Business Cases

Name:Amulya B Chougale

Roll No.:552

USN: 01FE24BCS416

1. Occurrence of Drought

Sub task: Provide optimal path for the water supply.

SDG: 11

Target: 11.6

Indicator: 11.6.1

There is a possibility of having periodic droughts which could affect several factors of the city like Agricultural productivity, Industrial area, Wildlife sanctuary.

Hence, the problem could be addressed with a solution by using multiple algorithms and techniques.

1.1. Divide and Conquer Approach and BFS

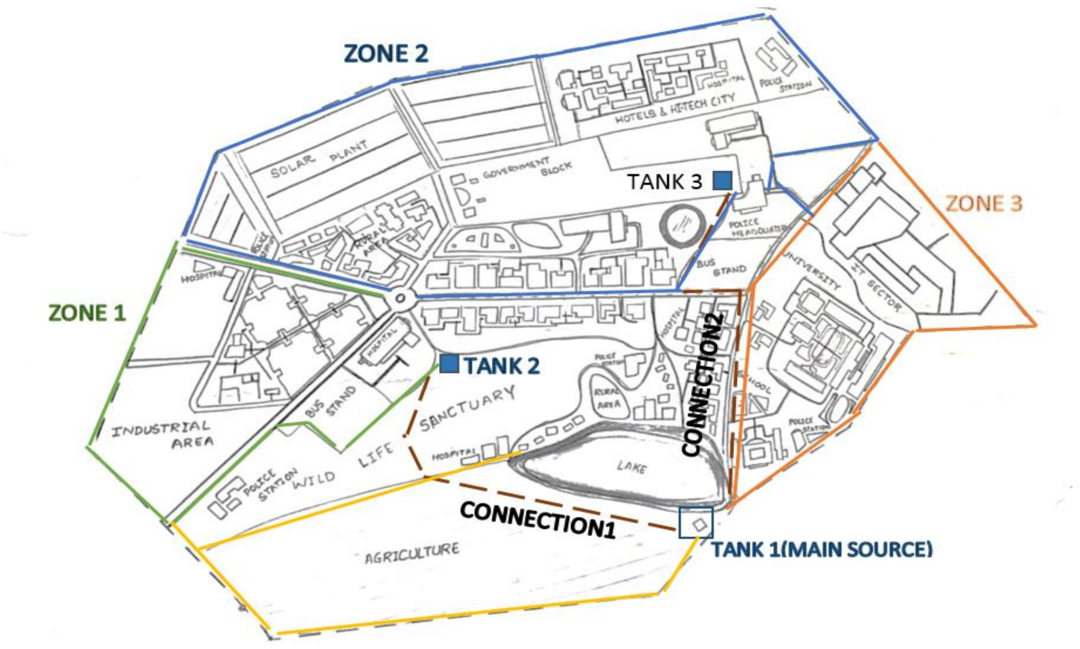


Fig.Divide and Conquer Approach to divide the areas of the city into smaller zones and then providing them with the water supply.

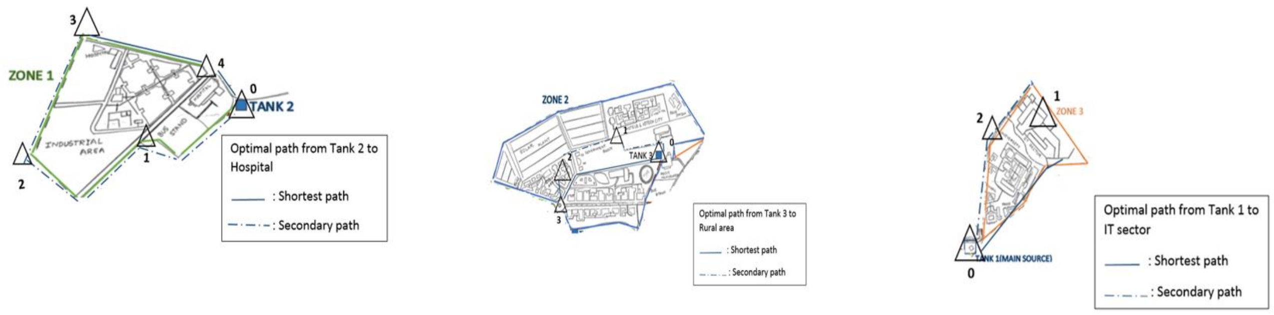


Fig.BFS traversal for the zones

The optimal path is prepared for the city using BFS. The code is implemented using the adjacency matrix,queue for the graph representation.The overall time complexity of the algorithm is O((V + E) log V), and the space complexity is O(V + E).

