UNIVERSITÄT DES SAARLANDES

ZBI ZENTRUM FÜR BIOINFORMATIK

Assignment 01

Programming with Python (for Bioinformatics)

Johanna Schmitz

24.04.2024; Abgabe **07.05.2024** vor 23:59

Assignment 01

Tests

To grade your assignments, we will run automated private tests. You will get points based on the number of tests that you passed.
For each exercise, we provide one exemplary test in the git repository that you can use to test your implementation and check that your implementation conforms to the specification.
For more information about testing, check out the tutorial `02-tests`.

Assignment 01

Task 1: Write a function `process_list` (2 points)

Write a function `process_list` that modifies each element in a list and returns the maximum value of the modified list.

Specification

- Name: `process_list`
- Parameter: list of integers
- Return value: largest number in the list after modification

Modification:

1. If an elements is
   - odd, multiply the element by 2;
   - even, divide the element by 2.
2. If the position in the list is a multiple of 7, add the position to the value.

Assignment 01

Task 2: DNA Reverse Complement (2 points)

Write a function `DNA_complement` that takes a string as input and computes its reverse DNA complement (`A` $\leftrightarrow$ `T`, `C` $\leftrightarrow$ `G`).
For example, `ACGATCGATCGATTC` $\leftrightarrow$ `GAATCGATCGATCGT`.
You can assume that the input string only contains the upper case letters `A,C,G,T`.

Assignment 01

Task 3: Extracting k-mers (substrings of length k) from a DNA sequence (3 points)

1. Write a function `list_kmers` that, given a DNA sequence s and an integer k, returns a list of all k-mers (strings of length k, from left to right, overlapping). Handle edge cases correctly ($k > |s|$?!).
2. Write a function `number_of_unique_kmers` that, given s and k, returns the number of unique k-mers in s (k-mers that appear only once).

Assignment 01

Task 4: Integer encoding of k-mers (3 points)

For a given k, there are 4^k different possible k-mers. Let's encode every k-mer bijectively as an integer $0 \leq c < 4^k$.

Use this encoding: Convert `A` $\mapsto 0$, `C` $\mapsto 1$, `G` $\mapsto 2$, `T` $\mapsto 3$.

Interpret the sequence of digits as an integer with base 4.

Example: `ACGT` $\mapsto (0,1,2,3)_4 = 0 \cdot 4^3 + 1 \cdot 4^2 + 2 \cdot 4^1 + 3 \cdot 4^0 = 16 + 8 + 3 = 27$.

- Write a function `kmer_code(kmer)` that returns the integer encoding of the string `kmer` using **bit operations**.
- Write a function `kmer_decode(code, k)` that returns the correct string for a given integer encoding and k-mer length k using **bit operations**.

Hints:

- Your are not allowed to use any packages, like numpy, in this exercise.