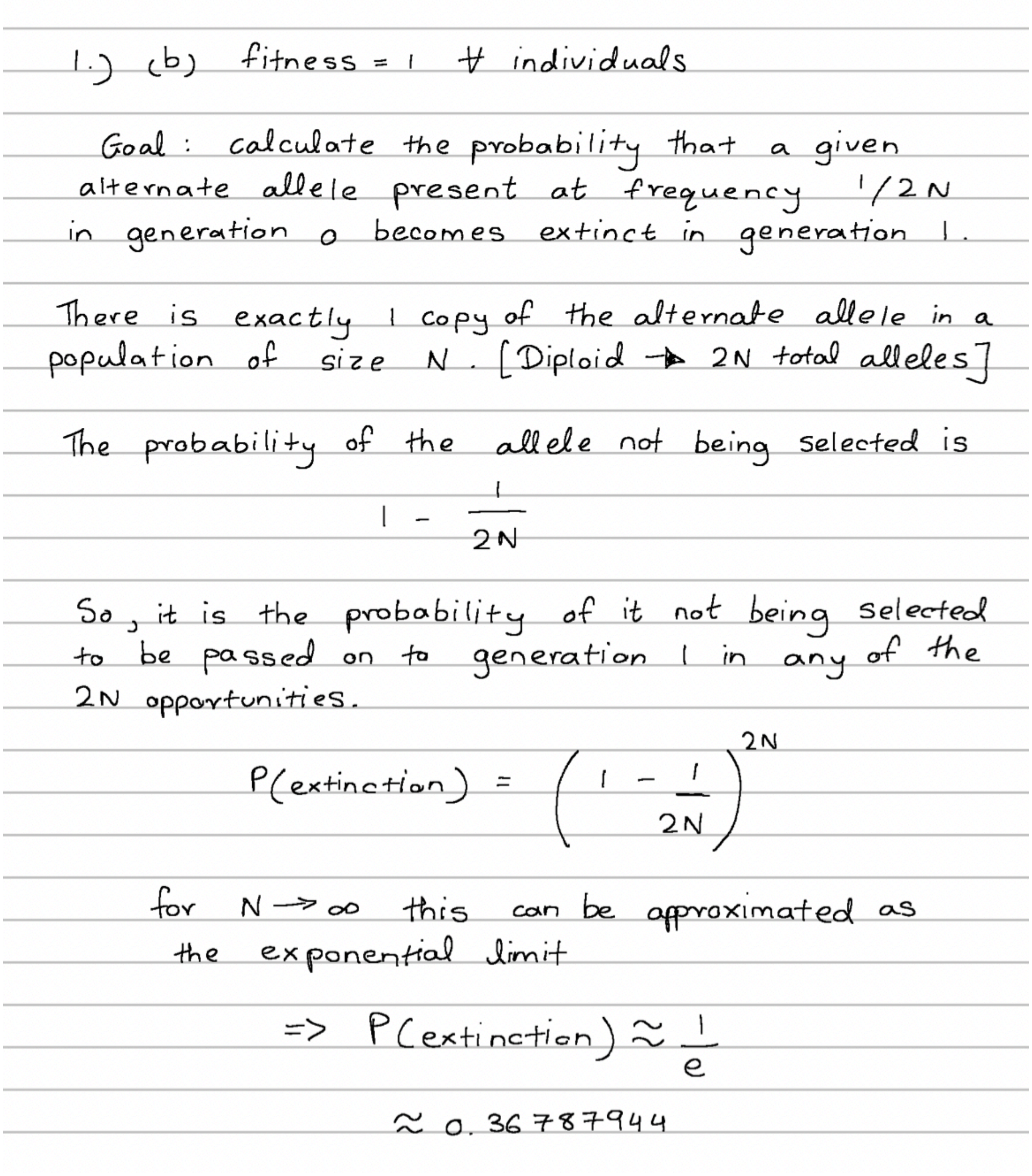
Homework - 2

Q.1.

a. Code attached to submission



1.) (b) fitness = 1 ∀ individuals

Goal : calculate the probability that a given alternate allele present at frequency $1/2N$ in generation 0 becomes extinct in generation 1.

