ДЗ №3 Кондратюк Александр Группа 3

Часть 1.

Задание 1.

Я выбрал следующие белки человека:

1. GBRP_HUMAN
2. CIROP_HUMAN
3. SOCS2_HUMAN
4. CLRN2_HUMAN
5. PPM1D_HUMAN
6. GLYG2_HUMAN
7. FLRT2_HUMAN
8. CHSTA_HUMAN
9. HAIR_HUMAN
10. ANGL7_HUMAN

Также были найдены ортологи у шимпанзе(pan troglodytes)

Далее построим парные выравнивания, используя

https://www.ebi.ac.uk/Tools/psa/emboss_needle/

1)GBRP_HUMAN

Ортолог - A0A2J8LXV6

```
#=======================================
#
# Aligned_sequences: 2
# 1: GBRP_HUMAN
# 2: A0A2J8LXV6_PANTR
# Matrix: EBLOSUM62
# Gap_penalty: 10.0
# Extend_penalty: 0.5
#
# Length: 440
# Identity:     439/440 (99.8%)
# Similarity:   440/440 (100.0%)
# Gaps:           0/440 ( 0.0%)
# Score: 2271.0
#
```

2) CIROP_HUMAN

Ортолог - A0A2I3SAV6

```
#=======================================
#
# Aligned_sequences: 2
# 1: CIROP_HUMAN
# 2: A0A2I3SAV6_PANTR
# Matrix: EBLOSUM62
# Gap_penalty: 10.0
# Extend_penalty: 0.5
#
# Length: 788
# Identity:     778/788 (98.7%)
# Similarity:   781/788 (99.1%)
# Gaps:           2/788 ( 0.3%)
# Score: 4163.0
```

3) SOCS2_HUMAN

Ортолог - A0A6D2VVR1

```
#=======================================
#
# Aligned_sequences: 2
# 1: SOCS2_HUMAN
# 2: A0A6D2VVR1_PANTR
# Matrix: EBLOSUM62
# Gap_penalty: 10.0
# Extend_penalty: 0.5
#
# Length: 198
# Identity:     197/198 (99.5%)
# Similarity:   198/198 (100.0%)
# Gaps:           0/198 ( 0.0%)
# Score: 1050.0
#
```

4) CLRN2_HUMAN

Ортолог - A0A2I3RH01

```
#=======================================
#
# Aligned_sequences: 2
# 1: CLRN2_HUMAN
# 2: A0A2I3RH01_PANTR
# Matrix: EBLOSUM62
# Gap_penalty: 10.0
# Extend_penalty: 0.5
#
# Length: 232
# Identity:     228/232 (98.3%)
# Similarity:   230/232 (99.1%)
# Gaps:           0/232 ( 0.0%)
# Score: 1164.0
```

5) PPM1D_HUMAN

Ортолог - A0A2J8JD59

```
#=======================================
#
# Aligned_sequences: 2
# 1: PPM1D_HUMAN
# 2: A0A2J8JD59_PANTR
# Matrix: EBLOSUM62
# Gap_penalty: 10.0
# Extend_penalty: 0.5
#
# Length: 607
# Identity:     422/607 (69.5%)
# Similarity:   422/607 (69.5%)
# Gaps:         179/607 (29.5%)
# Score: 2222.5
```

6) GLYG2_HUMAN

Ортолог - A0A2I3S1P4

```
#=======================================
#
# Aligned_sequences: 2
# 1: GLYG2_HUMAN
# 2: A0A2I3S1P4_PANTR
# Matrix: EBLOSUM62
# Gap_penalty: 10.0
# Extend_penalty: 0.5
#
# Length: 501
# Identity:     425/501 (84.8%)
# Similarity:   425/501 (84.8%)
# Gaps:          71/501 (14.2%)
# Score: 2182.0
```

7) FLRT2_HUMAN

Ортолог - A0A6D2WRA8

```
#=======================================
#
# Aligned_sequences: 2
# 1: FLRT2_HUMAN
# 2: A0A6D2WRA8_PANTR
# Matrix: EBLOSUM62
# Gap_penalty: 10.0
# Extend_penalty: 0.5
#
# Length: 660
# Identity:     657/660 (99.5%)
# Similarity:   658/660 (99.7%)
# Gaps:           0/660 ( 0.0%)
# Score: 3469.0
```

8) CHSTA_HUMAN

Ортолог - A0A6D2XWF8

```
#=======================================
#
# Aligned_sequences: 2
# 1: CHSTA_HUMAN
# 2: A0A6D2XWF8_PANTR
# Matrix: EBLOSUM62
# Gap_penalty: 10.0
# Extend_penalty: 0.5
#
# Length: 356
# Identity:     353/356 (99.2%)
# Similarity:   354/356 (99.4%)
# Gaps:           0/356 ( 0.0%)
# Score: 1907.0
#
```

9) HAIR_HUMAN

Ортолог - A0A6D2W5E7

```
#=======================================
#
# Aligned_sequences: 2
# 1: HAIR_HUMAN
# 2: A0A6D2W5E7_PANTR
# Matrix: EBLOSUM62
# Gap_penalty: 10.0
# Extend_penalty: 0.5
#
# Length: 1189
# Identity:    1117/1189 (93.9%)
# Similarity:  1121/1189 (94.3%)
# Gaps:          55/1189 ( 4.6%)
# Score: 6102.0
#
```

