you have to change the target proteins

#########################################################################################################################################

# GET SNPS INFORMATION
# Execute de function get_snps_info for all the given proteins using a loop
dicc_of_snps = {}
list_errors = []
for pos in range(0,len(list_of_uniprot_codes)):
    target_code = list_of_uniprot_codes[pos]
    try:
        snp_list = get_snps_info(target_code)
        dicc_of_snps[target_code] = snp_list
    except:
        list_errors.append(target_code)
del(pos); del(target_code); del(snp_list) # remove the created variables inside the loop

#########################################################################################################################################

# GET PROTEIN SEQUENCE FROM UNIPROT
# Execute de function get_sequences
dicc_of_seqs = {}
for pos in range(0,len(list_of_uniprot_codes)):
    target_code = list_of_uniprot_codes[pos]
    seq = get_sequences(target_code)
    dicc_of_seqs[target_code] = seq
del(pos); del(target_code); del(seq) # remove the created variables inside the loop

#########################################################################################################################################

# CREATE FINAL DATA FRAME OBJECT
# Create a data.frame object to store the obtained SNP's information
df = create_dataframe(dicc_of_snps, dicc_of_seqs)

# Select and change working directory where you want to save the file
os.chdir("/home/ramon/Escritorio/Arnau_Cordomi/practical")

# # Store the data.frame object as an excel file
writer = pd.ExcelWriter('SNP_dataframe.xlsx')
df.to_excel(writer,'Sheet1')
writer.save()

#########################################################################################################################################

# READ THE DATA FRAME OBJECT FROM THE CREATED EXCEL FILE
# Read the created data frame (Extra)
dataframe_from_file = pd.read_excel("SNP_dataframe.xlsx", "Sheet1")
```

4. Example

Random UniProt codes:
Q8NHA8, P30953, P47881, Q8NGI9, Q9Y585, O43749, P30953

Index	UniProt	SNP	Original_aa	Changed_aa	Position	Sequence
0	Q8NHA8	NA	NA	NA	NA	MEGKNQTNISEFLLLGFSSWQ…
1	O43749	dbSNP:rs1834026	F	S	75	MSGTNQSSVSEFLLLGLSRQP…
2	O43749	dbSNP:rs8045183	V	I	305	MSGTNQSSVSEFLLLGLSRQP…
3	P30953	dbSNP:rs1735011	P	L	129	MMGQNQTSISDFLLLGLPIQP…
4	P30953	dbSNP:rs150989	A	T	143	MMGQNQTSISDFLLLGLPIQP…
5	P30953	dbSNP:rs379856	I	V	221	MMGQNQTSISDFLLLGLPIQP…
6	Q8NGI9	dbSNP:rs17153691	F	L	103	MAVGRNNTIVTKFILLGLSDH…
7	Q8NGI9	dbSNP:rs1453547	P	L	172	MAVGRNNTIVTKFILLGLSDH…
8	Q9Y585	dbSNP:rs56058341	L	F	244	MKKENQSFNLDFILLGVTSQQ…
9	Q9Y585	dbSNP:rs2241091	G	C	256	MKKENQSFNLDFILLGVTSQQ…
10	Q9Y585	dbSNP:rs2469791	R	C	260	MKKENQSFNLDFILLGVTSQQ…
11	Q9Y585	dbSNP:rs12150427	W	C	293	MKKENQSFNLDFILLGVTSQQ…
12	P47881	dbSNP:rs16952828	S	G	78	MQPESGANGTVIAEFILLGLL…
13	P47881	dbSNP:rs703903	A	D	120	MQPESGANGTVIAEFILLGLL…

	A	B	C	D	E	F	G	H
1		UniProt	SNP	Original_aa	Changed_aa	Position	Sequence	
2	0	Q8NHA8	NA	NA	NA	NA	MEGKNQTNISEFLLLGFSSV	
3	1	O43749	dbSNP:rs1834026	F	S	75	MSGTNQSSVSEFLLLGLSRQ	
4	2	O43749	dbSNP:rs8045183	V	I	305	MSGTNQSSVSEFLLLGLSRQ	
5	3	P30953	dbSNP:rs1735011	P	L	129	MMGQNQTSISDFLLLGLPI	
6	4	P30953	dbSNP:rs150989	A	T	143	MMGQNQTSISDFLLLGLPI	
7	5	P30953	dbSNP:rs379856	I	V	221	MMGQNQTSISDFLLLGLPI	
8	6	Q8NGI9	dbSNP:rs17153691	F	L	103	MAVGRNNTIVTKFILLGLS	
9	7	Q8NGI9	dbSNP:rs1453547	P	L	172	MAVGRNNTIVTKFILLGLS	
10	8	Q9Y585	dbSNP:rs56058341	L	F	244	MKKENQSFNLDFILLGVTS	
11	9	Q9Y585	dbSNP:rs2241091	G	C	256	MKKENQSFNLDFILLGVTS	
12	10	Q9Y585	dbSNP:rs2469791	R	C	260	MKKENQSFNLDFILLGVTS	
13	11	Q9Y585	dbSNP:rs12150427	W	C	293	MKKENQSFNLDFILLGVTS	
14	12	P47881	dbSNP:rs16952828	S	G	78	MQPESGANGTVIAEFILLG	
15	13	P47881	dbSNP:rs703903	A	D	120	MQPESGANGTVIAEFILLG	

Pandas

Data frame

Excel table

5. Difficulties

- **Extracting information from UniProt using “urllib”.**

- **Merge all the functions inputs/outputs together.**

Thanks for your attention!!

Ramon Cierco Jiménez