There is exactly 1 copy of the alternate allele in a population of size N. [Diploid → 2N total alleles]

The probability of the allele not being selected is

$$1 - \frac{1}{2N}$$

So, it is the probability of it not being selected to be passed on to generation 1 in any of the 2N opportunities.

$$P(\text{extinction}) = \left(1 - \frac{1}{2N}\right)^{2N}$$

for $N \rightarrow \infty$ this can be approximated as the exponential limit

$$\Rightarrow P(\text{extinction}) \approx \frac{1}{e}$$

$$\approx 0.36787944$$

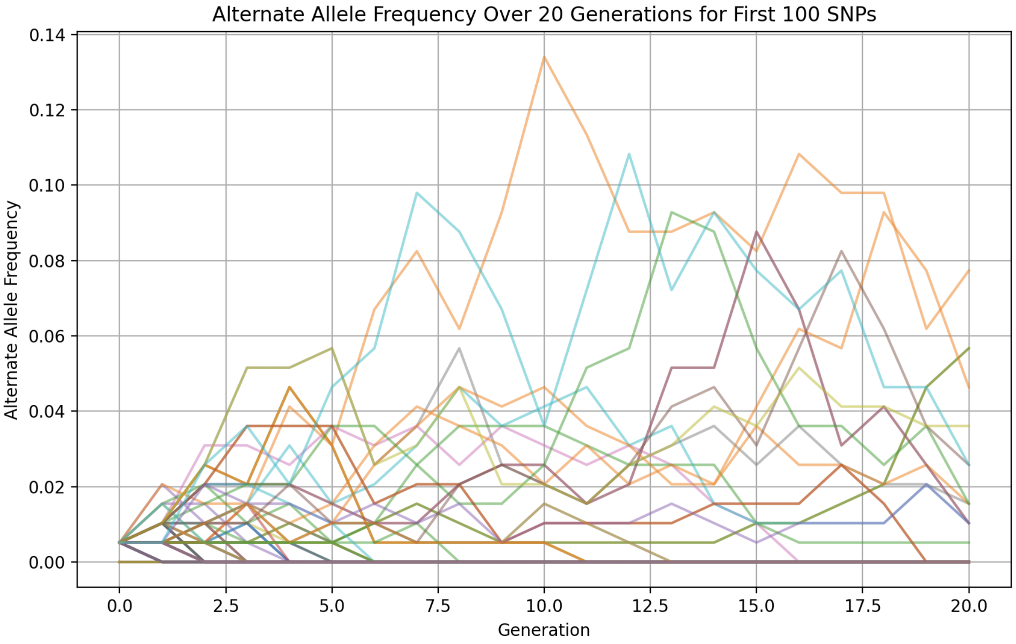
b.

c. Output:

