**Part 2 – Develop an algorithm to de-multiplex the samples**

Write up a strategy for writing an algorithm for de-multiplexing files and reporting index- hopping. That is, given four files (2 with biological reads, 2 with index reads) and known indexes, sort reads by index, outputting one forward file and one reverse file per index, plus a pair of files for unknown indexes. Additionally, your algorithm should report the number of properly matched indexes (per index) and the level of index hopping observed. You should not write any code for this portion of the assignment. Be sure to:

* **Define the problem**

Due to the nature of high throughput and multiplexing capacities of NGS, index misassignment between multiplexed libraries can potentially lead to misalignment and inaccurate results downstream.

* **determine/describe what output would be informative**

Separating out true index assignment from incorrect, quantifying levels of index swapping by plotting both data (before and after demultiplexing - Ilumina’s white paper), keeping only high quality reads according to part 1 threshold.

* **Write examples (unit tests):**

**Include four properly formatted input fastq files**

**Include the appropriate number of properly formatted output fastq files**

* **Develop your algorithm using pseudocode**
* **Determine high level functions**

**Description**

**Function headers**

**Test examples for individual functions**

**Return statement**

\*Information obtained from technician doing the sequencing

Goal: to generate 48 fastq files (with correctly assigned barcodes by evaluating each read for both reverse and forward barcodes). Whatever doesn’t match to anything or have a single undetermined base call (N) on the barcode sequence, redirect to 2 unknown fastq files (reverse and forward unknown) to verify % of index swapping occurring. Also, report low quality reads (less than 36 for sequence read and less than 37 for barcode, according to part 1 of assignment).

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# since the goal is to output information that returns counter for matching, incorrect library match, low quality score and N base call, set counter to zero for each :

**matching\_correct, index\_swapping, low\_qual, undetermined**

FastQ structure

(header)

(seq)

(plus)

(qual\_score)

# 1. Name files and load into python:

fw\_read = 1294\_S1\_L008\_001.fastq (forward read)\*

rv\_read = 1294\_S1\_L008\_004.fastq (reverse read)\*

fw\_barcode = 1294\_S1\_L008\_002.fastq (forward barcode)\*

rev\_barcode = 1294\_S1\_L008\_003.fastq (reverse barcode)\*

# 2. create a dictionary for all 24 known/expected barcodes in the format below:

Bar\_dict[ ]

Key value

B1 GTAGCGTA

A5 CGATCGAT

C1 GATCAAGG

B9 AACAGCGA

C9 TAGCCATG …

Test unit will verify if both key and value relationship is correct:

Assert bar\_dict [B1] = GTAGCGTA

(also, choose a key that is not present in dictionary to verify output)

# 3. Define convert\_phred function

Def convert\_phred(letter)

Qual\_score = ord (letter) - 33

Return (qual\_score)

Test unit will verify correct conversion according to ASCII parameters; test for letter with different ranges (low, medium, high, outside of range for phred scores)

example:

Assert convert-phred (A) = 32

\*\* I’m not sure on where to place this function in the script. Any suggestions?

# 4. First, I want to parse through fw and rv barcode files, count files with unknown base call (N) and store under unknown fastq fw and rv separately. Also, I want to know how many barcodes have low quality

Example for barcode sequences:

For line in file fw\_barcode

Strip lines into separate units (head, seq, plus, qual\_score)

If “N” in seq, add to undetermined counter

Else, continue to next line

Set qual\_threshold == 36

If qual < qual\_threshold, add to low\_qual counter

Else, continue to next line

Break when done reading through lines

Return counts for undetermined and low\_quality barcodes

For line in file fw\_read

Strip lines into separate units (head, seq, plus, qual)

Set qual\_threshold == 37

If qual < qual\_threshold, add to low\_qual counter

Else, continue to next line

Break when done reading through lines

Return counts for low\_quality reads

# Should I reverse one of the read sequences to find potential matches? Which one? Verify directionality (5 → 3)

# 5. Find out how many barcodes in my dictionary match my fw and reverse barcode? Place those that do not match under undetermined fw and rv.

For line in file fw\_barcode

Strip lines into separate units (head, seq, plus, qual)

If seq in bar\_dict, find in fw\_read seq line

If barcode seq in bar\_dict and fw\_read seq line, write to new fastq and add to matched counter

Else, add to undetermined count and undetermine\_fw fastq

If seq not in bar\_dict, move to next head line

Break when no head line found

Return counter for undetermined\_fw, undetermined\_rv

#how to make sure they are uniquely directed to new fastq files? One for each fw and rv barcode and sequence matching (48 total)? I am struggling to conceptualize the fact that python will be reading the lines and after reading header and starting to look at the sequence line, that’s when it will make the decision of where to place the 4 lines as a group for that particular record.

# output information returns counter for : matching\_correct, index\_swapping, low\_qual, undetermined; as well as 48 matching fastq files and 2 fastq (for fw and rv separately), containing libraries that have index swapping, low quality and/or undetermined base call.