1. Build a hash table for the provided list of sequences “seqs” (same as in the paper). Print the table.

```
('AA', [(2, 19)])
('AC', [(1, 9), (2, 5), (2, 11)])
('AG', [(1, 15), (2, 35)])
('AT', [(2, 13), (3, 3)])
('CA', [(2, 3), (2, 9), (2, 21), (2, 27), (2, 33), (3, 21), (3, 23)])
('CC', [(1, 21), (2, 31), (3, 5), (3, 7)])
('CG', [(1, 5)])
('CT', [(1, 23), (2, 39), (2, 43), (3, 13), (3, 15), (3, 17)])
('GA', [(1, 3), (1, 17), (2, 15), (2, 25)])
('GG', [(1, 25), (1, 31), (2, 17), (2, 29), (3, 1)])
('GT', [(1, 1), (1, 27), (1, 29), (2, 1), (2, 37), (3, 19)])
('TA', [(3, 25)])
('TC', [(1, 7), (1, 11), (1, 19), (2, 23), (2, 41), (3, 11)])
('TG', [(1, 13), (2, 7), (3, 9)])
```

2. Calculate the sorted list of hits for the provided query sequence. Each hit should be a tuple of (index, shift, offset). Print the number of hits and the first and last hit in your list.

```
number of hits: 23
 first hit: (1, 5, 9)
 last hit: (3, 21, 23)
```

3. Calculate the maximum match for the provided query sequence, and print the alignment (the two sequences with the right offset, such that the characters match). For example:

```
    GAC G G
     . | | | |
ATTCACGGCTA
```

```
      TGCAACAT
GTCAACTGCAACATGAGGAACATCGACAGGCCCAAGGTCTTCCT
```

The query sequence is 100% aligned with part of sequence2 (local alignment)

4. DATA: http://www.bioinformatics.nl/courses/BIF-31306/TAIR10.fasta Write a fasta parser to read in the Arabidopsis thaliana genome. How many sequences does the genome file contain and what is the total sequence length? Does this match your expectation on the Arabidopis genome? NOTE: parsing the fasta file should only take a few seconds, otherwise you are doing something very inefficient.

```
sequence: 1508184 total length:  119146348
```

The large scale of the data corresponds with my expectations of the size of a genome.

5. Build a hash table for the Arabidopis chromosomes. Use k = 15. How many keys (different 15-mers) does your hash contain?

There are 7160602 different 15-mers in the hash table (take each line in the file as a separate sequence)

6. DATA: http://www.bioinformatics.nl/courses/BIF-31306/athal_query.fasta For the query sequences in the FASTA file athal_query.fasta, find the maximum hit(s) in the genome. Report your findings (if any).

The longest match is as follows:

AAGTATATGGAGTTGATCAGGTAAGAGAGAATGCATTCGATAGACGCTGGTGAAATCTTCAGTTAG AGAGTAAGGG

The maximum hits list is as follows:

[(636542, -48, 1), (636542, -48, 16), (636542, -48, 31), (636542, -48, 46), (636542, -48, 61)]

From the output we can refer that the longest match occurs on the 636543th sequence in the Arabidopis chromosomes genome (start from the second base in line 636543 in the data file)

(ps. I joined the 3 query sequence together and didn't distinguish separate query sequence, though I'm not sure whether it is appropriate to do so)

7. What is the algorithm trying to optimize?

This algorithm tries to align as many consecutive bases in the query sequence with the bases in the subject sequence as possible. (obtain the maximum exact alignment)

8. What did you learn about the time complexity of the search part of the algorithm?

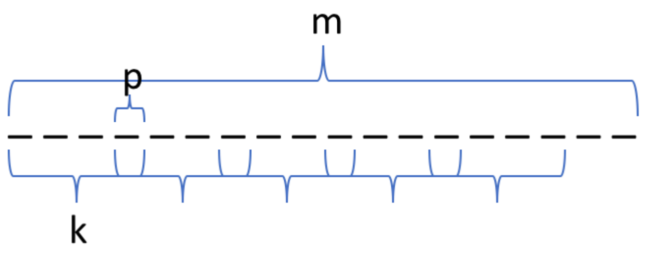
SSAHA is generally fast because the database is hashed, i.e., positions of k-mers in the database are stored in a hash table beforehand. Therefore, as long as the hash table is obtained, we can locate the k-mer positions in the query sequence directly with the help of hash table. The time complexity is $O(n)$

9. What is the main difference in time complexity of the SSAHA algorithm and BLAST?

In BLAST, first we need to scan the database sequence for an exact match to query words. Then extending matching words to both direction using un-gapped alignments. Extension continues as long as score does not fall below a given threshold (this is an high scoring pair). Then dynamic programming is employed to calculate alignment score for the extend gapped high scoring pair. From the description of BLAST, we can observe that the time complexity is polynomial.

For SSAHA, the time needed is dominated by constructing the hash table. But once the hash table is obtained, we only need to search the location of k-mers in query sequence with the help of hash table, the time complexity can be seen as constant.

10.How many k-mers with an overlap of p bases does a reference sequence of length m contain?



$$m - k + p <= (k - p)*(x - 1) + k <= m$$

$$\rightarrow (m-k)/(k-p) <= x <= (m-p)/(k-p)$$

The number of k-mers in a reference sequence of length m with an overlap of p bases is an integer falls in range of $[\frac{m-k}{k-p}, \frac{m-p}{k-p}]$