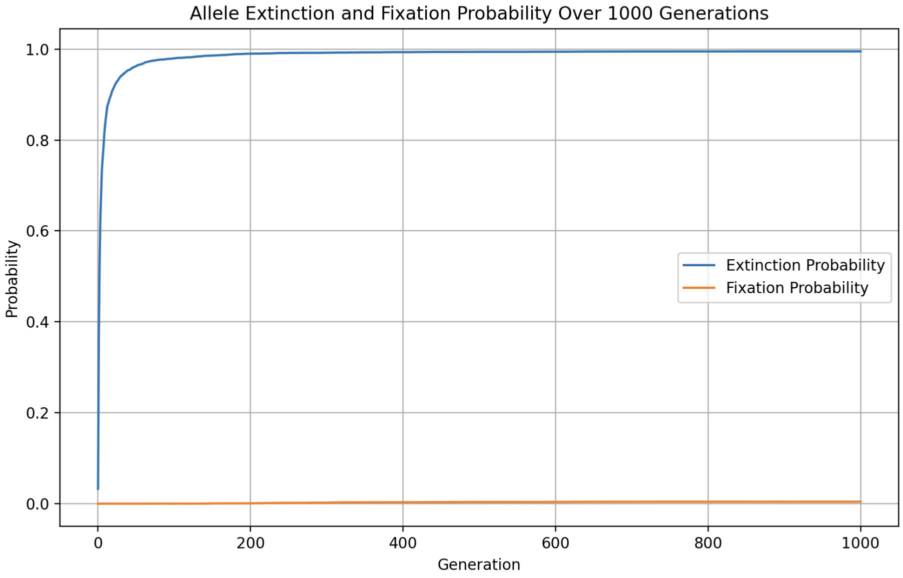
```
Estimated extinction probability: 0.3514
Theoretical extinction probability: 0.36787944117144233
Extinction count: 3514, Total SNPs: 10000
```

The output derived approximately matches our expected value of extinction probability.

d.



e.



f.) alt allele on individual at SNP42 (heterozygous) so fitness = 1.5

population fitness = $(N-1) + 1.5 = N + 0.5$

case:01

*parent 1

P(child inherits alt allele) = 0.5

P(chosen as parent and child does not inherit alt allele) $= \frac{1.5}{(N+0.5)} \times 0.5$

case:02

P(chosen as parent 2) $= \frac{1.5}{(N+0.5-1)} = \frac{1.5}{N-0.5}$

P(child inherits alt allele) = 0.5

P(chosen as parent and child does not inherit alt allele) $= \frac{1.5}{(N-0.5)} \times 0.5$

case:03

P(i chosen as p1) $= \frac{1.5}{(N+0.5)}$ P(i chosen as p2) $= \frac{1.5}{N-0.5}$

P(not chosen)

$= 1 - \frac{1.5}{N+0.5} - \frac{1.5}{N-0.5}$

mutually exclusive

P(child does not inherit alt alleles)

$$= \left(\frac{1.5}{N+0.5}\right) * 0.5 + \left(\frac{1.5}{N-0.5}\right) * 0.5$$

$$+ \left(1 - \frac{1.5}{N+0.5}\right) - \frac{1.5}{N-0.5}$$

$$= 1 - \left(\frac{1.5}{N+0.5}\right) * 0.5 - \frac{1.5}{(N-0.5)} * 0.5$$

$$= \frac{(N+0.5)(N-0.5) - 0.5(1.5)(N-0.5) - (0.5)(1.5)(N+0.5)}{(N+0.5)(N-0.5)}$$

$$= \frac{N^2 - 1.5N - 0.25}{N^2 - 0.25} \longrightarrow \text{runs } N \text{ times}$$

$$\therefore P(\text{extinct alt allele}) = \left(\frac{N^2 - 1.5N - 0.25}{N^2 - 0.25}\right)^N$$

$\therefore$ for $N = 100$

$\Rightarrow$ using calculator $\approx \underline{\underline{0.22}}$

f. Output obtained:

   Estimated extinction probability at SNP42 after 1 generation: 0.237

g. Output obtained:

   Estimated extinction probability at SNP42 after 100 generations: 0.015
   Estimated fixation probability at SNP42 after 100 generations: 0.985

h. Output obtained:

   Estimated extinction probability at SNP42 (deleterious) after 100 generations: 1.0
   Estimated fixation probability at SNP42 (deleterious) after 100 generations: 0.0