10) ANGL7_HUMAN

Ортолог - A0A6D2WJB5

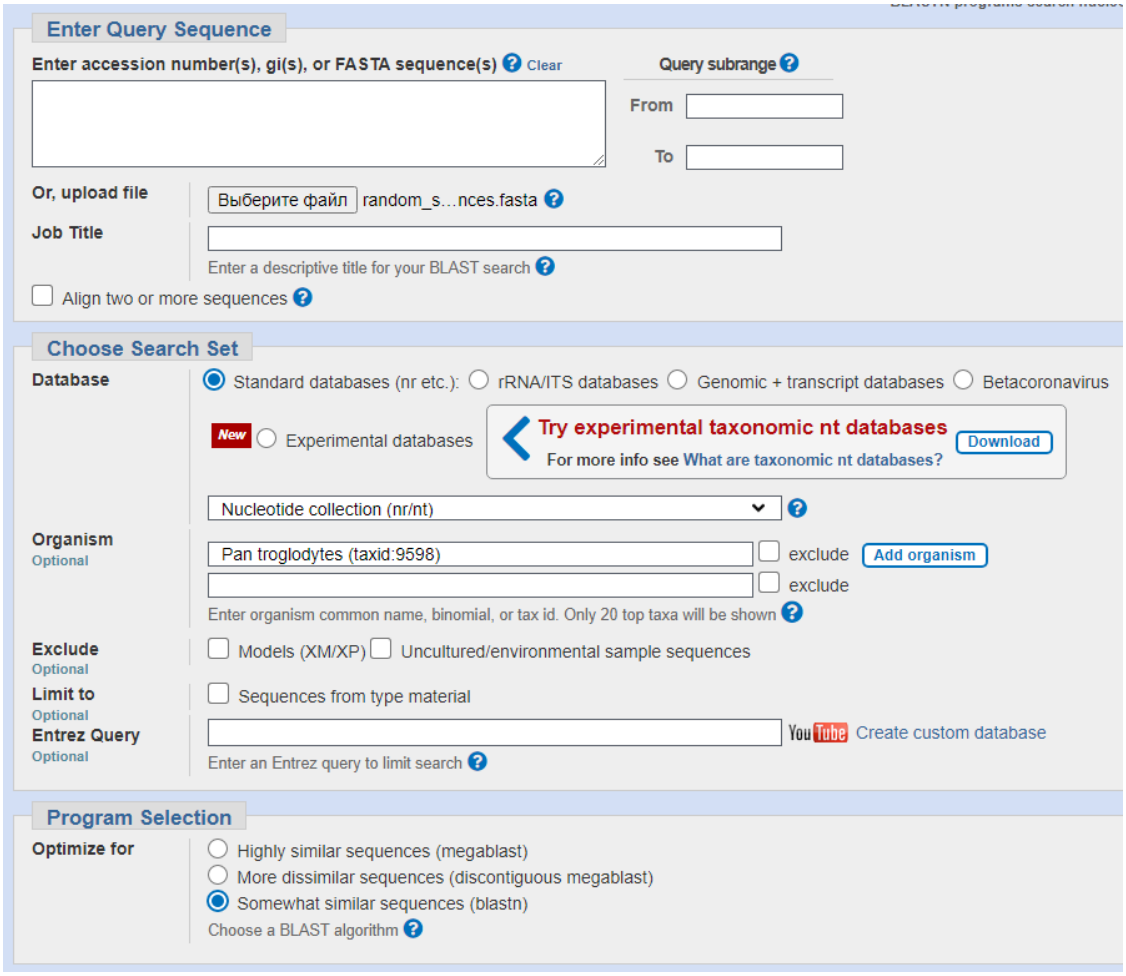
```
#=======================================
#
# Aligned_sequences: 2
# 1: ANGL7_HUMAN
# 2: A0A6D2WJB5_PANTR
# Matrix: EBLOSUM62
# Gap_penalty: 10.0
# Extend_penalty: 0.5
#
# Length: 346
# Identity:     344/346 (99.4%)
# Similarity:   344/346 (99.4%)
# Gaps:           0/346 ( 0.0%)
# Score: 1858.0
#
```

Задание 2.

Оценим сходство геномов найдя среднее сходство для 100 нуклеотидных последовательностей длины 100 выбранных из генома случайно.
Возьмем с NCBI человеческий геном, и с помощью скрипта извлечем 100 случайных последовательностей и запишем их в файл.

```
from Bio import SeqIO
import random
genome_file = 'C:/Users/АлександрКондратюк/PycharmProjects/pythonProject18/GRCh38_latest_genomic.fna'

num_sequences = 100
sequence_length = 100

#output_file = 'C:/Users/АлександрКондратюк/PycharmProjects/pythonProject18'

genome_sequences = list(SeqIO.parse(genome_file, "fasta"))

random_sequences = random.sample(genome_sequences, num_sequences)

seq_records = [SeqIO.SeqRecord(seq.seq[:sequence_length], id=seq.id, description="") for seq in random_sequences]

with open("random_sequences.fasta", "w") as output_file:
    SeqIO.write(seq_records, output_file, "fasta")
```

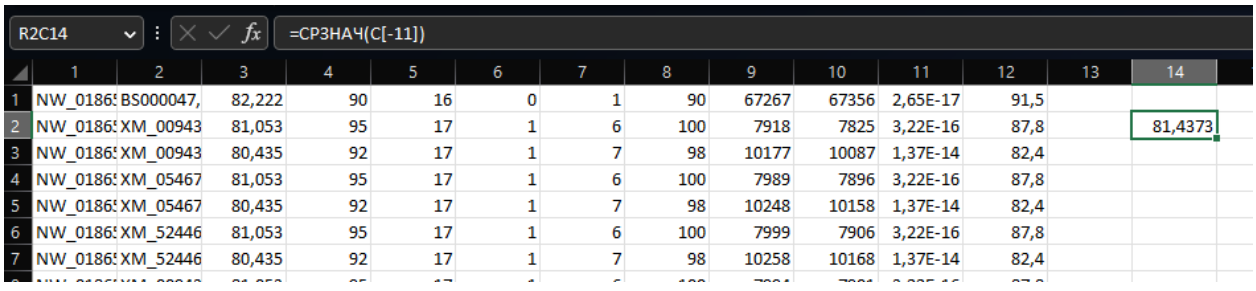
Полученный файл загружаем в BLAST (https://blast.ncbi.nlm.nih.gov/Blast.cgi?PROGRAM=blastn&PAGE_TYPE=BlastSearch&BLAST_SPEC=&LINK_LOC=blasttab&LAST_PAGE=blastx)



