**WIA1002 Data Structure**
**Tutorial 5: Stack**

1. Create the ListNode.java and Stack.java file (only the methods required based on the lecture notes). Then, perform the following:
    a. Insert three characters A, N, and D into the stack
    b. Remove one character from the stack
    c. Print the element on top of the stack
    d. Print the number of elements in the stack

    Example output:

```
Stack implementation using Linked List
Insert three nodes into the stack
 <-- D <-- N <-- A
Pop one node from the stack
 <-- N <-- A
The element on the top of the stack is N
 <-- N <-- A
The number of elements in the stack is 2
```

2. Create an ADT stack named **ArrayStack**. The ADT consists of the following method and the maximum size of the ADT is 10.
    a. Constructor
    b. isEmpty
    c. isFull
    d. getSize
    e. Push
    f. Pop
    g. Peek
    h. showStack

    Example output:

```
Stack implementation using Array
Insert three nodes into the stack
 <-- D <-- N <-- A
Pop one node from the stack
 <-- N <-- A
The element on the top of the stack is N
 <-- N <-- A
The number of elements in the stack is 2
```

3. Write a program to reverse a sentence and reverse the words in a sentence. You can use the stack implementation with linked list.

Example output:

```
Enter a sentence : Software-defined networking (SDN) technology is an approach to network management
Original Sentence : Software-defined networking (SDN) technology is an approach to network management
The sentence in reverse oder : tnemeganam krowten ot hcaorppa na si ygolonhcet )NDS( gnikrowten denifed-erawtfoS
The word in reverse order :management network to approach an is technology (SDN) networking Software-defined
```

4. Fibonacci is series of numbers in which each number is the sum of the two preceding numbers. Example 1, 1, 2, 3, 5, 8, etc. Write a method to solve the Fibonacci, Fib(N) given Fib(1)=1 and Fib(2)=1. You can use the stack implementation with linked list.

   Example output:

```
Fib(1): 1
Fib(5): 5
Fib(8): 21
Fib(14): 377
```

5. A common problem for compilers is to determine if the parentheses (in a string are balanced and properly nested. A string is balanced if contains properly nested pairs of parentheses. Example, ((()))() is balanced while )()( is not balanced.

   Example output:

```
(()) is balanced
)()( is not balanced
((())())() is balanced
((()))())) is not balanced
()) is not balanced
```

6. Create a program that generates 10 random number within 1 – 100 and store in an array. **Iterate** the array and insert the number into the stack according to this rule; the first number must be an odd number. Then, push the even number and odd number alternate into the stack. Iterate the array until the rule cannot be satisfied. Example output:

```
The random numbers are : 22 83 85 100 63 23 35 9 52 16
The numbers in the stack :  <-- 35 <-- 22 <-- 23 <-- 16 <-- 85 <-- 52 <-- 63 <-- 100 <-- 83
The numbers remain in the array : 9
```