Problem Statement

We have genetic annotation data in a file that we need to parse and query by user-provided input.

We need a functions to accomplish the following goals:

1. Load the data from external file and populate a data structure
2. Search the data structure by specific parameters
3. Provide a simple summary report of loaded annotation data

1. Load/parse the data into a structure

Write a function to read input files like `data_samples.25rows.bed.txt` with the following format: tab delimited, 5 columns wide, with columns as below

column number	name	column spec
1	chrom	string value like 'chrXX' where XX is integer from 1 - 22. Drop the 'chr' prefix when loading.
2	start position	integer value from 1 - 2 ** 32, must be > 0
3	end position	integer value from 1 - 2 ** 32, must be > 0 and > start position
4	feature name	string value from character set of: alphanumeric, underscore, hyphen, parentheses
5	strand	string value of '-' or '+'

Validate the inputs according to each column spec. If any column contains entries that do not agree with the column spec above, raise a `ValueError`. Populate a data structure of your choosing to store the information in `data_samples.25rows.bed.txt` for use in subsequent functions.

2. Implement search/query functions

- Provide 2 varieties of search function that operate on the data structure you initialized:
  - **search by position**
    - accept data structure, chrom value and optional position(s) as input
    - return matching records from data structure that occur on chrom, with (start position <= position < end position) if position is provided
    - return None if no matches
    - raise `ValueError` if the chromosome value does not have appropriate format (e.g., "chr6")
    - raise `ValueError` if the positional input is not an integer
  - **search by feature name**
    - accept data structure and feature name as string value input
    - return all records whose feature value matches the feature inputs exactly
    - return None if no matches
  - **perform search query**
    - select records on chromosome 6 between positions 169000000 and 170000000
    - select records with feature name "adsf1234qwer0987"

3. Test your code from part 2

- Use assert statements to write unit tests for your implementation of `search_by_position` from part 2. Be sure to test each condition in the specification!

NB: there is no need to use unit testing boilerplate (`pytest` , `nose` , etc) here; a simple `assert` statement will suffice. For example to test the simple function `def add(x, y): return x + y` An acceptable format for a unit test for the purposes of this quiz would be `assert add(1, 2) == 3`

4. Report simple data summary statistics

Each record in the data structure has two key attributes:

- location - defined as chromosome number and start position
- length - defined as (end position - start position)
- strand - either + or -, per the strand column

In the cells below, calculate and print some summary statistics on the features loaded from file:

- For each chromosome loaded:
  - the number of records found

- the number of records found per strand
- Min, Max, and average lengths of the records

You may use either the `pandas`, `numpy` or `math` libraries, but no others.