Tutorial 3 – Linked List

- Linked list

Problem 1

Answer the following questions:

a) What is the complexity of searching for a node?
b) In what case would we prefer linked list over array (static or dynamic)?
c) In what case would we want to use array instead of linked list?

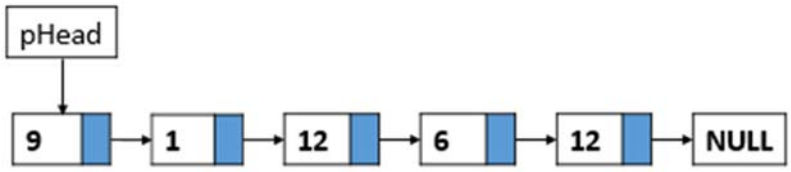
Problem 2

A circular linked list is a linked list where the last node (tail) has a next pointer pointing to the first node (head).

a) Propose a way to traverse a circular linked list.
b) Implement a function that prints every nodes of a circular linked list.

Problem 3

Implement a function to count the number of re-occurring data in a linked list, knowing that the data is only in the range [1, 20]. For example, given the following linked list:



Your function should create and return a linked list where:

- The first (1) node has a value of 1, because 1 appeared once in the above list.
- The 6th node has a value of 1.
- The 9th node has a value of 1.
- The 12th node has a value of 2.

Problem 4

Write a function to concatenate two linked lists:

```
PrintLinkedList(head1); // 1 (head) -> 7 -> 14 -> 0 -> NULL
PrintLinkedList(head2); // 9 (head) -> 4 -> 18 -> 18 -> 2 -> 4 -> NULL
Concat(head1, head2);
PrintLinkedList(head1); // 1 (head) -> 7 -> 14 -> 0 -> 9 -> 4 -> 18 -> 18 -> 2 -> 4 ->
NULL
```

Problem 5

Create:

- A linked list with 10 nodes.
- An array of type node with 10 elements.

Write the codes to:

- Print the address of each node in the linked list.
- Print the address of each element in the array.

Analyze the difference.