View Code

Output:

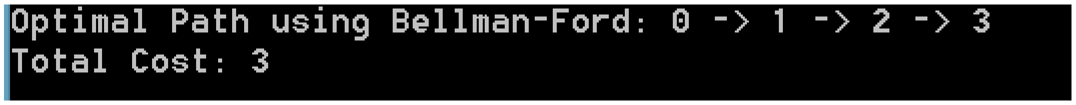
```
The optimal path for Zone 1 is: Tank2 -> Residential Area -> Hospital
The optimal path for Zone 2 is: Tank3 -> Rural Area
The optimal path for Zone 3 is: Tank1 -> IT Sector
```

1.2. Bellman-Ford's Algorithm

The Bellman-Ford algorithm efficiently finds the shortest path in the graph, ensuring reliable and optimal water supply distribution during periodic droughts. With a time complexity of O(V×E) the algorithm effectively handles negative edge weights and potential negative cycles. The primary data structures used are an adjacency list, which represents the graph, and distance and path arrays, which store the shortest distances and parent nodes, respectively.

View Code

Output:



```
Optimal Path using Bellman-Ford: 0 -> 1 -> 2 -> 3
Total Cost: 3
```

2. Public Transportation Management

Sub task: Optimizing Transportation Hub Placement

SDG: 11

Target: 11.2

Indicator: 11.2.1

Develop an optimal path from one location to another to ensure efficient travel that minimizes time and cost.
For example, creating a route from the Industrial Area to the area of Hotels and Hi-Tech City. This is essential because industries often have clients visiting for business purposes, and these clients require services from hotels located at a considerable distance. Ensuring an optimal travel path will enhance connectivity and convenience.

