1.

a)

Step 1 -Remove all loops and Parallel Edges

There is no loop or parallel edges in this graph.

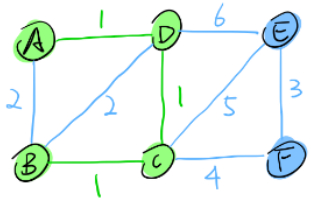
Step 2 - Arrange all edges in their increasing order of weight

A, D	C, D	B, C	A, B	B, D	E, F	C, F	C, E	D, E
1	1	1	2	2	3	4	5	6

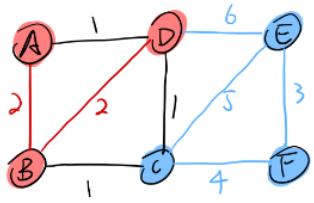
Step 3 - Add the edge which has the least weightage

Now we start adding edges to the graph beginning from the one which has the least weight. Throughout, we shall keep checking that the spanning properties remain intact. In case, by adding one edge, the spanning tree property does not hold then we shall consider not to include the edge in the graph.

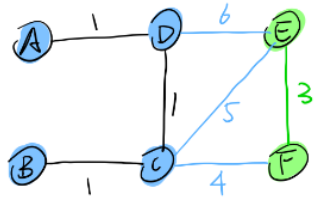
The least cost is 1 and edges involved are (A, D), (C, D) and (B, C). We add them. Adding them does not violate spanning tree properties, so we continue to our next edge selection.



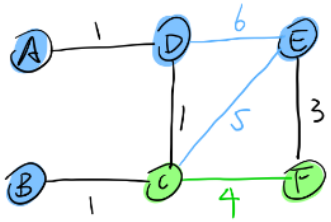
Next cost is 2, and we observe that adding them will create circuits in the graph. So we ignore them.



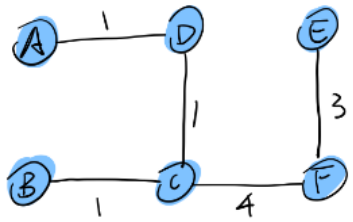
Next cost is 3, and associated edge is (E, F). We add it again.



Next cost is 4, and associated edge is (C, F). We add it again.

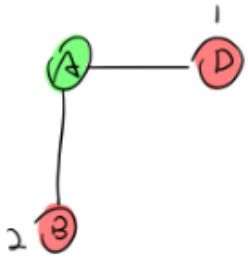


By adding edge (C, F) we have included all the nodes of the graph and we now have minimum cost spanning tree:

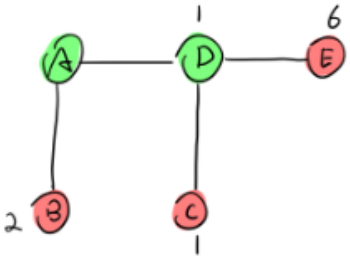


b)

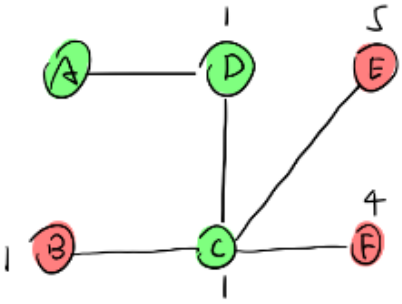
The set mstSet is initially empty and keys assigned to vertices are {0, INF, INF, INF, INF, INF} where INF indicates infinite. Now pick the vertex with the minimum key value. The vertex “A” is picked, include it in mstSet. So mstSet becomes {“A”}. After including to mstSet, update key values of adjacent vertices. Adjacent vertices of “A” are “B” and “D”. The key values of “B” and “D” are updated as 2 and 1. Following subgraph shows vertices and their key values, only the vertices with finite key values are shown. The vertices included in MST are shown in green color.



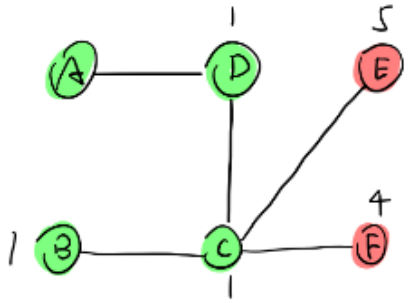
Pick the vertex with minimum key value and not already included in MST (not in mstSET). The vertex “D” is picked and added to mstSet. So mstSet now becomes {“A”, “D”}. Update the key values of adjacent vertices of “D”. The key value of vertex “C” becomes 1, and the key value of vertex “E” becomes 6.