Полученный файл скачиваем в формате CSV и с помощью Excel находим Identity

	1	2	3	4	5	6	7	8	9	10	11	12
1	NW_0186	BS000047.	82.222	90	16	0	1	90	67267	67356	2.65e-17	91.5
2	NW_0186	XM_00943	81.053	95	17	1	6	100	7918	7825	3.22e-16	87.8
3	NW_0186	XM_00943	80.435	92	17	1	7	98	10177	10087	1.37e-14	82.4
4	NW_0186	XM_05467	81.053	95	17	1	6	100	7989	7896	3.22e-16	87.8
5	NW_0186	XM_05467	80.435	92	17	1	7	98	10248	10158	1.37e-14	82.4
6	NW_0186	XM_52446	81.053	95	17	1	6	100	7999	7906	3.22e-16	87.8
7	NW_0186	XM_52446	80.435	92	17	1	7	98	10258	10168	1.37e-14	82.4
8	NW_0186	XM_00943	81.053	95	17	1	6	100	7994	7901	3.22e-16	87.8
9	NW_0186	XM_00943	80.435	92	17	1	7	98	10253	10163	1.37e-14	82.4
10	NW_0186	XM_00943	81.053	95	17	1	6	100	8004	7911	3.22e-16	87.8
11	NW_0186	XM_00943	80.435	92	17	1	7	98	10263	10173	1.37e-14	82.4
12	NW_0186	XR_00170	81.053	95	17	1	6	100	1171	1264	3.22e-16	87.8
13	NW_0186	AC183296.	81.053	95	17	1	6	100	162814	162907	3.22e-16	87.8
14	NW_0186	AC183296.	80.435	92	17	1	7	98	160555	160645	1.37e-14	82.4
15	NW_0186	AC183296.	75.294	85	19	2	7	90	61026	60943	2.32e-05	52.7
16	NW_0186	AC183296.	72.619	84	23	0	7	90	94765	94682	2.83e-04	49.1
17	NW_0186	AC183540.	81.053	95	17	1	6	100	126487	126394	3.22e-16	87.8
18	NW_0186	XR_00853	81.522	92	16	1	9	100	4252	4342	1.13e-15	86.9
19	NW_0186	XR_00853	81.522	92	16	1	9	100	4974	5064	1.13e-15	86.9
20	NW_0186	XM_52959	81.522	92	16	1	9	100	5758	5848	1.13e-15	86.9
21	NW_0186	AC193015.	79.000	100	21	0	1	100	66236	66335	1.13e-15	86.9
22	NW_0186	AC212917.	80.851	94	17	1	7	100	80033	79941	1.13e-15	86.0
23	NW_0186	AC198441.	79.787	94	19	0	7	100	110776	110869	3.93e-15	85.1
24	NW_0186	AC192724.	85.333	75	10	1	25	99	142628	142701	1.37e-14	83.3
25	NW_0186	AC192855.	85.333	75	10	1	25	99	142289	142362	1.37e-14	83.3
26	NW_0186	AC192155.	79.000	100	20	1	1	100	144654	144752	1.37e-14	83.3
27	NW_0186	CT841527.	85.333	75	10	1	25	99	83762	83689	1.37e-14	83.3
28	NW_0186	CT574568.	85.333	75	10	1	25	99	18931	18858	1.37e-14	83.3
29	NW_0186	AC200164.	86.111	72	9	1	2	72	1485	1556	1.37e-14	82.4
30	NW_0186	CU467489.	81.176	85	16	0	1	85	125024	125108	1.37e-14	82.4
31	NW_0186	AC148942.	80.435	92	17	1	7	98	105901	105991	1.37e-14	82.4
32	NW_0186	AC148942.	75.000	72	18	0	29	100	69155	69226	8.10e-05	50.0
33	NW_0186	AC186198.	80.435	92	17	1	7	98	25645	25735	1.37e-14	82.4
34	NW_0186	AC182639.	86.111	72	9	1	2	72	9826	9755	1.37e-14	82.4
35	NW_0186	CT737142.	81.176	85	16	0	1	85	10382	10466	1.37e-14	82.4
36	NW_0186	CT033850.	81.176	85	16	0	1	85	10371	10455	1.37e-14	82.4
37	NW_0186	AC213003.	79.348	92	19	0	1	92	162704	162613	4.78e-14	81.5

Посчитаем среднее Identity:



Получаем Identity = 81,4373.

Часть 2.

Задание 1.

Вопрос 0. На ПЦР отправили загрязненный образец состоящий из 2х молекул ДНК исследуемого организма и 3х молекул ДНК загрязнения. Считая, что после каждого цикла число молекул удваивается. Определите, сколько % молекул будет принадлежать исследуемому организму после
а) (0,25 балла) десяти циклов ПЦР
б) (0,25 балла) сорока циклов ПЦ

а) 2*2^10 / 2*2^10 + 3 = 0,9985 = 99,85%

б) 2 * 2^40 / 2 * 40 + 3 = 0,9999 = 99,99%

Получаем, что в независимости от количества циклов, доля будет равна 40%.

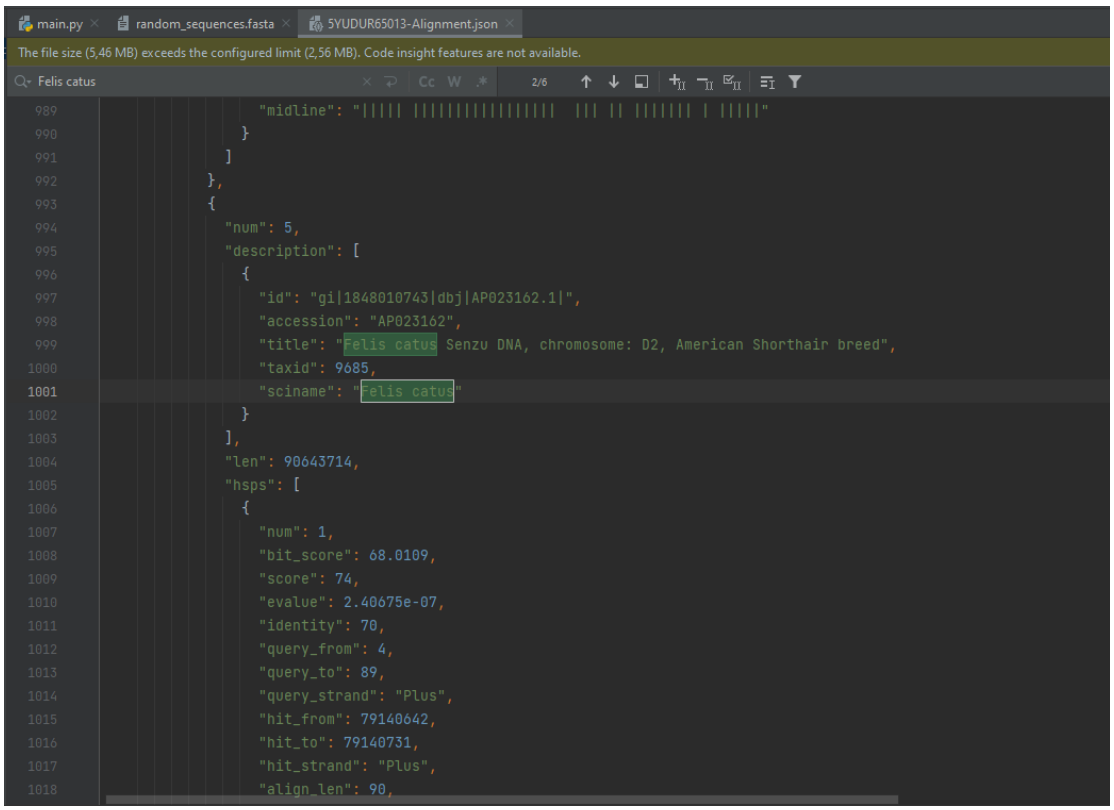
Вопрос 1.

Я скачал файл под номером 35.

Делаем BLAST и скачиваем результат.