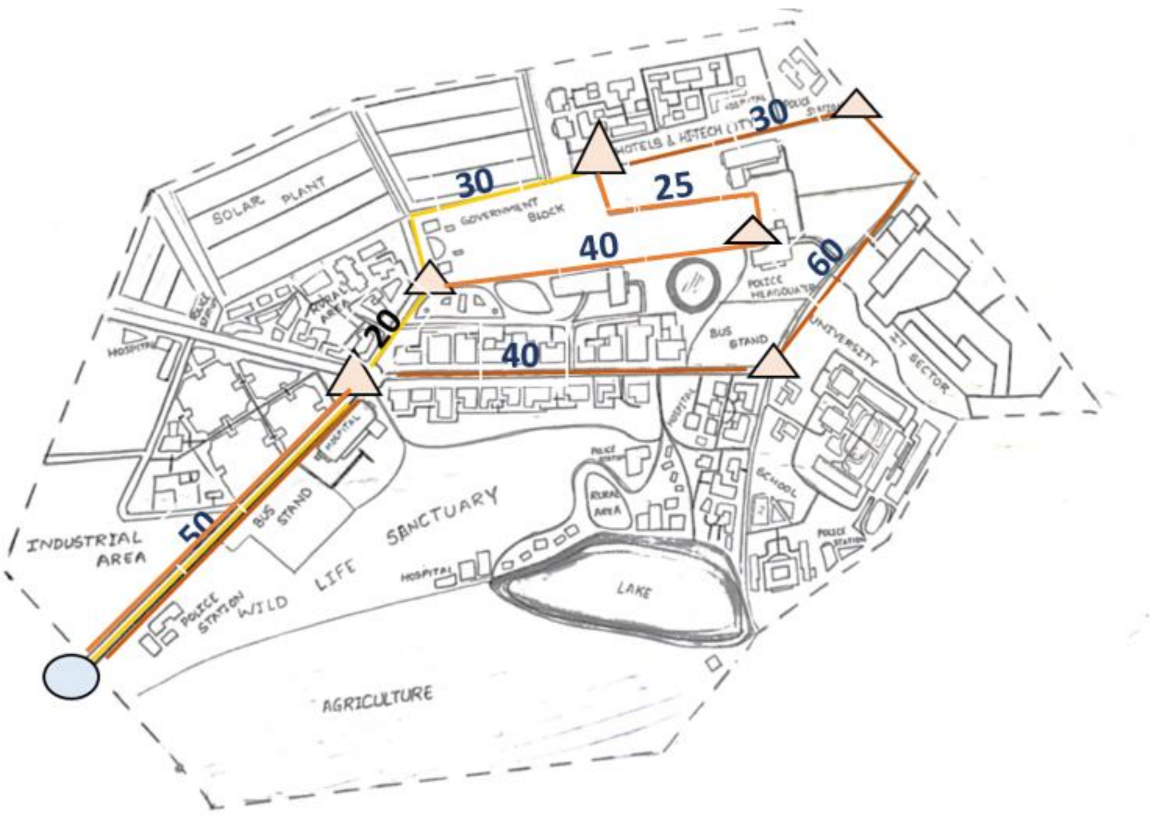


Fig.Mapping of the possible routes from Industrial Area to Hotels &Hi-Tech city

There are three potential routes from the Industrial Area to the Hotels. Therefore, the paths can be analyzed as follows:

Path 1: Industrial Area -> City Circle -> Rural Area -> Hotels

Path 2: Industrial Area -> City Circle -> Bus Stand -> Police Station -> Hotels

Path 3: Industrial Area -> City Circle -> Rural Area -> Police Headquarters -> Hotels

The table below provides a detailed comparison of three potential paths from the Industrial Area to the Hotels. Each path is described by the route it takes, the travel costs between each point along the route, and the total cost of traveling the entire path.

Path	Route	Travel Cost	Total Cost
Path 1	Industrial Area -> City Circle -> Rural Area -> Hotels	Industrial Area to City Circle: ₹50 City Circle to Rural Area: ₹20 Rural Area to Hotels: ₹30	₹50 + ₹20 + ₹30 =₹100
Path 2	Industrial Area -> City Circle -> Bus Stand -> Police Station -> Hotels	Industrial Area to City Circle: ₹50 City Circle to Bus Stand : ₹40 Bus Stand to Hotels: ₹60 Police Station to Hotels: ₹20	₹50 + ₹40 + ₹60+₹20 =₹170
Path 3	Industrial Area -> City Circle -> Rural Area -> Police Headquarters -> Hotels	Industrial Area to City Circle: ₹50 City Circle to Rural Area: ₹20 Rural Area to Police Headquarters: ₹40 Police Headquarters to Hotels: ₹25	₹50 + ₹20 + ₹40 + ₹25 = ₹135

2.1The optimal path among the three can be depicted using Dijakstra's algorithm as:

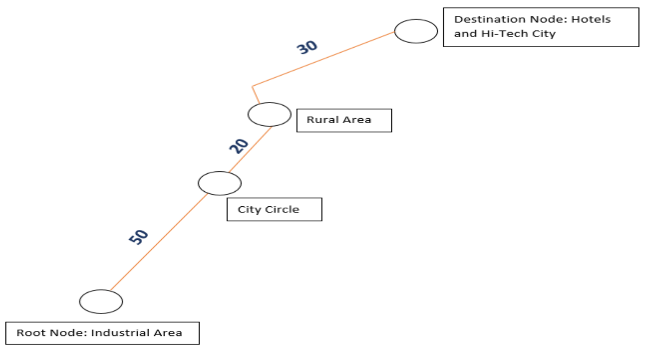
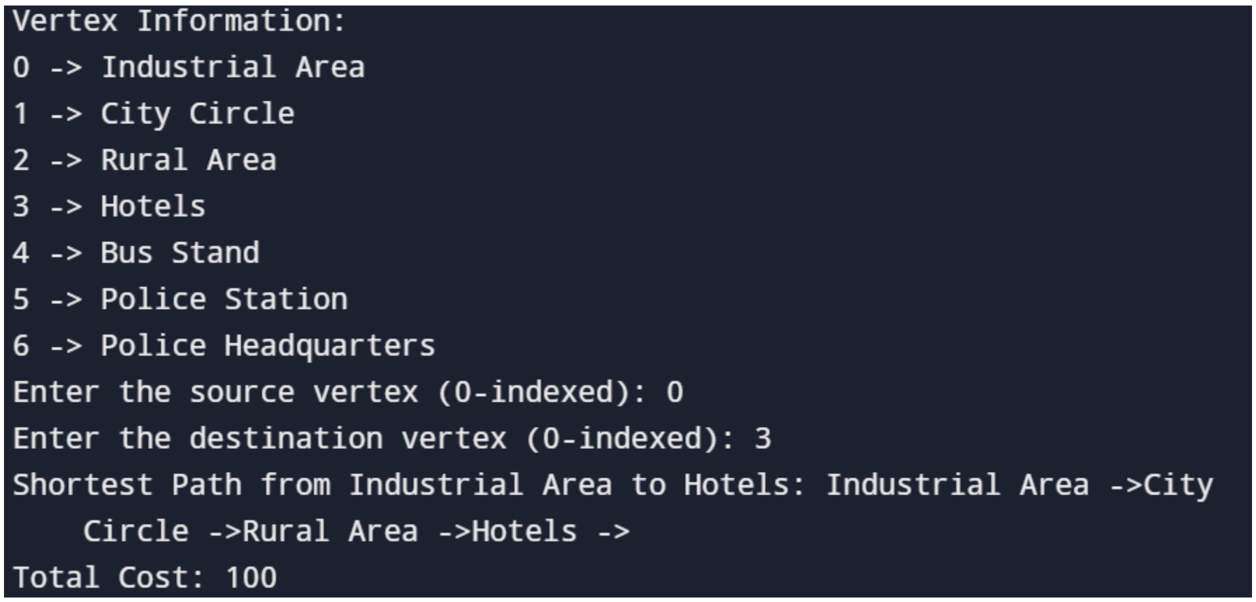

Destination Node: Hotels and Hi-Tech City
30
Rural Area
20
City Circle
50
Root Node: Industrial Area

Fig.Implementation using Dijakstra's algorithm

The optimal path is prepared for the city using Dijakstra's Algorithm. The code is implemented using the adjacency list for the graph representation and a priority queue (min-heap) for efficiently retrieving the node with the smallest distance. The overall time complexity of the algorithm is O((V + E) log V), and the space complexity is O(V + E) and the algorithm uses greedy technique.

View Code

Output:



```
Vertex Information:
0 -> Industrial Area
1 -> City Circle
2 -> Rural Area
3 -> Hotels
4 -> Bus Stand
5 -> Police Station
6 -> Police Headquarters
Enter the source vertex (0-indexed): 0
Enter the destination vertex (0-indexed): 3
Shortest Path from Industrial Area to Hotels: Industrial Area ->City
    Circle ->Rural Area ->Hotels ->
Total Cost: 100
```

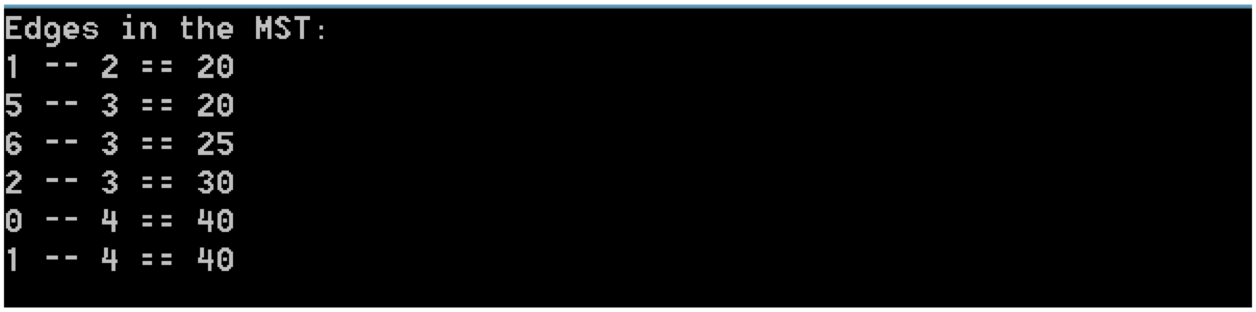
2.2. The optimal path among the three can be depicted using Kruskal’s algorithm as:

```
  Industrial Area
       |
      50
       |
   City Circle
       |
      20
       |
   Rural Area ----- 40 ----- Police Headquarters
       |                   |
      30                  25
       |                   |
     Hotels -------------------- 20 --------------- Police Station
```