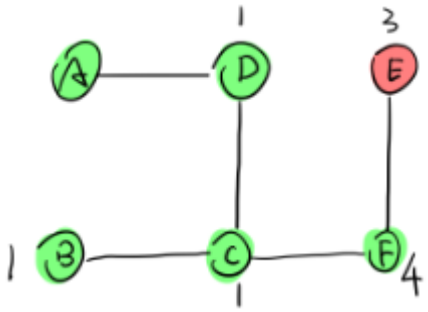
Pick the vertex with minimum key value and not already included in MST (not in mstSET). The vertex “C” is picked and added to mstSet. So mstSet now becomes {“A”, “D”, “C”}. Update the key values of adjacent vertices of “C”. The key value of vertex “F” becomes 4, the key value of vertex “B” becomes 1, and the key value of vertex “E” becomes 5.



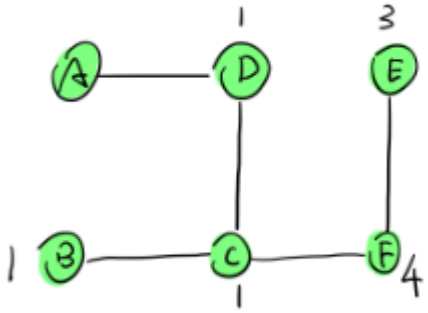
Pick the vertex with minimum key value and not already included in MST (not in mstSET). The vertex “B” is picked and added to mstSet. So mstSet now becomes {“A”, “D”, “C”, “B”}. Update the key values of adjacent vertices of “B”.



Pick the vertex with minimum key value and not already included in MST (not in mstSET). The vertex “F” is picked and added to mstSet. So mstSet now becomes {“A”, “D”, “C”, “B”, “F”}. Update the key values of adjacent vertices of “F”. The key value of vertex “E” becomes 3.



Pick the vertex with minimum key value and not already included in MST (not in mstSET). The vertex “E” is picked and added to mstSet. So mstSet now becomes {“A”, “D”, “C”, “B”, “F”, “E”}. Finally, we get the following graph.



e)

Algorithm	Time complexity	Space complexity
Kruskal’s	$O(E * \log(V))$	$O(E + V)$
Prim’s	$O(E * \log(V))$	$O(E + V)$

2.

Cell:

Cells are the basic building blocks of all living things. The human body is composed of trillions of cells. They provide structure for the body, take in nutrients from food, convert those nutrients into energy, and carry out specialized functions. Cells also contain the

body's hereditary material and can make copies of themselves.

Gene:

A gene is the basic physical and functional unit of heredity. Genes are made up of DNA. Some genes act as instructions to make molecules called proteins. However, many genes do not code for proteins.

Chromosome:

In the nucleus of each cell, the DNA molecule is packaged into thread-like structures called chromosomes. Each chromosome is made up of DNA tightly coiled many times around proteins called histones that support its structure.

DNA:

DNA, or deoxyribonucleic acid, is the hereditary material in humans and almost all other organisms.

Human Genome Project:

An international research effort called the Human Genome Project, which worked to determine the sequence of the human genome and identify the genes that it contains, estimated that humans have between 20,000 and 25,000 genes.

Cell Nucleus:

The nucleus is an organelle found in eukaryotic cells. Inside its fully enclosed nuclear membrane, it contains the majority of the cell's genetic material. This material is organized as DNA molecules, along with a variety of proteins, to form chromosomes.

Genomic Language:

The DNA molecule carries information in the form of a sequence of four nucleotide bases, adenine (A), cytosine (C), guanine (G) and thymine (T), which can be thought of as the letters of the genomic language. Short sequences of the letters form 'DNA words' that determine when and where proteins are made in the body.

DNA Mutations:

DNA mutations are permanent changes in the DNA sequence of a gene. Mutations range in their severity. Some damage the way a cell or whole organism functions, or even cause lethality, while others have no effect. Mutations also range in the amount of DNA altered. They can involve from a single nucleotide up to large segments of chromosomes.