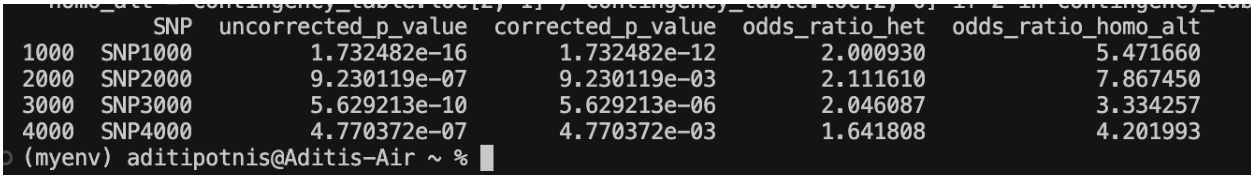
Q.2.

a. Code attached to submission.
b. Output from code:

   Number of SNPs with p-value < 0.05: 458
   Expected number of SNPs with p-value < 0.05 by chance: 500.0

c. Output from code:

SNP	Uncorrected p_value	Corrected p-value	Diseases Odds ratio for Heterozygous individuals	Disease Odds ratio for homozygous alternate individuals
1000	1.732482e-16	1.732482e-12	2.000930	5.471660
2000	9.230119e-07	9.230119e-03	2.111610	7.867450
3000	5.629213e-10	5.629213e-06	2.046087	3.334257
4000	4.770372e-07	4.770372e-03	1.641808	4.201993

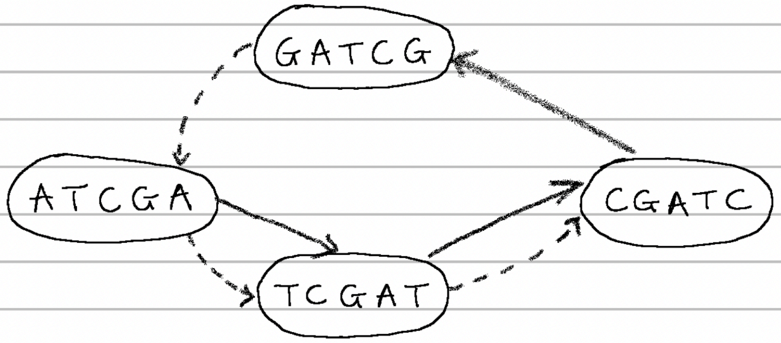


```
  homo_alt = contingency_table.loc[2, 1] / contingency_table.loc[2, 0] if 2 in contingency_tab
          SNP  uncorrected_p_value  corrected_p_value  odds_ratio_het  odds_ratio_homo_alt
1000  SNP1000         1.732482e-16       1.732482e-12        2.000930             5.471660
2000  SNP2000         9.230119e-07       9.230119e-03        2.111610             7.867450
3000  SNP3000         5.629213e-10       5.629213e-06        2.046087             3.334257
4000  SNP4000         4.770372e-07       4.770372e-03        1.641808             4.201993
(myenv) aditipotnis@Aditis-Air ~ % 
```

d. Even if two SNPs have similar disease odds ratios, the chi-squared value will depend on **the actual counts** in each cell of the contingency table, **the expected counts** under the null hypothesis, and how evenly distributed the genotypes are among diseased and non-diseased individuals. If one SNP has a better representation in each genotype category, the observed data may deviate less from the expected, resulting in a smaller chi-squared value and thus a higher p-value.