From the above calculations, it is clear that the optimal path is through the Rural Area with a total travel cost of ₹100.

View Code

Output:



```
Edges in the MST:
1 -- 2 == 20
5 -- 3 == 20
6 -- 3 == 25
2 -- 3 == 30
0 -- 4 == 40
1 -- 4 == 40
```

3. Inventory Management System

3.1. Sub task: Optimizing Inventory Management

SDG: 11

Target: 11.6

Indicator: 11.6.1

Managing inventory for retail stores across the city to ensure that products are available when needed, thereby reducing waste and improving operational efficiency. For instance, ensuring that perishable goods in grocery stores are prioritized based on expiration dates and demand, and managing stock levels of medical supplies in healthcare facilities efficiently.

Inventory can be sorting using various sorting algorithms like quick sort,merge sort,etc.

1. Sorting of inventory using quick sort

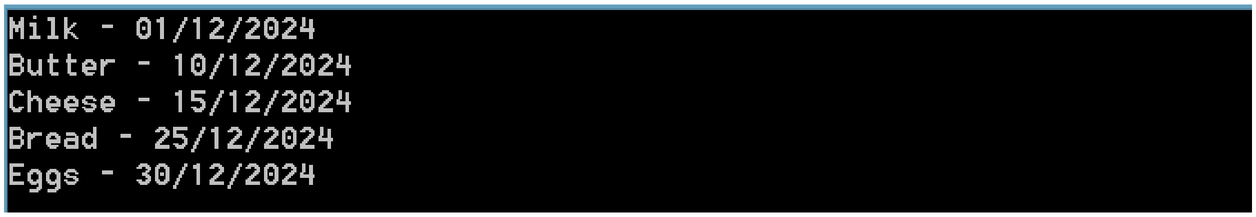
The inventory items are sorted based on attribute such as expiration dates. This helps in identifying and prioritizing items that need immediate attention, reducing waste due to expired or unsold products.

Item	Stock Level
Milk	01/12/2024
Bread	25/12/2024
Cheese	15/12/2024
Butter	10/12/2024
Eggs	30/12/2024

The Quick Sort algorithm, used for sorting by expiration dates, has a time complexity of O(n log n) on average. The primary data structure used is vectors and divide and conquer technique has been used.

View Code

Output:



```
Milk - 01/12/2024
Butter - 10/12/2024
Cheese - 15/12/2024
Bread - 25/12/2024
Eggs - 30/12/2024
```

2. Sorting inventory using merge sort

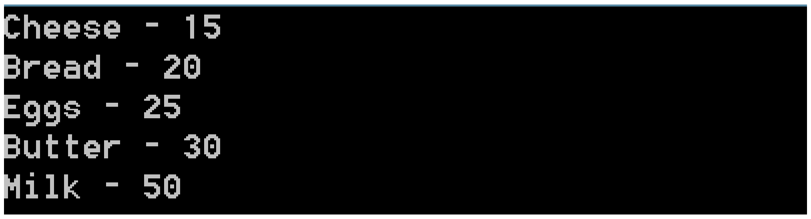
The Inventory items are sorted based on stock levels. This helps in organizing inventory efficiently, ensuring that items with lower stock levels are prioritized for restocking.