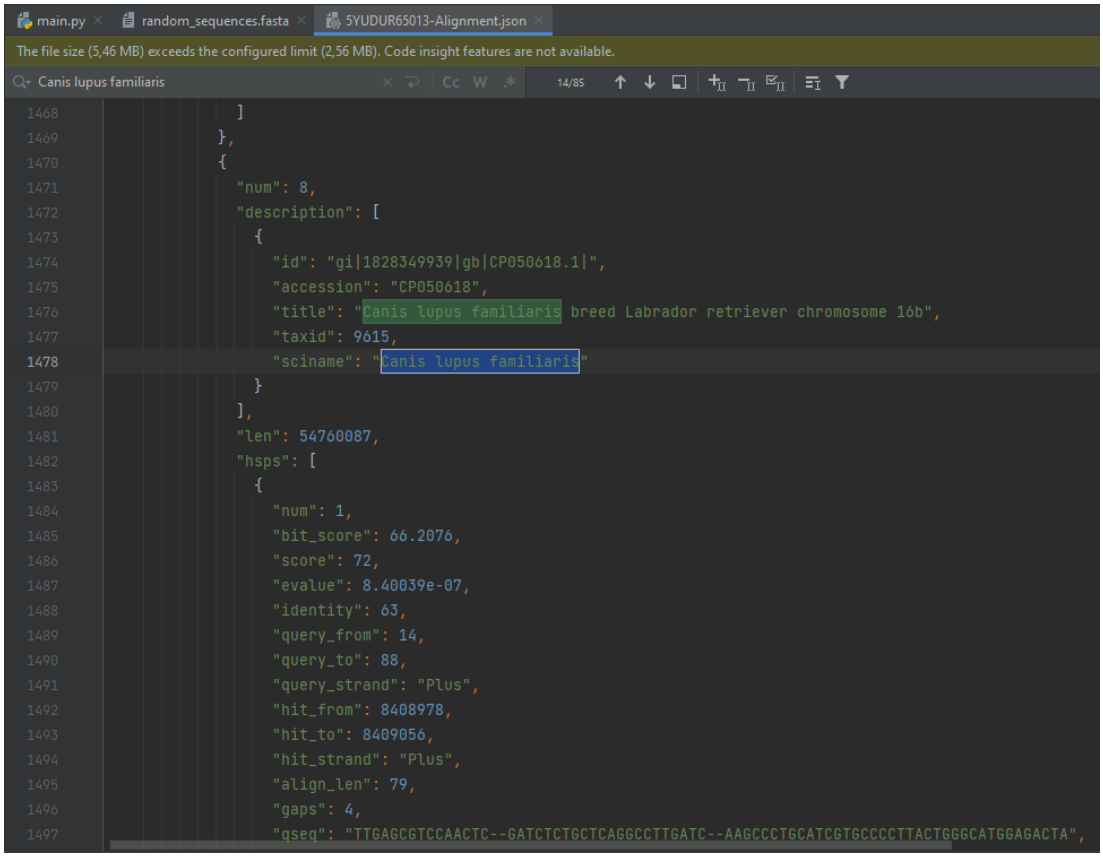
Произведем поиск для определения есть ли среди них кошка (*Felis catus*) или собака (*Canis lupus familiaris* или *Canis familiaris*).

Сделаем поиск кошки:



Видим, что для кошки найдено некоторое количество совпадений.

Теперь произведем поиск собаки



Для собаки тоже были найдены совпадения, но их количество гораздо больше, чем у кошки, также есть записи с identity 100%. Могу сделать вывол, что скорее всего загрязнение появилось по вине Иванова.

Вопрос 2.

Определим, к геному какого организма относится каждое чтение и найдем долю для каждого представленного вида.

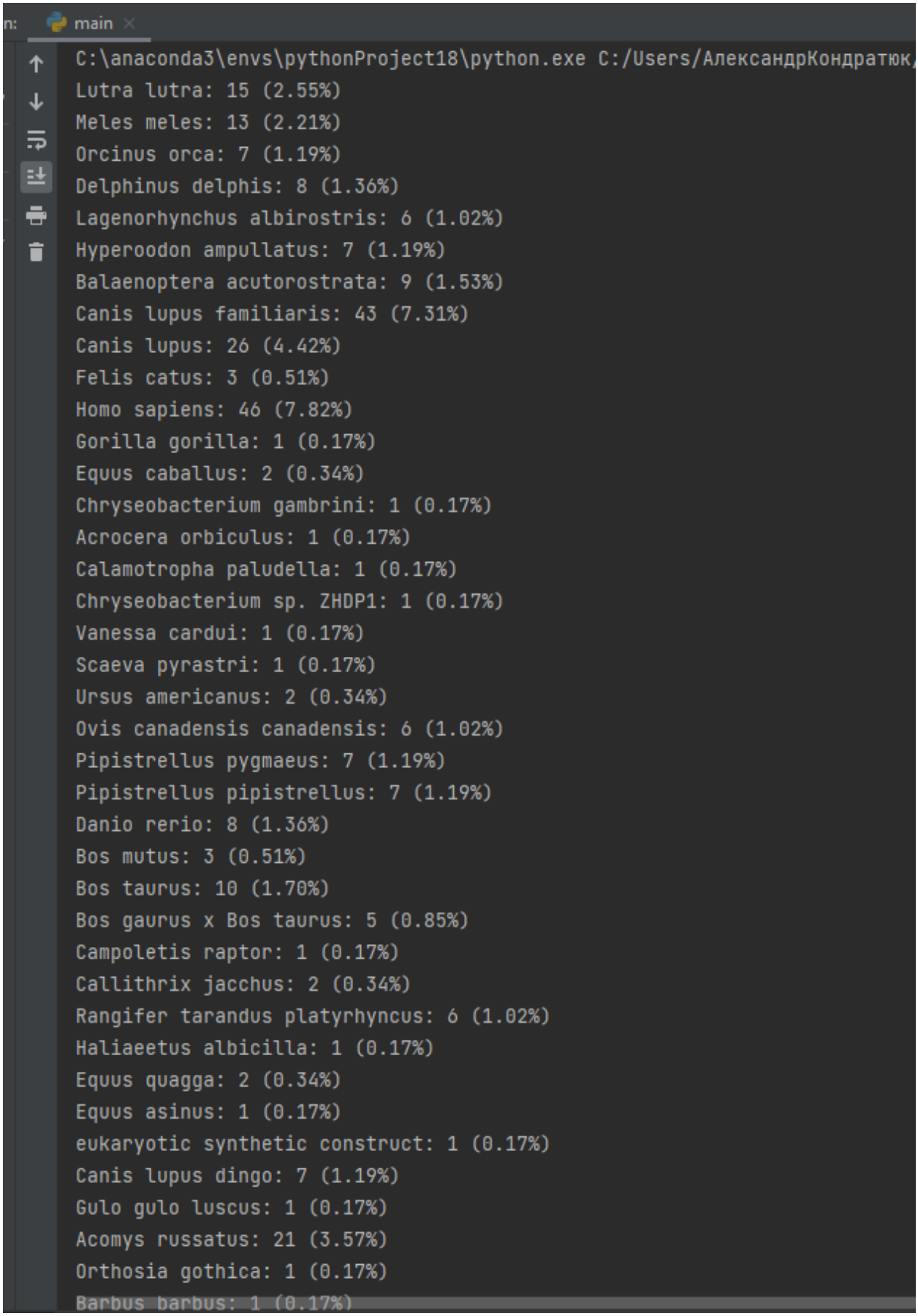
Напишем небольшой код:

main.py | q2.py | random_sequences.fasta | blast_results.json

```python
import json
import matplotlib.pyplot as plt
from collections import Counter

with open('blast_results.json') as f:
    data = json.load(f)

name_list = []

for result in data['BlastOutput2']:
    try:
        hits = result['report']['results']['search']['hits']
        for hit in hits:
            description = hit['description'][0]
            sciname = description['sciname']
            name_list.append(sciname)
    except:
        print("Results are empty")

counts = Counter(name_list)
total_count = len(name_list)

for name, count in counts.items():
    percentage = (count / total_count) * 100
    print(f"{name}: {count} ({percentage:.2f}%)")

names = list(counts.keys())
counts = list(counts.values())

plt.bar(names, counts)
plt.xlabel('Организмы')
plt.ylabel('Количество')
plt.title('Распределение организмов')
plt.xticks(rotation=45)

for i, count in enumerate(counts):
    percentage = (count / total_count) * 100
    plt.text(i, count, f"{percentage:.2f}%", ha='center', va='bottom')
plt.show()
```

Мы проходимся по json файлу и извлекаем все организмы из каждого результата. Затем подсчитываем количество элементов и их долю. (Файл со всеми прикреплю отдельно в формате .txt)

Вот часть результатов работы кода:



На 9-10.

Теперь чуть изменим код и сделаем топ 10 организмов

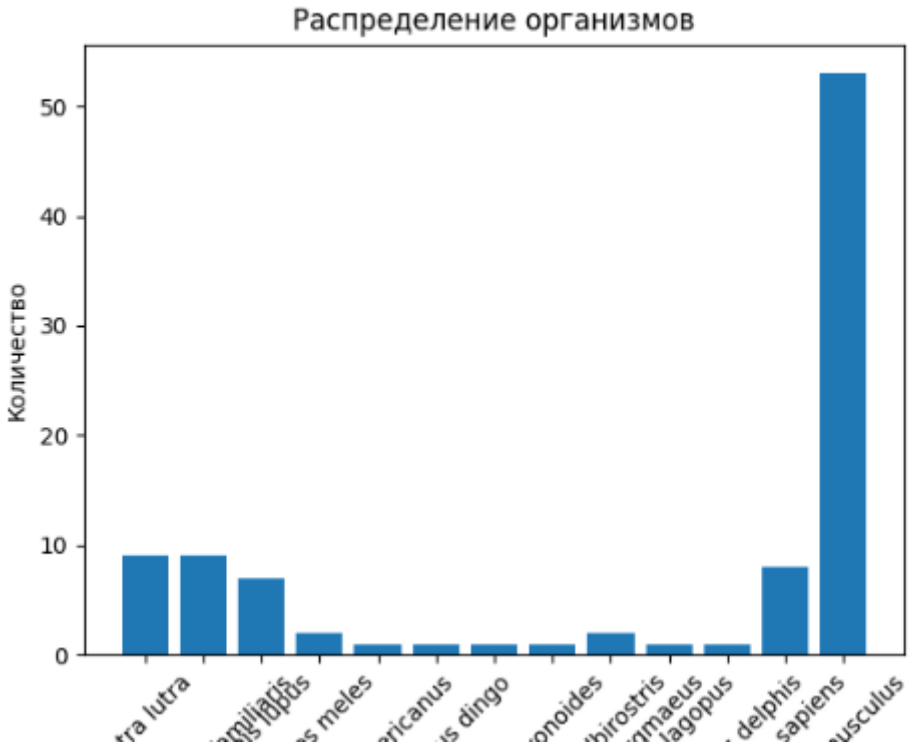
Чуть изменим код:

```
    percentage = (count / total_count) * 100
    plt.text(i, count, f"{percentage:.2f}%", ha='center', va='bottom')

plt.show()
```

Получаем результат:

```
Mus musculus: 123 (20.92%)
Homo sapiens: 46 (7.82%)
Canis lupus familiaris: 43 (7.31%)
Canis lupus: 26 (4.42%)
Acomys russatus: 21 (3.57%)
Lutra lutra: 15 (2.55%)
Chionomys nivalis: 15 (2.55%)
Meles meles: 13 (2.21%)
Bos taurus: 10 (1.70%)
Vulpes lagopus: 10 (1.70%)
```

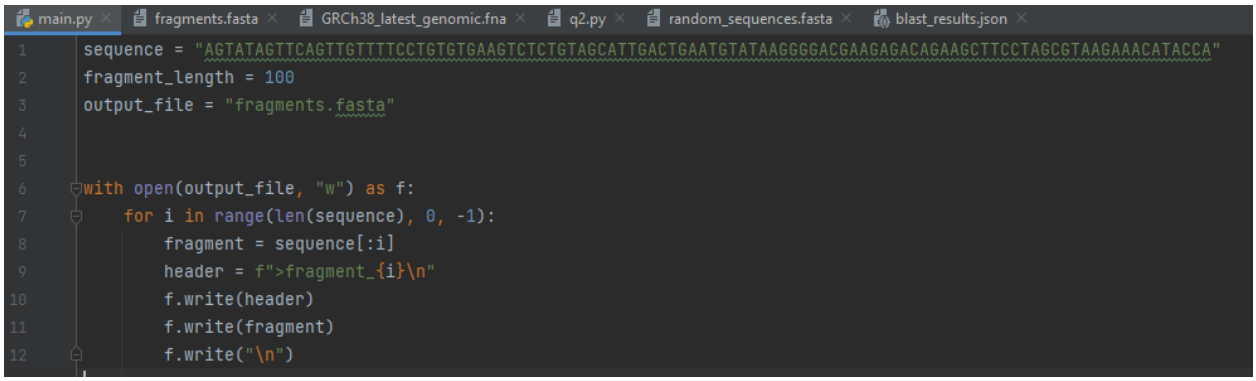
Также получим таблицу:



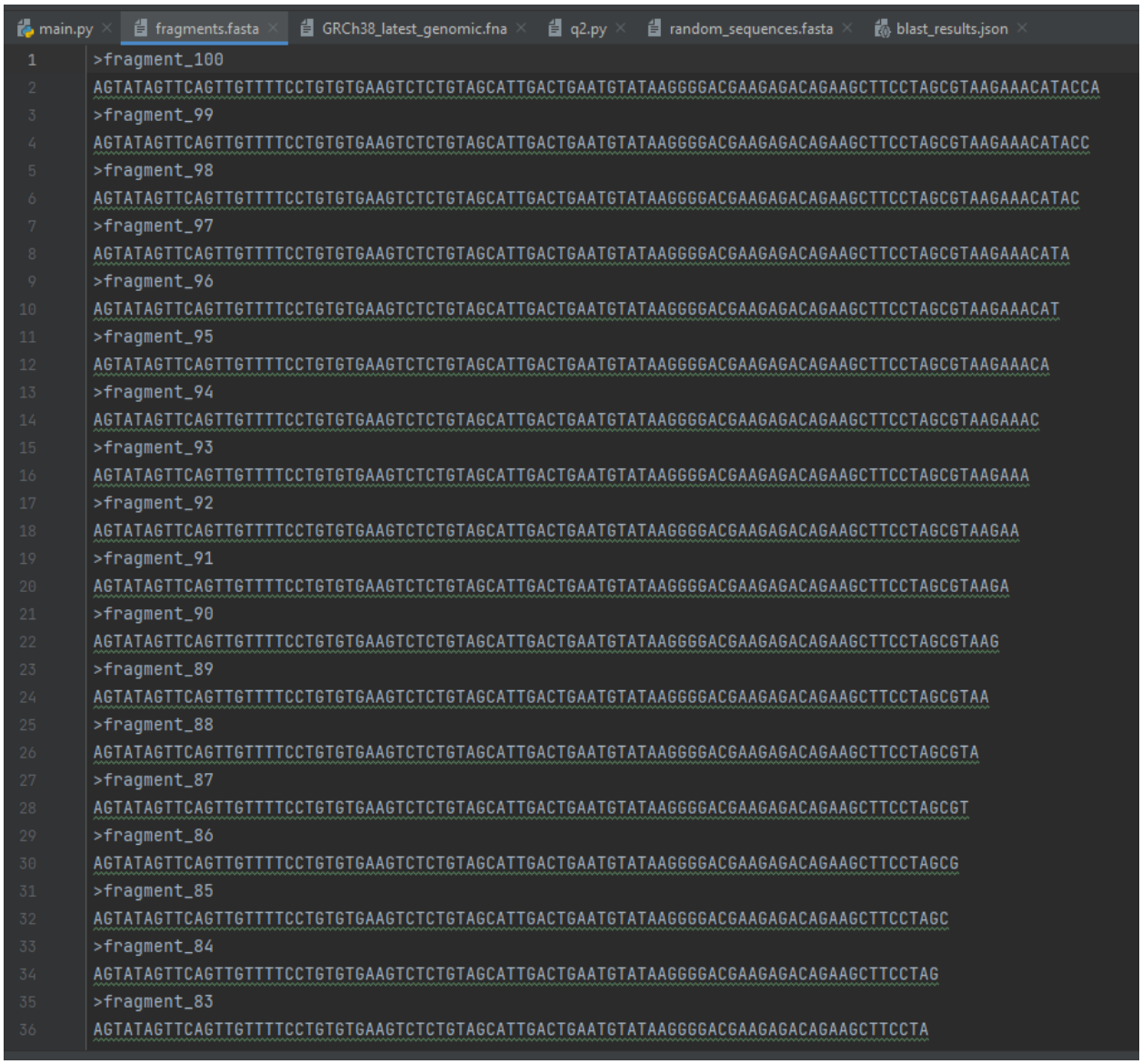
(названия немного поехали)

Задание 2. Фрагментация ДНК

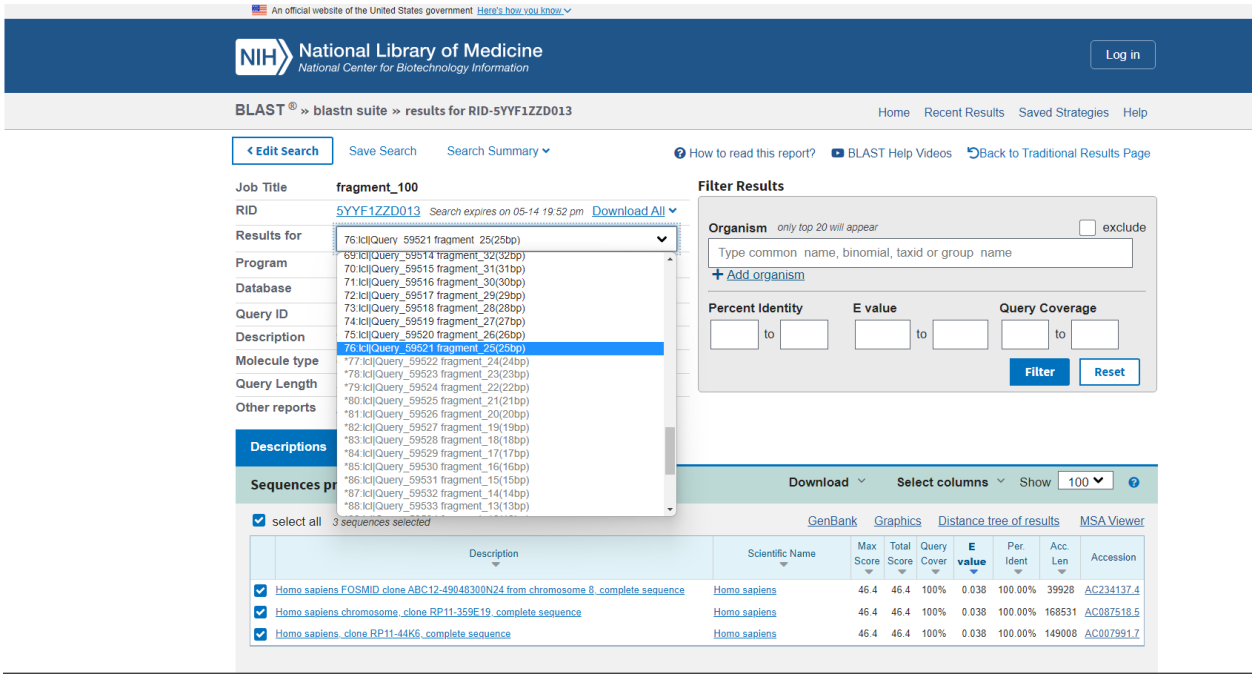
Сначала напишем скрипт, который сделает нам нужный файл, и возьмем AGTATAGTTCAGTTGTTTTCCTGTGTGAAGTCTCTGTAGCATTGACTGAATGTATAAGGGGACGAAGAGACAGAAGCTTCCTAGCGTAAGAAACATACCA



Получаем файл fragments.fasta (прикреплю)