Q.3) Take G = GATCGATC

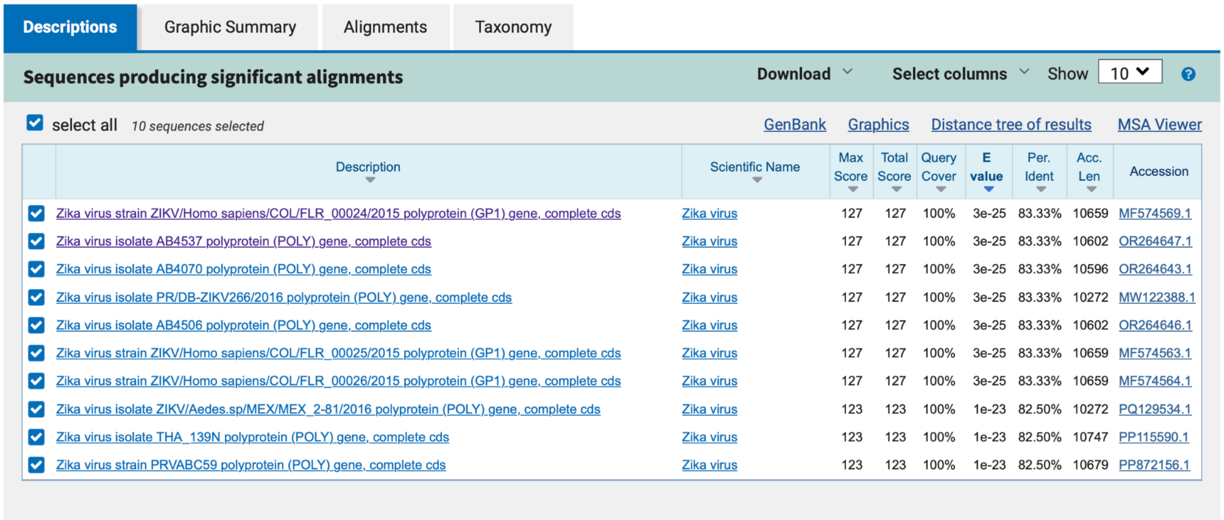
length = 8



G' = ATCGATCG length = 8

Q.4.

a. According to the BLAST results it is clear that the infection is a Zika Virus.



Sequences producing significant alignments

10 sequences selected

	Description	Scientific Name	Max Score	Total Score	Query Cover	E value	Per. Ident	Acc. Len	Accession
☑	Zika virus strain ZIKV/Homo sapiens/COL/FLR_00024/2015 polyprotein (GP1) gene, complete cds	Zika virus	127	127	100%	3e-25	83.33%	10659	MF574569.1
☑	Zika virus isolate AB4537 polyprotein (POLY) gene, complete cds	Zika virus	127	127	100%	3e-25	83.33%	10602	OR264647.1
☑	Zika virus isolate AB4070 polyprotein (POLY) gene, complete cds	Zika virus	127	127	100%	3e-25	83.33%	10596	OR264643.1
☑	Zika virus isolate PR/DB-ZIKV266/2016 polyprotein (POLY) gene, complete cds	Zika virus	127	127	100%	3e-25	83.33%	10272	MW122388.1
☑	Zika virus isolate AB4506 polyprotein (POLY) gene, complete cds	Zika virus	127	127	100%	3e-25	83.33%	10602	OR264646.1
☑	Zika virus strain ZIKV/Homo sapiens/COL/FLR_00025/2015 polyprotein (GP1) gene, complete cds	Zika virus	127	127	100%	3e-25	83.33%	10659	MF574563.1
☑	Zika virus strain ZIKV/Homo sapiens/COL/FLR_00026/2015 polyprotein (GP1) gene, complete cds	Zika virus	127	127	100%	3e-25	83.33%	10659	MF574564.1
☑	Zika virus isolate ZIKV/Aedes.sp/MEX/MEX_2-81/2016 polyprotein (POLY) gene, complete cds	Zika virus	123	123	100%	1e-23	82.50%	10272	PQ129534.1
☑	Zika virus isolate THA_139N polyprotein (POLY) gene, complete cds	Zika virus	123	123	100%	1e-23	82.50%	10747	PP115590.1
☑	Zika virus strain PRVABC59 polyprotein (POLY) gene, complete cds	Zika virus	123	123	100%	1e-23	82.50%	10679	PP872156.1

100% Query Cover means that your entire query sequence was aligned to the Zika virus genome. This indicates that the given infected section of the sequence matches the full length of these Zika virus sequences. The **E-value** is **3e-25** for most of the matches, which is extremely low. This indicates that the probability of this alignment occurring by chance is negligible. In other words, the match is highly significant and very unlikely to be random.

b. **Position 6–64**: The sequence comparisons highlight where the strains have identical nucleotides and where they differ. The given patient's strain (MF574569.1) appears to be **most similar to strain 4** for a large portion of the sequence. It is also quite similar to strain 5 but with more differences.

