2024-10-16

Лабораторная работа 3: Элементарный алгоритм сжатия, реализованный через классы

1. Получить случайную последовательность ДНК из 1000 нуклеотидов на Random DNA Sequence (https://www.bioinformatics.org/sms2/random_dna.html)
2. Представить каждый уникальный нуклеотид (Нуклеотиды — Википедия) как бинарное число (подсказка: хватит 2 битов)
3. Реализовать класс, содержащий метод сжатия (упаковки), распаковки и строкового представления хранимых данных (в распакованном виде). При этом метод сжатия должен быть `скрытым/приватным`, что должно быть отражено в принципе его именования. Он не должен вызываться напрямую, его вызов осуществляется при инициализации экземпляра класса.
4. Экземпляр класса при инициализации должен принимать на вход строковое (`str`) представление последовательности нуклеотидов.
5. Хранение сжатой последовательности должно осуществляться при помощи бинарного целого числа (`int`, `0b00`)
6. При битовых операциях использовать операцию битового сдвига влево `<<=` и битовую операцию `OR` `|=` (см. подробнее о битовых операциях, например, здесь: BitwiseOperators - Python Wiki, Bitwise Operators in Python – Real Python)
7. Метод сжатия можно реализовать через последовательность `if...elif..else` или применить `match`
8. При распаковке вам понадобится битовый сдвиг вправо `>>` и операция обратного среза (`[::-1]`) или метод `reverse()`
9. При запуске программы выводить оригинальный и сжатый размер последовательности в памяти через `sys.getsizeof()`
10. Протестировать работу программы на последовательностях из `1000`, `10_000`, `100_000`, `1_000_000` и `10_000_000` нуклеотидов.

```
import sys

original_dna_str = 'gggctgcgtgcctccgga'
compressed_dna_str = CompressedDNA(original_dna_str)
```

```
print(f'Исходная строка: {sys.getsizeof(original_dna_str)} байтов')
print(f'Сжатая строка: {sys.getsizeof(compressed_dna_str.bits)} байтов')
```