**Инструкция по использованию программы AlAn v. 1.2.2**

**Требуемые компоненты**

Для работы программы требуется, чтобы на компьютере был установлен python 3 с библиотекой pandas.

**Входные файлы**

В качестве входных файлов используются результаты работы программы npg\_explorer, а именно файлы “genes/partition-ungrouped.tsv” и “mutations/mut.tsv”.

**Синтаксис**

Запускается программа из командной строки командой “python alan.py -i [название входной директории] -o [название выходной директории] -c [требуемая доля ядра для признания группы идеальной] -e”. Все аргументы опциональны. -i отвечает за указание входной директории (по умолчанию текущая директория), -o – за указание выходной директории (по умолчанию текущая), -c – за указание требуемой доли ядра от длины группы для признания группы идеальной (по умолчанию 0.9), -e – за формат выходных файлов (если не дан параметр, то будет xlsx, иначе csv).

**Примерное время работы**

Программа работает около 2 – 3 минут на пангеноме из 10 – 15 геномов.

**Выходные файлы**

В результате работы программы будет создано два основных файла: “groups.xlsx”, в котором перечислена информация обо всех группах, составленных с учётом направления генов в блоках, и “groups\_no\_ori.xlsx”, в котором перечислена информация обо всех группах, составленных без учёта направления генов в блоках. Также создаётся две директории: “groups” и “groups\_no\_ori”, в которых приведены таблицы для каждой из групп, перечисленных в соответствующих таблицах. Директория “mut\_split” техническая, она требуются для работы программы.

В таблице “groups.xlsx” каждая строка – это группа, а колонки: первая неотмеченная – это индекс группы, он же её название, “block” – название блока, в котором составлена группа, “block\_type” – тип блока, “seq\_num” – число последовательностей в блоке, “block\_len” – длина блока, “ori” – направление всех фрагментов группы (в таблице “groups\_no\_ori.xlsx” добавляется значение 0, что означает, что группа содержит фрагменты, идущие в разных напрвлениях), “start” – координаты начала группы в блоке, “stop” – координаты конца группы в блоке, “len” – длина группы, “core\_start” – координаты начала ядра группы в блоке, “core\_stop” – координаты конца ядра группы в блоке, “core\_len” – длина ядра группы, “genes\_num” – число фрагментов в группе, “is\_inside” – будет иметь значение True, если все фрагменты группы являются полными генами (то есть не обрубаются границами блока, а начинаются со старт-кодона и заканчиваются стоп-), “core\_%” – доля длины ядра от длины группы, “is\_ideal” – пересечение колонок “is\_inside” и “is\_90”, дальше идут колонки с названиями последовательностей, где указано, есть ли в данной группе фрагмент из указанной последовательности.

В таблицах по каждой группе (которые лежат в папках “groups” и “groups\_no\_ori”) название соответствует индексу группы, каждая строка – это фрагмент, который есть в данной группе, первая неназванная колонка – это индекс фрагмента, “gene” – название гена, “start” – координата начала фрагмента в блоке, “stop” – координата конца фрагмента в блоке, “len” – длина фрагмента, “start\_rf” – в какой рамке считывания находится первый нуклеотид во фрагменте, “stop\_rf” – в какой рамке считывания находится последний нуклеотид во фрагменте, “is\_start” – является ли начало фрагмента началом гена (или он начинается в другом блоке), “is\_stop” – является ли конец фрагмента концом гена (или он кончается в другом блоке).

**Направления дальнейшей работы**

Глобально в программу требуется добавить две функции: аннотацию неаннотированных участков геномов, а также реаннотацию, то есть исправление существующей аннотации на более правильную.

**Известные ошибки**

Если в качестве выходной директории задать две вложенные несуществующие директории (например, “result/attempt\_1”, при том, что ни result, ни attempt\_1 не существуют), то программа завершится с ошибкой.