Tree Rendering results

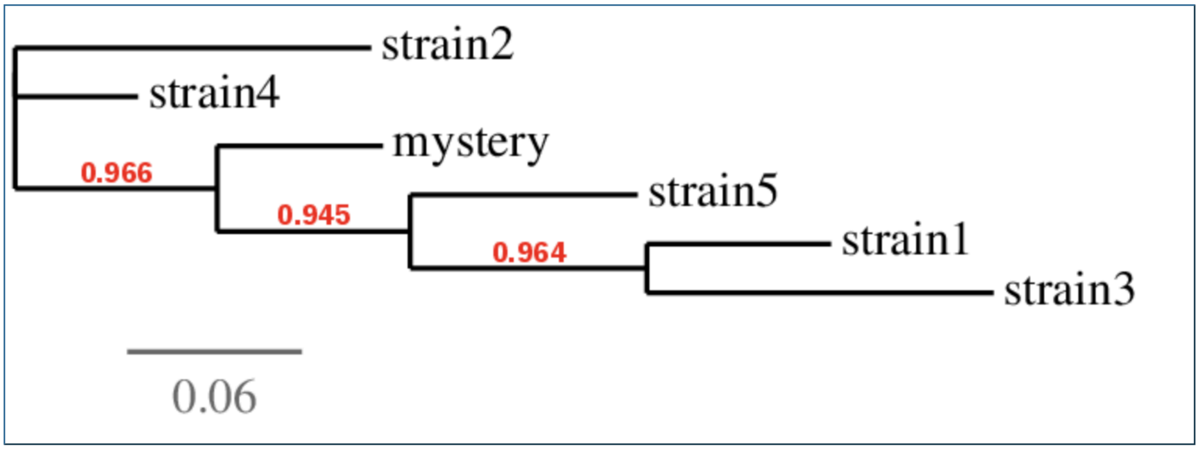
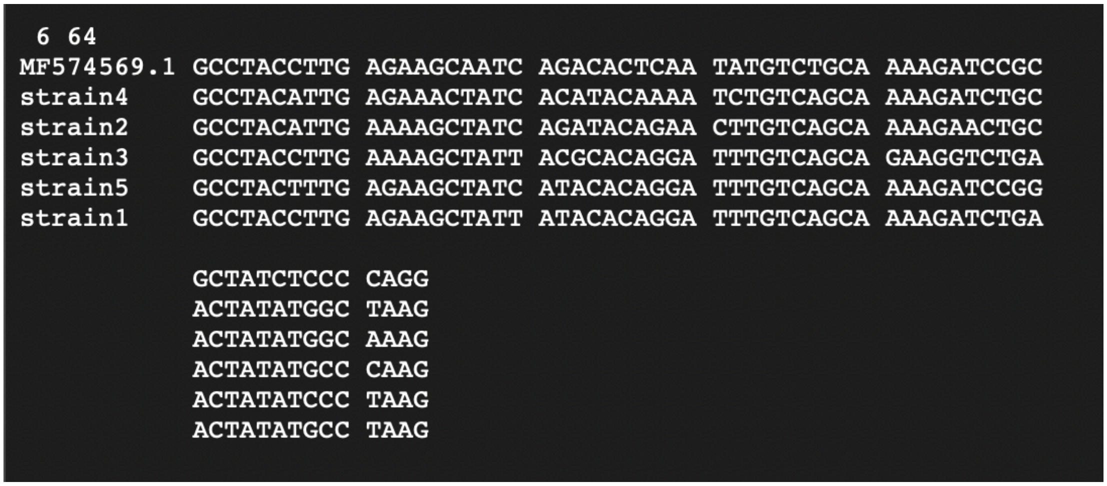


