# Josephus Problem

In a children's game, boys and girls sit in a circle and number off. Every Nth person must leave the circle. Whoever's left at the end is the winner. For a given *N* and *count*, can you predict the position of the winner?

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Michael | Hannah | Jacob | Ruth | Matthew |
|  |  |  |  |  |
| Matthew | Michael | Hannah | Jacob |  |
|  |  |  |  |  |
| Matthew | Michael | Hannah |  |  |
|  |  |  |  |  |
| Michael | Hannah |  |  |  |
|  |  |  |  |  |
| Michael |  |  |  |  |

For *N* = 5 and *count* = 4:

Starting position 1 is the winner.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Michael | Hannah | Jacob | Ruth | Matthew |
|  |  |  |  |  |
| Jacob | Ruth | Matthew | Michael |  |
|  |  |  |  |  |
| Matthew | Michael | Jacob |  |  |
|  |  |  |  |  |
| Jacob | Matthew |  |  |  |
|  |  |  |  |  |
| Jacob |  |  |  |  |

For *N* = 5 and *count* = 2:

Starting position 3 is the winner.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Michael | Hannah | Jacob | Ruth | Matthew |
|  |  |  |  |  |
| Hannah | Jacob | Ruth | Matthew |  |
|  |  |  |  |  |
| Jacob | Ruth | Matthew |  |  |
|  |  |  |  |  |
| Ruth | Matthew |  |  |  |
|  |  |  |  |  |
| Matthew |  |  |  |  |

For *N* = 5 and *count* = 1:

Starting position 5 is the winner.

# 

# The trick is to figure out where to stand so as to be the last person left standing.

# Historical note

# This problem gets its name from Flavius Josephus. In the 1st century AD, Josephus was a leader of a Jewish rebellion against Rome, was trapped in a cave, and somehow, amazingly, was the one who survived the circle of suicide. He then switched sides, advised the Roman armies and emperor, and ended up writing several books about it all.

# The Circular Singly-linked List

head

The Josephus problem can be modeled by a circular linked list in which all nodes are linked in a continuous circle, without using null. If we want to make a circular linked list storing B-C-D-E-F, most people will make "head" point to "B", as shown in the arrangement at the top. In that case,

1. What is the Big-O to print the list?\_O(n)\_

2. What is the Big-O to insert an "A" at the beginning, before the "B"? \_O(n)\_\_\_\_

3. What is the Big-O to insert a "G" at the end, after the "F"? O(n)\_\_\_

head

Now consider the arrangement of the nodes as shown at the bottom:

4. What is the Big-O to print the list?\_O(n)\_\_\_

5. What is the Big-O to insert an "A" at the beginning, before the "B"? O(1)

head.setNext(new ListNode(“A”, head.getNext)); \_\_\_\_\_

6. What is the Big-O to insert a "G" at the end, after the "F"? O(1)\_\_\_\_\_

* The list is B-C-D-E-F, but it is stored and processed as F-B-C-D-E for Big-O reasons.

Because of where it begins to count the first person

# Josephus Lab Assignment

We will divide this program into two days. The first day, write the print and insert methods and test them. Remember, the list is stored as F-B-C-D-E but is printed as B-C-D-E-F.

The insert method should insert one (1) node, taking care to keep the list circular. The insert method has two cases. Draw them here:

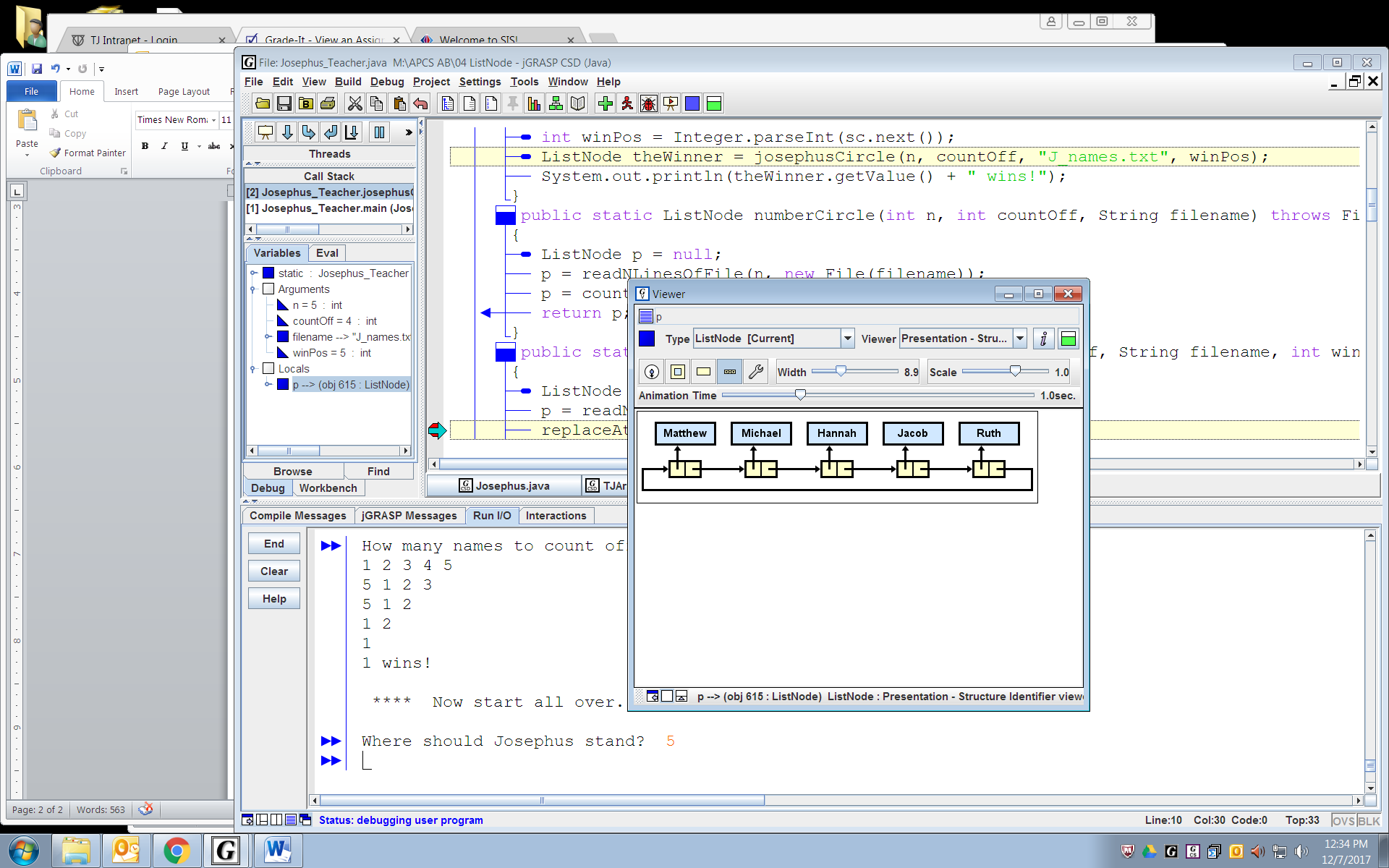
*Case 1:*

*P -> Null*

*Case 2:*

*Loops back and prints to itself*

On the second day, implement the Josephus game as a circular linked list. The driver plays the game twice, once with integers, which finds the winning position, and again with names. After reading the names Michael, Hannah, Jacob, Ruth, and Matthew, your data structure should look like:



This time, *replace* the name at the winning position with "Josephus", so that Josephus will always win.

***Sample run:*** *For full credit format the output exactly as shown below.*

B C D E F  
How many names (2-20)? 5  
How many names to count off each time? 2  
1 2 3 4 5  
3 4 5 1  
5 1 3  
3 5  
3  
3 wins!  
  
 \*\*\*\* Now start all over. \*\*\*\*   
  
Where should Josephus stand? 3  
Michael Hannah Josephus Ruth Matthew  
Josephus Ruth Matthew Michael  
Matthew Michael Josephus  
Josephus Matthew  
Josephus  
Josephus wins!