# Data Structure TABA

## Name: Xiangnan Liu ID: X22190562

## Question 1

Since my student ID is **X22190562**, so the graph for me is shown as below. The shortest path is **X→L→M→Y** or **X→L→M→P→Y** and the distance is **5**. The following are the calculation steps.

图表, 雷达图

描述已自动生成

### Step 1:

In the 1st step, the starting node is **X**, and the ending nodes are **J** and **L**.

|  |  |  |  |
| --- | --- | --- | --- |
| Target node | Path |  | Total distance |
| J | X→J |  | 2 |
| L | X→L |  | 1 |
| M | - |  | ∞ |
| N | - |  | ∞ |
| P | - |  | ∞ |
| Y | - |  | ∞ |

The starting node is X, and the shortest path is **X→L** (shown as the green color), so choose the node **L** as the next starting node, and there are 2 ending nodes, **J** and **M**.

### Step 2:

|  |  |  |
| --- | --- | --- |
| Target node | Path | Total distance |
| J | X→J | 2 |
| L | X→L | 1 |
| M | X→L→M | 3 |
| N | - | ∞ |
| P | - | ∞ |
| Y | - | ∞ |

We can find that the new shortest path is X→J, so choose node **J** as the next starting node, and there are 1 ending node, **M**, and the shortest path is still X→L→M.

### Step 3:

|  |  |  |
| --- | --- | --- |
| Target node | Path | Total distance |
| J | X→J | 2 |
| L | X→L | 1 |
| M | X→L→M | 3 |
| N | - | ∞ |
| P | - | ∞ |
| Y | - | ∞ |

Then choose node **M** as the next starting node, and there are 3 ending nodes: **N**, **Y**, **P**. Choose them as the next starting nodes.

### Step 4:

|  |  |  |
| --- | --- | --- |
| Target node | Path | Total distance |
| J | X→J | 2 |
| L | X→L | 1 |
| M | X→L→M | 3 |
| N | X→L→M→N | 12 |
| P | X→L→M→P | 3 |
| Y | X→L→M→Y | 5 |

We can see that **X→L→M→P** is the new shortest path. Then choose **P** as the next starting node, and there are only 1 ending node, **Y** and its path has the same distance as X→L→M→Y.

### Step 5:

|  |  |  |
| --- | --- | --- |
| Target node | Path | Total distance |
| J | X→J | 2 |
| L | X→L | 1 |
| M | X→L→M | 3 |
| N | X→L→M→N | 12 |
| P | X→L→M→P | 3 |
| Y | X→L→M→Y/ X→L→M→P→Y | 5 |

There is only 1 path left, X→L→M→N, choose N as the starting node, and there is only 1 ending node, **Y**. The distance of X→L→M→N→Y is 18, which is longer than X→L→M→Y or X→L→M→Y, so the shortest path from **X** to **Y** is **X→L→M→Y** or **X→L→M→P→Y** and the distance is **5**.

## Question 2

Singly Linked List and Doubly Linked List are common linked list data structures, which store data in a way of connecting nodes in memory. They share many similarities in features and applications, but also have some important differences. The following will contrast single linked list and double linked list and illustrate their characteristics and uses through examples.

### Singly Linked List

A singly linked list consists of a series of nodes, each node contains a data element and a reference to the next node. Each node of a singly linked list can only access the next node and cannot directly access the previous node. This is the most notable feature of the singly linked list, but also one of its limitations. Since there is only one reference between nodes, a singly linked list has less memory footprint than a doubly linked list.

#### Advantages:

* Small memory footprint, suitable for large-scale data storage.
* Insertion and deletion operations are fast and only need to modify the references of adjacent nodes.
* Relatively simple to implement and maintain.

#### Disadvantages:

* It cannot be traversed directly in reverse, and needs to be traversed from the beginning, which is less efficient.
* The operation is limited, and it is not suitable for scenarios that require bidirectional traversal.

### Double Linked List

Doubly linked lists also consist of a series of nodes, each node containing a data element, a reference to the next node, and a reference to the previous node. The double-linked list overcomes the reverse traversal limitation of the single-linked list, making it more flexible and efficient in some scenarios.

#### Advantages:

* Supports bidirectional traversal, enabling efficient forward and reverse traversal.
* Insertion and deletion are still faster, but slightly more complex than singly linked lists.
* It is suitable for scenarios that require two-way traversal, such as implementing the browser's forward and backward functions.

#### Disadvantages:

* It takes up a lot of memory, and each node needs to store two references.
* The insertion and deletion operations need to modify the references of the two nodes before and after, which is a little more complicated.
* Compared with singly linked list, implementation and maintenance are slightly more complicated.

### Example

Consider a playlist for a music player, using a doubly linked list to store songs. Each node contains song information, a reference to the previous song, and a reference to the next song. Users can freely switch songs in the list, and both forward and reverse traversal operations are handy.

To sum up, there are differences between single-linked list and double-linked list in terms of memory usage, insertion and deletion operations, and traversal methods. The choice of which linked list to use depends on the specific needs. If you need efficient two-way traversal or need to implement more complex data structures in some scenarios, double-linked list may be more suitable; if memory footprint and simplicity are more important to you, single-linked list may be more suitable.