Three mutations:

i. Insertion/Duplication/Deletion: the addition or subtraction of nucleotides from DNA sequence. They can be as small as single nucleotides or large enough to visualize on a chromosome and involve tens to hundreds of thousands of nucleotides.

ii. Point Mutation: the change in one nucleotide for another. For example, an "A" becomes a "T".

iii. Translocation: the movement of a segment of DNA from one chromosome to another.

3.

a)

Cancer is a genetic disease—that is, it is caused by changes in DNA that control the way cells function, especially how they grow and divide. These changes can be inherited, but most arise randomly during a person's lifetime, either as a result of errors that occur as cells divide or from exposure to DNA-damaging carcinogens. Each person's cancer has a unique

combination of genetic changes, and tumor DNA sequencing—sometimes called genetic profiling or genetic testing—is a test to identify these unique DNA changes.

b)

EGFR gene. Mutations in the EGFR gene that make cells divide rapidly are found in some people's lung cancer cells. A patient whose lung cancer cells harbor an EGFR mutation may respond to treatment with drugs called EGFR inhibitors. Clinical tumor DNA sequencing can reveal whether a patient's lung tumor has an EGFR mutation.

c)

EGFR gene. Cancer is a genetic disease—that is, it is caused by changes in DNA that control the way cells function, especially how they grow and divide. These changes can be inherited, but most arise randomly during a person's lifetime, either as a result of errors that occur as cells divide or from exposure to DNA-damaging carcinogens.

4.

A)

Cells are the basic building block of all living things.

Therefore, in each cell, there is the same set of chromosomes. Chromosome are basically the strings of DNA.

B)

i. Firstly, we defined our initial population as our countrymen.

ii. We defined a function to classify whether is a person is good or bad.

iii. Then we selected good people for mating to produce their off-springs.

iv. And finally, these off-springs replace the bad people from the population and this process repeats.

This is how genetic algorithm actually works, which basically tries to mimic the human evolution to some extent.

So to formalize a definition of a genetic algorithm, we can say that it is an optimization technique, which tries to find out such values of input so that we get the best output values or results.

C)

True.

For an optimization problem, a certain number of candidate solutions (called individuals) can be abstractly represented as chromosomes, which makes the population evolve to a better solution.

5.

A)

Initial population – Fitness function – selection – crossover – mutation – termination

B)

The code creates two classes which are individual and population. It first calls initializePopulation function to initial population with size 10. The fitness function is realized through the calcFitness function in the individual class. The selection is realized through the selection function which selects the two fittest individuals. The crossover is realized through the crossover function. It randomly selects crossover points and swap values among parents.

The mutation is realized through the mutation function which selects a random mutation point and flip values at the mutation point. The termination point of the algorithm is when population gets an individual with maximum fitness which is realized in the 31 line of the code.

C)

```
Generation: 8 Fittest: 3
Generation: 9 Fittest: 3
Generation: 10 Fittest: 3
Generation: 11 Fittest: 3
Generation: 12 Fittest: 3
Generation: 13 Fittest: 3
Generation: 14 Fittest: 3
Generation: 15 Fittest: 2
Generation: 16 Fittest: 4
Generation: 17 Fittest: 2
Generation: 18 Fittest: 3
Generation: 19 Fittest: 3
Generation: 20 Fittest: 5

Solution found in generation 20
Fitness: 5
Genes: 11111
```

The result is the fittest individual in the last generation, with 11111 genes. The fitness of the individual is 5. The execution stops when the fittest individual appears in the latest generation.

D)

When population gets an individual with maximum fitness.

F)

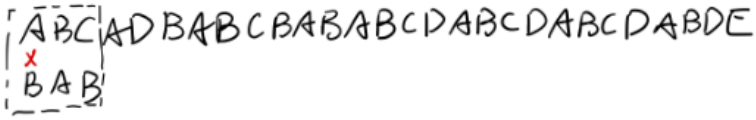
The average generation that is needed to get the fittest individual increases. That is because the number of genes increases which means there are more possibilities in selection, mutation and crossover, which means it is harder to get the fittest individual.

6.

a) Brute-Force:

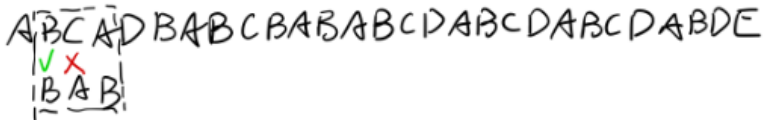
Compares the pattern with the text one by one.

