Solution@16-08-2023

160. Intersection of Two Linked Lists

```
package leetcode.easy;

import util.ListNode;

import java.util.HashSet;
import java.util.Set;

class IntersectionOfTwoLinkedLists {

  ListNode getIntersectionNodeSet(ListNode headA, ListNode headB) {

    if (headA == null) return headA;
    if (headB == null) return headB;

    Set<ListNode> nodeAddress = new HashSet<>();

    while (headA != null) {

      nodeAddress.add(headA);
      headA = headA.next;
    }

    ListNode result = null;

    while (headB != null) {
      if (nodeAddress.contains(headB))
        return headB;
```

```
    headB = headB.next;
  }

  return result;
}

ListNode getIntersectionNode(ListNode headA, ListNode headB) {

  int lenA = getListLength(headA);
  int lenB = getListLength(headB);

  while (lenA > lenB) {
    lenA--;
    headA = headA.next;
  }

  while (lenB > lenA) {
    lenB--;
    headB = headB.next;
  }

  // Now both heads are at same distance from intersection
  // Start moving them both until they meet
  while(headA != headB) {
    headA = headA.next;
    headB = headB.next;
  }

  return headA;
}
```

```
 private int getListLength(ListNode head) {
  int len = 0;

  while (head != null) {
   len++;
   head = head.next;
  }

  return len;
 }

}
```

172. factorial trailing zeroes

```
public class Solution {
    public int trailingZeroes(int n) {
        int result = 0;
        while (n != 0) {
            n = n / 5;
            result += n;
        }
        return result;
    }
}
```