Bioinformatical problem solving with Python

Wednesdays 17:30-19:00, M801

alexander.nater@uni-konstanz.de

Exercises

- Download the vcf file ‘Apo_cl2_scaffold2_5Mb.vcf’ from Github.
- The vcf file contains phased genotype data over 5 Mb on scaffold2 for 10 indiviudals (no missing data).
- Calculate mean heterozygosity for each individual and print the values together with the individual names to a text file.
- Get counts for each unique haplotype in the region scaffold2:2’000’000-2’020’000.

Searching strings

- Testing if substring contained in string:
  string = “This is a string.”
  substring = “his”
  if substring in string:
      print(“Substring”, substring, “present in”, string)

- Testing if string starts with substring:
  string.startswith(substring) → False
  string.startswith(“This”) → True

- Finding the position (index) of first match of substring in string:
  string.find(substring) → 1

Exercises

- Given the sequence string "ACTAGCGTTTACT“, extract the substring ‘GTT’ together with the leading and trailing base (i.e. in this case ‘CGTTT’).
- Write a function that takes a string and a substring and returns a list of starting indices for each occurrence of the substring in the string.

Regular expression

- Often we want to search a specific pattern rather than an exact substring. → Use regular expressions (regex)
- A regular expression is a pattern against which strings are matched.
- Functionality for regex searches in the ‘re’ module of the Python standard library:
  import re
- re.match and re.search methods to search pattern at start of string or in entire string, respectively.

Search patterns

char	meaning
^	beginning of string
$	end of string
.	any character except newline
|	alternative
()	grouping / storing
[]	set of characters
[^]	set of excluded characters

repetition	example
*	match 0 or more times
+	match 1 or more times
?	match 0 or 1 times
{m}	match exactly m times
{m,n}	match between m and n times
{m,}	match at least m times
{,n}	match at most n times

character classes	meaning
\w	alphanumeric and underscore
\W	non-alphanumeric
\s	whitespace
\S	non-whitespace
\d	decimal digit
\D	non-digit

Some examples

regex	matches	doesn't match
[ab]+bc	**bbbc**	bc
[CG]{5}	**CCCGG**TT	GGGTT
^abc	**abc**def	xabc
^a.*e$	**ae**	abc
^a.*e$	**abcde**	abcdef
^$	<empty string>	<anything else>
^[^P]	start with anything *except* P	
^A.*E$	matches entire string if it starts with A and ends with E	
T(AA|AG|GA)	**TAA**	TGG

Match objects

- re.match and re.search return match object or None.
  mobj = re.search(pattern, string)

- mobj.start() to get the index of the match.

- string[mobj.start(): mobj.end()] or mobj.group(0) to get the matched substring.

- mobj.group(groupid) to get matches in capture groups:
  string = “The current local time in London is 17:30.”
  mobj = re.search(‘in\s+(\w+)\s+is\s+(\d+):(\d+)’, string)
  location = mobj.group(1)
  hours = mobj.group(2)
  minutes = mobj.group(3)

- Download the vcf file ‘Apo_cl2_scaffold2_5Mb_stats.vcf’ from Github.
- This vcf file contains a complete header (lines starting with ‘##’).
- Each variant site has now an INFO field with different statistics.
- Expand your previous script to filter variant sites by the DP (>1000) and AC (>10) statistic. Calculate mean individual heterozygosities for the filtered sites only.