First compares the first three chars.



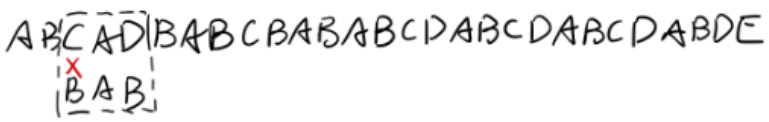
‘A’ doesn’t match ‘B’, so move to next three chars.

Compares next three chars.

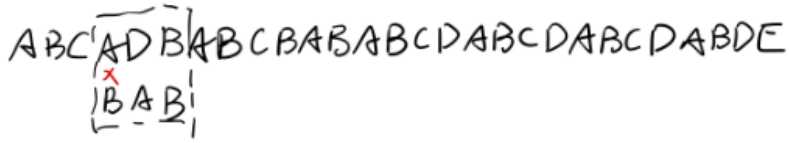


‘B’ matches ‘B’, but ‘C’ doesn’t match ‘A’, so move to next three chars.

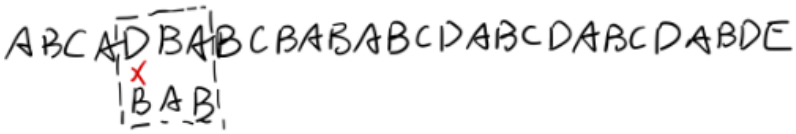
Compares next three chars.



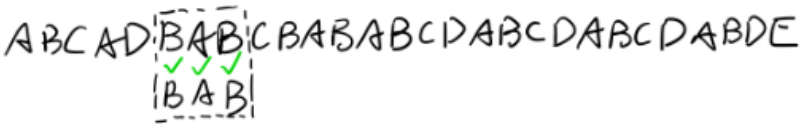
‘C’ doesn’t match ‘B’, so move to next three chars.
Compares next three chars.



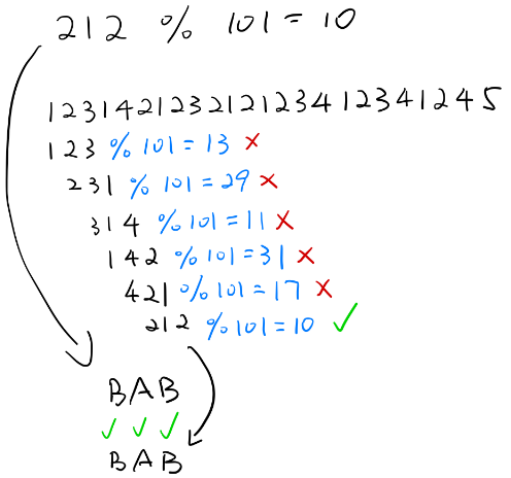
‘A’ doesn’t match ‘B’, so move to next three chars.
Compares next three chars.



‘D’ doesn’t match ‘B’, so move to next three chars.
Compares next three chars.



‘B’ matches ‘B’, ‘A’ matches ‘A’, ‘B’ matches ‘B’.
Repeat the steps until the end of the text.
Then we got 5, 9, 11 which means the pattern is found at 5, 9, 11 of the text.
b)
To make the example easier to present, q is set to 10 and A is 1, B is 2, C is 3, D is 4, E is 5.
We first get the hash code for the pattern and the first possible subset of the text. In this case, the hash code of 212 is $\big((2 * \mathrm{d} + 1) * \mathrm{d} + 2\big)\%\mathrm{q} = 10$. The hash code of 123 is $\big((1 * \mathrm{d} + 2) * \mathrm{d} + 3\big)\%\mathrm{q} = 13$. 10 doesn’t match 13, so move on to the next subset. Repeat until pattern matches the subset. As shown in the picture, we then find out the original subset that corresponds to the hash code which is ‘BAB’. Then we compare ‘BAB’ with the pattern ‘BAB’ which are the same.



Repeat the steps until the end of the text.
Then we got 5, 9, 11 which means the pattern is found at 5, 9, 11 of the text.
c)

O(mn) in Rabin-Karp is much faster than O(nm) in brute force. The reason is that comparing Pattern letters with Text letters is expensive operation. With Robin-Karp method you only do comparisons if hash(pattern) == hash(substring), otherwise you Don't do any comparison.

