Finding Closest Pair of Sequences

Template Executed by: Tony Kabilan Okeke

Write your function in a file named `"ptns_closestpair.py"` , which is imported in this notebook.

You do not need to change anything in this Jupyter notebook. Run this file to produce the outputs. Then save it as a PDF. Submit both your PDF file, as well as ptns_closestpair.py file on Blackboard.

In [ ]:
```
%load_ext autoreload
```

In [ ]:
```
# Imports
%autoreload 2
import bmes

from Bio import SeqIO
from ptns_closestpair import ptns_closestpair
```

Test 1

In [ ]:
```
ptns = ['ANNA','ALLIE','HANNAH']

result = ptns_closestpair( ptns )
print(result)
```

```
{'pair': [1, 3], 'ident': 100}
```

Test 2

In [ ]:
```
ptns = ['AHMET','AMY','EMILY'];

result = ptns_closestpair( ptns );
print(result)
```

```
{'pair': [2, 3], 'ident': 50}
```

Test 3

In [ ]:
```
file = bmes.downloadurl('https://sacan.biomed.drexel.edu/lib/exe/fetch.php?rev=&media=course:binf:data:uteroglobin.blas
ptns = [str(fastaptn.seq) for fastaptn in SeqIO.parse(file,'fasta') ]

# Remove any duplicates
ptns = list(set(ptns))
```

In [ ]:
```
# Select first 10 proteins in list because it would take too long to iterate throug
# every possible pair of proteins
ptns = ptns[:10]

result = ptns_closestpair( ptns );
print(result)
```

```
{'pair': [4, 9], 'ident': 80}
```

Appendix

In [ ]:
```
# Printing the file here as well for easy reference when grading.
from pathlib import Path
txt = Path('ptns_closestpair.py').read_text()
print(txt)
```

```
# Author: Tony Kabilan Okeke <tko35@drexel.edu>
# Date: February 7, 2022

# Imports
from itertools import combinations
from Bio.Align import substitution_matrices
from Bio import pairwise2

def ptns_closestpair(ptns: list):
  """
  This function finds the most similar pair in a list of proteins
  based on their local alignment scores using a BLOSUM62 scoring matirx.
  It returns a dictionary containing the indicies of items in the pair,
  as well as the percent identity of the alignment.

  Parameters
  ----------
  ptns: list
    A list of at least 2 proteins
  """

  # Remove any gap characters from proteins
  for i in range(len(ptns)):
    ptns[i] = ptns[i].replace('-', '')

  # Create list of unique pairs (combinations)
  pairs = [pair for pair in combinations(ptns, 2)]

  # Load and store substituotion matrix
  subs_mat = substitution_matrices.load('BLOSUM62')

  # Loop through pairs
  max_score = 0
  for pair in pairs:
    # Compute alignment score
    score = pairwise2.align.localds(
      pair[0], pair[1], match_dict=subs_mat, open=-5, extend=-5,
      one_alignment_only=True, score_only=True
    )

    # Assign new maxiumum and identify the corresponding pair
    if score > max_score:
      max_score = score
      most_similar = pair

  # Compute the alignment for the most similar pair
  align = pairwise2.align.localds(
    *most_similar, match_dict=subs_mat, open=-5, extend=-5,
    one_alignment_only=True
  )[0]

  # Keep only characters that are part of the alignment
  aligned_seqA = align.seqA[align.start:align.end]
  aligned_seqB = align.seqB[align.start:align.end]

  # Count matches in the aligned sequences
  total = 0
  for i in range( len(aligned_seqA) ):
    if aligned_seqA[i] == aligned_seqB[i]:
      total += 1

  # Compute percent identity and store value as an integer
  pct_identity = int( total / len(aligned_seqA) * 100 )

  # Get indices for items in pair
  ind = []
  for ptn in most_similar:
    # Indexes are incremented to return matlab style indexes
    ind.append( ptns.index(ptn) + 1 )
  ind.sort()

  return {'pair': ind, 'ident': pct_identity}
```