Item	Stock Level
Milk	50
Bread	20
Cheese	15
Butter	30
Eggs	25

The Merge Sort algorithm, used for sorting by stock levels, guarantees a time complexity of O(n log n) in all cases. The primary data structure used is vectors and divide and conquer has been used.

View Code

Output:



```
Cheese - 15
Bread - 20
Eggs - 25
Butter - 30
Milk - 50
```

3.2. Sub Task: Efficiently retrieving medical supplies.

SDG: 11

Target: 11.6

Indicator: 11.6.1

Locating medical supplies in healthcare facilities to ensure that items are available for patient care as needed. For instance, quickly finding the stock levels of specific items like "Bandages" or "Antibiotics" for restocking purposes.

1.The availability of Bandages has been checked using the Binary search algorithm.

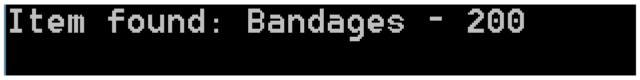
Inventory Data:

Item	Stock Level
Bandages	200
Antibiotics	150
Syringes	300
Gloves	250
Face Masks	180

The Binary Search algorithm efficiently locates the target item in the sorted inventory list, with a time complexity of O(log n). The primary data structure used is a vector, which holds the inventory items.

View Code

Output:



```
Item found: Bandages - 200
```

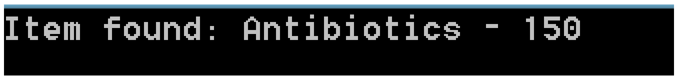
2.The same has been done using Brute Force Search algorithm for the item Antibiotics as follows:

Item	Stock Level
Bandages	200
Antibiotics	150
Syringes	300
Gloves	250
Face Masks	180

The Brute Force search algorithm sequentially checks each item in the inventory to locate the target item, with a time complexity of O(n). The primary data structure used is a vector, which holds the inventory items.

View Code

Output:



```
Item found: Antibiotics - 150
```

4. University Course Scheduling System

Sub Task: Course Scheduling

SDG: 11

Target: 11.3

Indicator: 11.3.1

Implementing course scheduling system that optimizes the allocation of classrooms, laboratories, and study spaces. This system aims to enhance operational efficiency, reduce waste, and promote a sustainable campus environment.

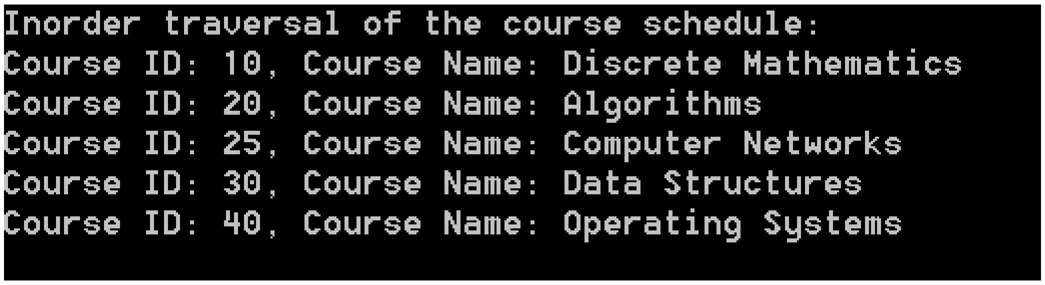
4.1.Course Scheduling Using Binary Search Tree (BST):

BST to organize and manage course schedules, allowing for efficient searching, insertion, and deletion of course schedules.

The course scheduling system for the university is optimized using a Binary Search Tree (BST). The code is implemented using nodes for the BST representation, allowing for efficient insertion, searching, and deletion of course schedules. The overall time complexity for these operations is O(log n) on average, and the space complexity is O(n). The algorithm employs a divide-and-conquer technique to manage and organize course schedules effectively.

[View Code]

Output:



```
Inorder traversal of the course schedule:
Course ID: 10, Course Name: Discrete Mathematics
Course ID: 20, Course Name: Algorithms
Course ID: 25, Course Name: Computer Networks
Course ID: 30, Course Name: Data Structures
Course ID: 40, Course Name: Operating Systems
```