7.

a)

Let the given source vertex be 0. Initialize all distances as infinite, except the distance to the source itself. Total number of vertices in the graph is 5, so all edges must be processed 4 times.

A	B	C	D	E
0	∞	∞	∞	∞

Let all edges are processed in the following order: (B, E), (D, B), (B, D), (A, B), (A, C), (D, C), (B, C), (E, D). We get the following distances when all edges are processed the first time. The first row shows initial distances. The second row shows distances when edges (B, E), (D, B), (B, D) and (A, B) are processed. The third row shows distances when (A, C) is processed. The fourth row shows when (D, C), (B, C) and (E, D) are processed.

A	B	C	D	E
0	∞	∞	∞	∞
0	-1	∞	∞	∞
0	-1	4	∞	∞
0	-1	2	∞	∞

The first iteration guarantees to give all shortest paths which are at most 1 edge long. We get the following distances when all edges are processed second time (The last row shows final values).

A	B	C	D	E
0	∞	∞	∞	∞
0	-1	∞	∞	∞
0	-1	4	∞	∞
0	-1	2	∞	∞
0	-1	2	∞	1
0	-1	2	1	1
0	-1	2	-2	1

The second iteration guarantees to give all shortest paths which are at most 2 edges long. The algorithm processes all edges 2 more times. The distances are minimized after the second iteration, so third and fourth iterations don't update the distances.

b)

i) It means the more you go through the cycle the shorter the path is from source to the destination. The shortest path from source to the destination becomes infinitely small.

ii) While finished the V-1 cycle, add another cycle to see whether there is a change in the distance of all the vertices. If there is a change, negative cycle detected.

8.

a)

Sort the array A in ascending order: A={1, 2, 3, 4, 5, 6, 7, 8}

pick the first element in A: 1<15
Then pick the second element in A: 1+2=3<15
Then pick the third element in A: 1+2+3=6<15
Then pick the fourth element in A: 1+2+3+4=10<15
Then pick the fifth element in A: 1+2+3+4+5=15=15
Then pick the sixth element in A: 1+2+3+4+5+6=21>15
So 5 things could be done at most.
c)
When it comes to scheduling problem, each case is given a priority. For example, in this case, the priority could be P={6, 3, 8, 2, 1, 7, 4, 5}.
For each element in P and A, do P[i]/A[i]: P/A={6, 1.5, 2.7, 0.5, 0.2, 1.2, 0.57, 0.63}
Sort the new array: {6, 2.7, 1.5, 1.2, 0.63, 0.57, 0.5, 0.2} which corresponds to A={1, 3, 2, 6, 8, 7, 4, 5}
Then the we should first pick 1, then 3, then 2, then 6, so on and so forth.
d)
(a) is a special case of (c) while all of the work has the same priority. So we just need to finish the work that takes the least time. But in (c), we need to take the priority into consideration to judge which work to do first.

9.
B) What elements in Algorithm-1 relates to Genetic algorithm, explain
"pop": the population size
"maxgen": The maximum number of generations
"pc": the crossover probability
"pm": the mutation probability
function "crossover": the function to get offspring from two parents.
function "mutation": the function to get one offspring from one individual.
function "select": the function to randomly choose individuals or chromosome.
C) How do you initialize population
I created a population class to collect all the individuals which are spanning trees in this case. Then I create new random spanning trees repeatedly and use a function to judge them whether they meet the requirement of "α". Once a tree meets the requirement, add it to the population object. Repeat until there are "pop" individuals in the population object.
D) Read the Abstract and explain what the paper intends to accomplish
The paper is trying to use genetic algorithm to find the balanced spanning tree, which the distance of all of the vertices from the source is smaller than α * minimum distance, that has the smallest total weight.
D) Discuss your Input Data, Data Structures, and Outputs
Input:
int[][] graph, which is the original graph.
int pop, which is the population size.
int maxgen, which is the maximum generation.
double pc, which is the crossover possibility.
double pm, which is the mutation possibility

int[] Ts, which is the distance of each node of shortest path tree
Tree Tm, which is the minimum spanning tree.
Output:
The best individual, which has the least total weight, of the last generation.