Figure 1: *Phylogenetic tree (the branch length is proportional to the number of substitutions per site).*



```
 6 64
MF574569.1 GCCTACCTTG AGAAGCAATC AGACACTCAA TATGTCTGCA AAAGATCCGC
strain4    GCCTACATTG AGAAACTATC ACATACAAAA TCTGTCAGCA AAAGATCTGC
strain2    GCCTACATTG AAAAGCTATC AGATACAGAA CTTGTCAGCA AAAGAACTGC
strain3    GCCTACCTTG AAAAGCTATT ACGCACAGGA TTTGTCAGCA GAAGGTCTGA
strain5    GCCTACTTTG AGAAGCTATC ATACACAGGA TTTGTCAGCA AAAGATCCGG
strain1    GCCTACCTTG AGAAGCTATT ATACACAGGA TTTGTCAGCA AAAGATCTGA

           GCTATCTCCC CAGG
           ACTATATGGC TAAG
           ACTATATGGC AAAG
           ACTATATGCC CAAG
           ACTATATCCC TAAG
           ACTATATGCC TAAG
```

Q.5.

a.

```
def small_parsimony_tree(T, D, k, M):
  // given that:
  // T: A tree with n leaves and m internal nodes
  // D: Matrix (n x k) where each row is the k-dimensional trait vector for a species
  // M: Maximum possible value for any trait
  n = len(D)  # number of leaves (species)

  // Step 1: Initialize dynamic programming table
  DP = {}  # DP[u][i] will store minimum cost at node u for trait i
  Traceback = {}  # Traceback[u][i] will store the best trait value for internal node u, trait i

  // Step 2: Assign leaf nodes with given trait vectors
  for leaf in T.leaves():
    DP[leaf] = {}
    for i in range(k):
      DP[leaf][i] = {D[leaf][i]: 0}  # No cost at leaves, trait values are fixed

  // Step 3: Bottom-up dynamic programming
  for node in T.postorder_traversal():
    if node not in T.leaves():
      DP[node] = {}
      Traceback[node] = {}
      for i in range(k):  # Iterate over each trait
        DP[node][i] = {}
        min_cost = float('inf')
```

```
            for val in range(M + 1):  # Iterate over all possible trait values for this node
                cost = 0
                for child in T.children(node):
                    # Find minimum cost of assigning val to current node's trait i
                    min_child_cost = float('inf')
                    best_val = -1
                    for child_val in DP[child][i]:
                        current_cost = DP[child][i][child_val] + abs(val - child_val)
                        if current_cost < min_child_cost:
                            min_child_cost = current_cost
                            best_val = child_val
                    cost += min_child_cost
                    Traceback[node][i] = best_val

                DP[node][i][val] = cost  # Store the minimum cost for node's trait i having value 'val'
```

// Step 4: Traceback to find the best trait assignment for internal nodes

```
Du = {}  # Du will store the best trait vector for each internal node
root = T.root()
Du[root] = [0] * k

# Initialize the root's trait vector based on DP values
for i in range(k):
    best_val = min(DP[root][i], key=DP[root][i].get)  # Best trait value for trait i
    Du[root][i] = best_val

# Traceback to assign values to other internal nodes
for node in T.preorder_traversal():
    if node != root:
        parent = T.parent(node)
        Du[node] = [0] * k
        for i in range(k):
            Du[node][i] = Traceback[parent][i]
```

// Step 5: Return the trait vector assignment for each internal node

```
return Du, DP[root]
```

b. Time complexity can be derived by analyzing the 2 steps:

- For the tree traversal the runtime will be O(n), as the tree is traversed twice (during DP and traceback) and the tree has n nodes (n species).
- There are k traits.. For each node and each trait, you have O(M^2) operations (forward and backward passes through possible trait values).

With these considerations the time complexity is O(n * k * M^2) assuming M is significant enough to consider else O(n * k).