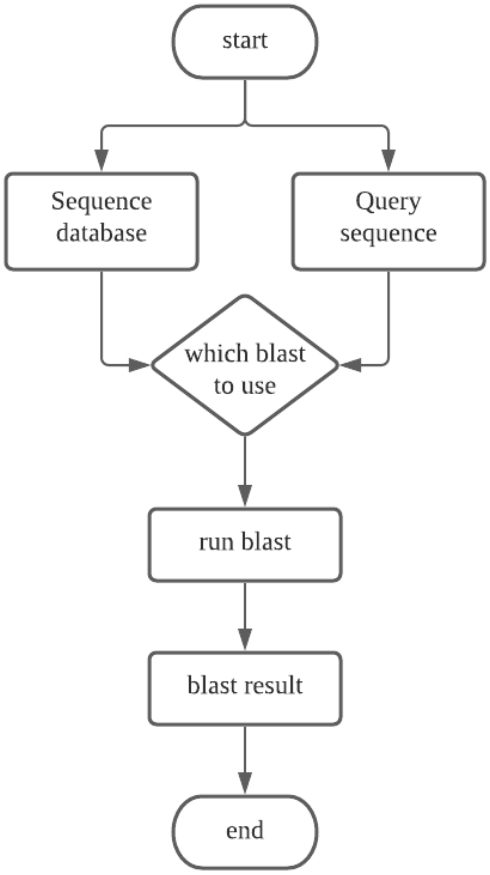
THE UNIVERSITY *of* EDINBURGH
School of Biological Sciences

[Bioinformatics Programming and System Management]
[Final Exam]

Student Exam Number: B193818

EXAM

Workflow



The main workflow is work as previous.

First, the program will ask the user which type of sequence that they use as database and the path of sequences saved in FASTA format. This could be done by using the `input` function in Python3. User can provide another input while they input an invalid parameter.

```
#!/usr/bin/python3
from Bio import Entrez, SeqIO
import os
import subprocess
Entrez.email = "s123456@ed.ac.uk"
Entrez.api_key=subprocess.check_output("echo       $NCBI_API_KEY",
```

```
shell=True).rstrip().decode()

while True:
    db_type = input('Please input the database type:')
    if not db_type:  # handle empty input
        print('Empty input!')
        continue
    elif db_type not in ['nucl', 'prot']:
        print('Should be one of "nucl" or "prot"!')
        continue
    break

while True:
    db_fasta = input('Please input the database fasta path:')
    if not db_type:  # handle empty input
        print('Empty input!')
        continue
    elif not os.path.isfile(db_fasta):
        print('{} is not a valid file!'.format(db_fasta))
        continue
    break
```

After received the user input, the program should check the input, for example wether the sequence type is `nucl` for nucleoside or `prot` for protein and the input FASTA exist. The sequence type of query and FASTA file path of query should be the same procedure.

Next, the program will make a BLAST database using the database FASTA file. This can be done by using the `subprocess` module.

```
args = ['makeblastdb', '-in', db_fasta, '-dbtype', db_type, '-out',
db]
subprocess.check_call(args)
```

Then, the program should determine which BLAST command should be used by the database sequence type and query sequence type.

```
BLAST_COMMANDS = {
    'nucl': {'nucl': 'blastn', 'prot': 'tblastn'},
    'prot': {'nucl': 'blastx', 'prot': 'blastp'}
}
command = BLAST_COMMANDS[db_type][query_type]
```

```
if db_type == 'nucl' and query_type == 'nucl':
    while True:
        translate = input('Do you want to translate both database
and query into protein?')
        if translate == 'y' or translate == 'yes':
            command = 'tblastx'
        break
```

The choices of BLAST command can be store in a two level dictionary which the first level key is the database sequence type and the second level key is quey sequence type. So the program can determine the BLAST command by access value from the dictionary. Except, when the database sequence and query sequence type are both nucleotide, the program should ask user whether they wanna to translate all the sequence into protein then do the search. That means the program should use `tblastx`.

Finally, the program will perform the BLAST search, this can be done jsut like before by using `subprocess`.

```
args = [
    command, '-db', db, '-query', query
]
with open(result, 'w') as stdout:
    subprocess.check_call(args, stdout=stdout)
```

The result of BALST analysis will output from stdout, the program will catch that and save it to a file.

Details

Beside the main flow, there are several details worth considering:

1. Automatic download sequence from NCBI Entrz database using `edirect` package. This will need user to input some conditions to search the database and download sequence.
2. Automatic determine the sequence type, this might be possible through a sequence type alphabet.
3. Support more BLAST parameters.
4. Maybe more according to the demands ...