De-Duplication. PCR dup removal

Define

- INPUT: A sorted sam file with uniquely mapped reads
- OUTPUT: the same sorted sam file, with all PCR duplicates removed
- Description: WIll remove the PCR duplicates that are identifiable from the same 5 prime starting positions, strandedness, and UMI code.

- Same alignment position
  - Chromosome
  - 5' start of read
  - Strand (strand specific?)
- Same Unique Molecular Index (UMI or "randomer")

RNAME (SAM col 3)

POS (SAM col 4) + **CIGAR** (SAM col 6)

FLAG (SAM col 2)

QNAME (SAM col 1)

Soft clipping - When the starting position of the read is changed because part of the read did not align due to error. This is important because two duplicate PCR reads can now have different starting positions, but still be duplicates and need to have one removed.
Solution - Remove the soft clipping value from the start position to get the real full 5 prime start position to see if it aligns with other 5 prime positions.

Soft outline needs

- A dictionary for storing each start starting position,
  - and a way to clean it to keep memory down
  - Maybe have a buffer of a certain size, and clear it when it fills
- A way to store 1 of each pair exactly
- Function to get each lines information about: strandedness, chromosome
- Function to see if soft clipped from the cigar string, and find 5 prime starting position
- Function for checking UMI
- Core logic to go over each sam entry, parse input, check dictionary, and store if its not duplicate

What to do when the sequences are different, just take the first?
Does length matter? Why not?
So does soft clipping on the end matter?
UMI not in list?

PSEUDO CODE —----------
Make dictionary of {5prime start, UMI, strand, chrom}
Make List of the UMI that are known

Skip headers
Current pos = 0
For each line

Get UMI from header
Determine in UMI is in known UMI list, if not then go next
New Current pos = position in the sam line
Call function: Calc 5 prime position with the CIGAR string and current pos
Grab strand and grab chrom from line

Check if these are in the dictionary,
    if so then move on
    If not, then add them and write to the read to file

If new current pos is greater then read length + last current pos
    Remove entries from dictionary to keep memory low

FUNCTIONS —----------

5_PRIME_START
—----------
INPUT: cigar str : STR, start pos: STR
OUTPUT: 5 prime start: STR
DESCRIBE: This function will determine if there is soft clipping, and if so, will change the given start position by that amount to align with the true 5 prime start.

Def 5_prime_start(cigar_str: STR, start_pos: STR) -> INT:

Examples:
- Input: ‘5S95M’, ‘10004000’
- Output: 10003995

- Input: ‘100M’, ‘1000’
- Output: 1000

—----------