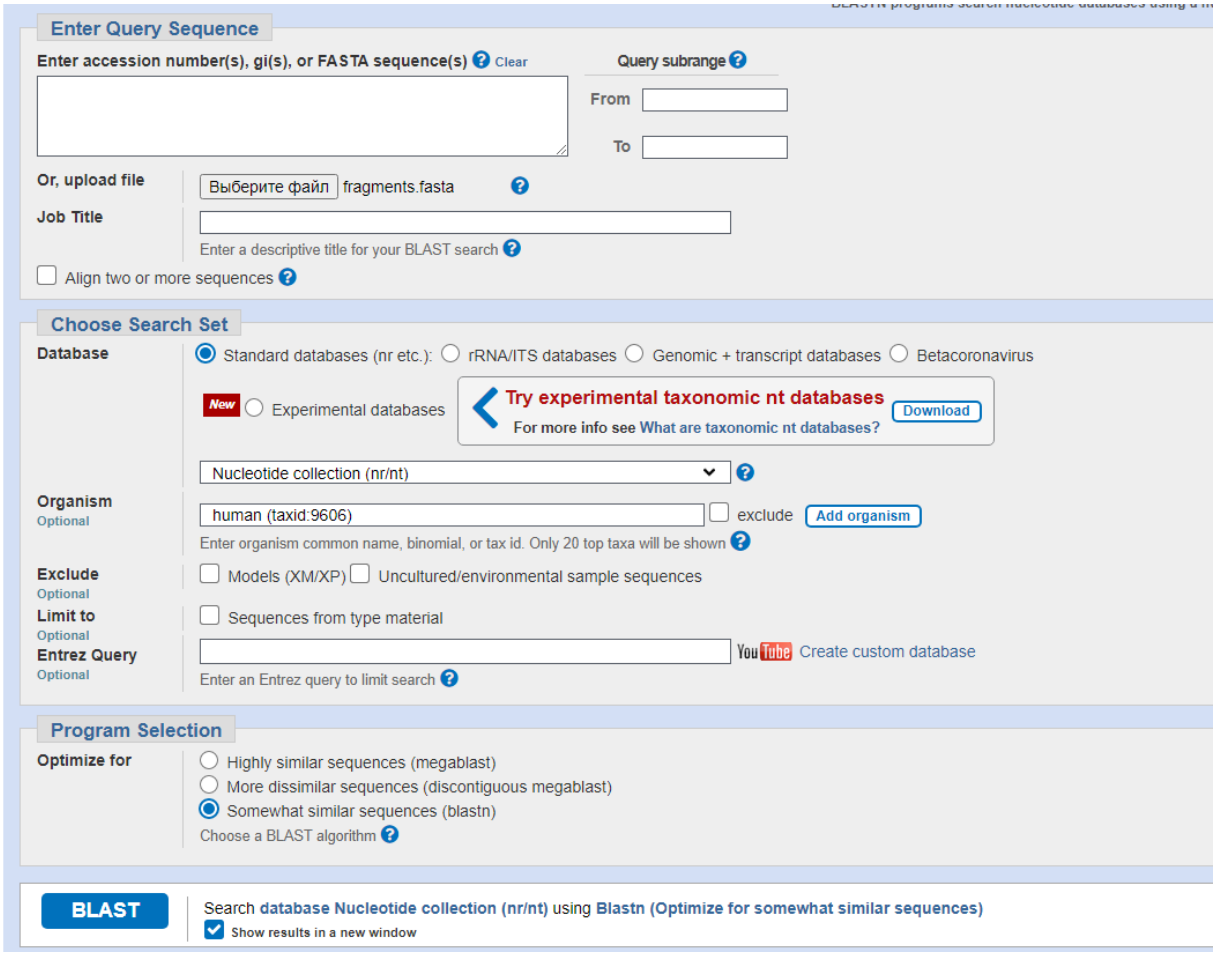


а) Делаем BLAST

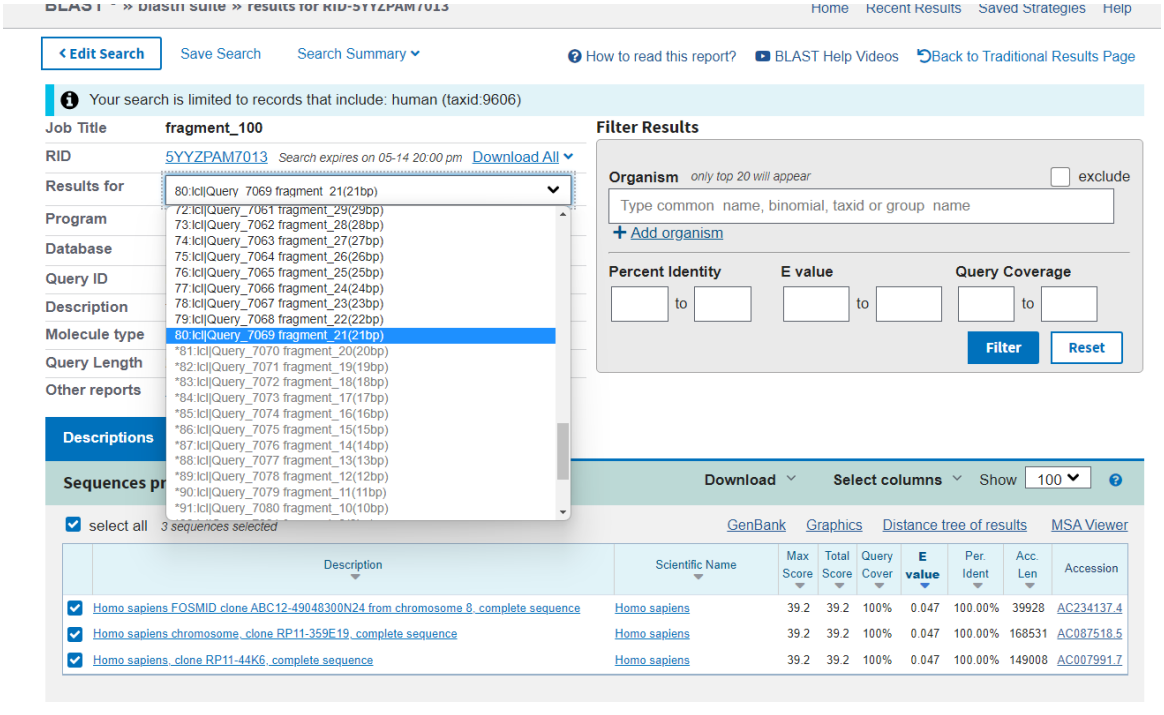


Видим, что после 76 последовательности Blast перестает что-либо находить, а ее E-value = 0.038. Получается что при длине последовательности меньше 25, E-value становится больше 0.05.

б) Теперь ограничим поиск человеком и посмотрим что изменится



Если запускать BLAST таким образом, то значение изменится и теперь E-value становится больше 0.05 только при последовательности меньше 21.



Так как мы ограничиваем поиск человеком, уменьшается и количество возможный выравниваний, следовательно те последовательности который раньше имели e-value больше 0.05, теперь имеют меньше.

