**ЛАБОРАТОРНАЯ РАБОТА №4 BIOPYTHON**

**Задание 2. GC-составы**

Вариарнт 1

**Цель работы:**

Разработка программы на Python с использованием библиотеки BioPython для анализа GC-состава последовательностей, содержащихся в файле GenBank, и вывода этих последовательностей в порядке возрастания их GC-составов.

**Задачи:**

1. Чтение данных из файла GenBank: Прочитать записи GenBank из указанного файла с использованием модуля Bio.SeqIO.

2. Вычисление GC-состава: Для каждой последовательности в файле GenBank вычислить GC-состав как отношение количества гуанина (G) и цитозина (C) к общей длине последовательности.

3. Сортировка последовательностей: Отсортировать записи GenBank по GC-составу в порядке возрастания.

4. Вывод результатов: Вывести идентификатор (ID) каждой записи, описание и вычисленный GC-состав в отсортированном порядке.

**Инструменты:**

• Python 3: Язык программирования.

• BioPython: Библиотека Python для работы с биологическими последовательностями.

• Bio.SeqIO: Модуль BioPython для чтения и записи файлов последовательностей.

**Алгоритмы:**

1. **Чтение файла GenBank:**

• Импортировать модуль Bio.SeqIO.

• Использовать функцию SeqIO.parse(file\_path, "genbank") для чтения записей GenBank из указанного файла.

2. **Вычисление GC-состава:**

• Для каждой записи GenBank:

\* Извлечь последовательность ДНК (объект Seq) из записи.

\* Вычислить количество символов "G" и "C" в последовательности с использованием метода count().

\* Вычислить общую длину последовательности с использованием функции len().

\* Рассчитать GC-состав как (count("G") + count("C")) / len(последовательности).

3. **Сортировка записей:**

• Создать список кортежей, где каждый кортеж содержит идентификатор записи (ID), описание и GC-состав.

• Использовать метод sort() со lambda функцией в качестве ключа для сортировки списка кортежей по GC-составу (третий элемент кортежа).

4. **Вывод результатов:**

• Перебрать отсортированный список кортежей.

• Для каждой записи вывести ID, описание и GC-состав, отформатированный до 5 знаков после запятой с использованием f-строк.

**Реализация кода (Python):**

#Вариант 1

#Задание 2

from Bio import SeqIO

file\_path = "E:\example cds.gb" #путь к файлу

records = []

for record in SeqIO.parse(file\_path, "genbank"):

gc\_content = (record.seq.count("G") + record.seq.count("C")) / len(record.seq)

records.append((record.id, record.description, gc\_content))

# Сортируем записи по GC-составу

records.sort(key=lambda x: x[2])

# Выводим результат

for record\_id, description, gc\_content in records:

print(f"{record\_id}: {description}, GC = {gc\_content:.5f}")

**Ошибки и их исправления:**

1. **Отсутствие библиотеки BioPython:** Перед запуском скрипта убедитесь, что библиотека BioPython установлена. Если она не установлена, выполните команду pip install biopython.

2. **Неправильный путь к файлу:** Убедитесь, что переменная file\_path содержит правильный путь к вашему файлу GenBank. Если файл находится в той же директории, что и скрипт, можно использовать относительный путь.

**Выводы:**

• Разработана программа для анализа GC-состава последовательностей GenBank с использованием библиотеки BioPython.

• Программа успешно читает файл GenBank, вычисляет GC-состав для каждой последовательности, сортирует последовательности по GC-составу и выводит результаты в консоль.

• Данный подход может быть использован для анализа GC-состава больших наборов биологических последовательностей.

**Задание 3. Трансляция**

.**Цель работы:**

Разработка программы на Python с использованием библиотеки BioPython для извлечения кодирующих областей (CDS) из файла GenBank и трансляции их в соответствующие белковые последовательности.

**Задачи:**

1. Чтение данных из файла GenBank: Прочитать записи GenBank из указанного файла с использованием модуля Bio.SeqIO.

2. Извлечение кодирующих областей (CDS): Для каждой записи GenBank извлечь все признаки (features) типа "CDS" (кодирующая область).

3. Трансляция CDS: Получить белковые последовательности из каждой CDS. Предполагается, что информация о трансляции уже есть в поле translation в qualifiers.

4. Вывод результатов: Вывести для каждой CDS идентификатор записи GenBank, описание, координаты CDS и транслированную белковую последовательность.

**Инструменты:**

• Python 3: Язык программирования.

• BioPython: Библиотека Python для работы с биологическими последовательностями.

• Bio.SeqIO: Модуль BioPython для чтения и записи файлов последовательностей.

• Bio.SeqFeature: Модуль BioPython для работы с признаками (features) последовательности.

**Алгоритмы:**

1. **Чтение файла GenBank:**

• Импортировать модуль Bio.SeqIO.

• Использовать функцию SeqIO.parse(file\_path, "genbank") для чтения записей GenBank из указанного файла.

2. **Обработка каждой записи GenBank:**

• Для каждой записи GenBank вызвать функцию translate\_cds(record), которая выполняет следующие шаги:

\* Перебрать все признаки (features) в записи.

Для каждого признака проверить, является ли он CDS (feature.type == "CDS").

\* Если признак является CDS:

\* Извлечь начальную и конечную позицию CDS из атрибута feature.location.

\* Получить транслированную белковую последовательность из квалификатора (qualifier) "translation" с помощью feature.qualifiers.get("translation", [""])[0]. Значение по умолчанию [""] используется, чтобы избежать ошибок, если квалификатор "translation" отсутствует.

\* Вывести идентификатор записи, описание, положение CDS и транслированную белковую последовательность.

3. **Вывод результатов:**

• Вывести информацию о каждой CDS в формате, указанном в задании (ID записи, описание, координаты CDS, транслированная последовательность).

**Реализация кода (Python):**

#Задание 3

from Bio import SeqIO

def translate\_cds(record):

for feature in record.features:

if feature.type == "CDS":

start, end = feature.location.start, feature.location.end

protein\_seq = feature.qualifiers.get("translation", [""])[0]

print(f"{record.id}: {record.description}")

print(f"Coding sequence location = [{start}:{end}]")

print(f"Translation =\n{protein\_seq}\n")

def parse\_genbank(file\_path):

"""Чтение файла GenBank и трансляция CDS"""

for record in SeqIO.parse(file\_path, "genbank"):

translate\_cds(record)

file\_path = "E:\example cds.gb" #путь к файлу

parse\_genbank(file\_path)

**Выводы:**

• Разработана программа для извлечения и вывода транслированных белковых последовательностей из файла GenBank.

• Программа успешно читает файл GenBank, определяет CDS и извлекает соответствующие белковые последовательности.

• Данный подход может быть использован для автоматической трансляции CDS из больших наборов данных GenBank.