На 9-10:

График lg(E) от n, где n длина фрагмента, а E-evalue лучшей находки:

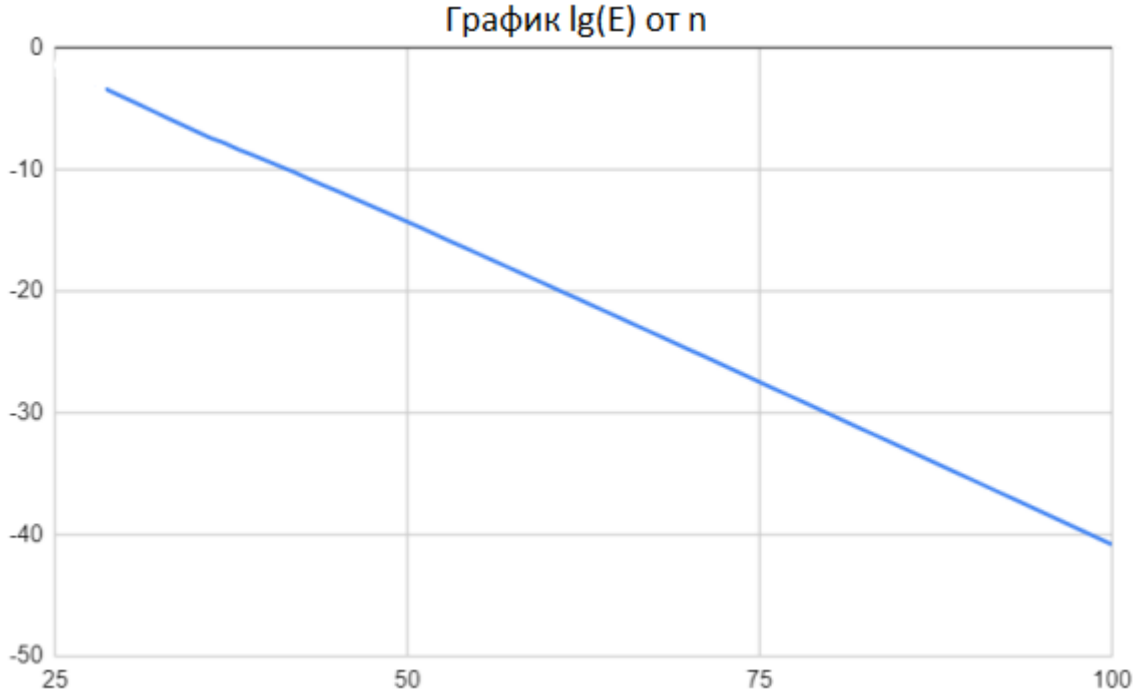


График max_N от n, где n длина фрагмента, а max_N число результатов с identity равным identity лучшего результата.

Вручную посмотрел в таблице BLAST, везде max_N для каждой длины фрагмента будет 3(вот доказательство, таблицу также приложу blast76 называется)

fragment_100	AC234137.	100.000	100	0	0	1	100	33114	33015	1.52e-41		181
fragment_100	AC087518.	100.000	100	0	0	1	100	54209	54308	1.52e-41		181
fragment_100	AC007991.	100.000	100	0	0	1	100	145459	145558	1.52e-41		181
fragment_100	OX465371	76.000	100	24	0	1	100	8591020	8591119	5.66e-09	73.4	
fragment_100	OX457075	75.248	101	24	1	1	100	5990608	5990508	8.40e-07	67.1	
fragment_100	OX465347	74.000	100	26	0	1	100	4750203	4750104	2.93e-06	64.4	
fragment_100	OX459093	74.000	100	26	0	1	100	4239208	4239109	2.93e-06	64.4	
fragment_100	OW44338:	74.000	100	26	0	1	100	4342146	4342047	2.93e-06	64.4	
fragment_100	CP050577.	72.000	100	22	1	1	100	25728280	25728187	0.002	56.3	
fragment_100	CP050618.	72.000	100	22	1	1	100	25757685	25757592	0.002	56.3	
fragment_100	HG994398	72.000	100	22	1	1	100	34929245	34929338	0.002	56.3	
fragment_100	OX463242	73.626	91	21	1	1	91	85841548	85841635	0.002	55.4	
fragment_100	OX460411	72.000	100	28	0	1	100	16946232	16946133	0.002	55.4	
fragment_100	XR_00151	75.000	76	19	0	6	81	445	370	0.019	52.7	
fragment_99	AC234137.	100.000	99	0	0	1	99	33114	33016	5.21e-41		179
fragment_99	AC087518.	100.000	99	0	0	1	99	54209	54307	5.21e-41		179
fragment_99	AC007991.	100.000	99	0	0	1	99	145459	145557	5.21e-41		179
fragment_99	OX465371	75.758	99	24	0	1	99	8591020	8591118	1.94e-08	71.6	
fragment_99	OX457075	75.000	100	24	1	1	99	5990608	5990509	2.89e-06	65.3	
fragment_99	OX465347	73.737	99	26	0	1	99	4750203	4750105	1.01e-05	62.6	
fragment_99	OX459093	73.737	99	26	0	1	99	4239208	4239110	1.01e-05	62.6	
fragment_99	OW44338:	73.737	99	26	0	1	99	4342146	4342048	1.01e-05	62.6	
fragment_99	OX463242	73.626	91	21	1	1	91	85841548	85841635	0.001	55.4	
fragment_99	CP050577.	71.717	99	22	1	1	99	25728280	25728188	0.005	54.5	
fragment_99	CP050618.	71.717	99	22	1	1	99	25757685	25757593	0.005	54.5	
fragment_99	OX460411	72.165	97	27	0	1	97	16946232	16946136	0.005	54.5	
fragment_99	HG994398	71.717	99	22	1	1	99	34929245	34929337	0.005	54.5	
fragment_99	XR_00151	75.000	76	19	0	6	81	445	370	0.018	52.7	
fragment_98	AC234137.	100.000	98	0	0	1	98	33114	33017	1.79e-40		178
fragment_98	AC087518.	100.000	98	0	0	1	98	54209	54306	1.79e-40		178
fragment_98	AC007991.	100.000	98	0	0	1	98	145459	145556	1.79e-40		178
fragment_98	OX465371	75.510	98	24	0	1	98	8591020	8591117	6.68e-08	69.8	
fragment_98	OX457075	74.747	99	24	1	1	98	5990608	5990510	9.91e-06	63.5	
fragment_98	OX465347	73.469	98	26	0	1	98	4750203	4750106	3.46e-05	60.8	
fragment_98	OX459093	73.469	98	26	0	1	98	4239208	4239111	3.46e-05	60.8	
fragment_98	OW44338:	73.469	98	26	0	1	98	4342146	4342049	3.46e-05	60.8	
fragment_98	OX463242	73.626	91	21	1	1	91	85841548	85841635	0.001	55.4	

Значит график будет выглядеть так: