Java 3 Activity 3 Task 2

30003389 – Kyer Potts

Table of Contents

[Table of Contents i](#_Toc25523484)

[What is the difference between a Queue and a Stack? 1](#_Toc25523485)

[Is it possible to add nodes to the beginning of a LinkedList? If so, how? What about adding a node to the end of a LinkedList? If this can be done, what method would be used? 1](#_Toc25523486)

[What is the purpose of implementing the Comparable interface? 1](#_Toc25523487)

[Course Codes HashMap Collection 2](#_Toc25523488)

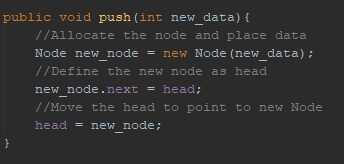
# What is the difference between a Queue and a Stack?

Queues are first in first out (FIFO) data structures, this means that the first element inserted into the Queue will also be the first element to come out of the Queue. This makes Queues valuable for sequential tasks or problems.

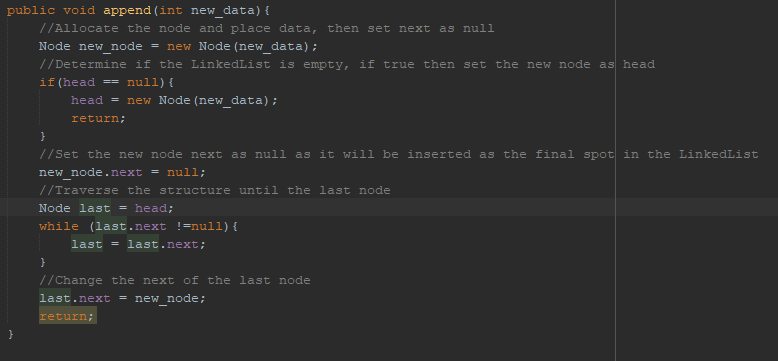
Stacks and last in first out (LIFO) data structures. The last element to be inserted into the Stack will also be the first element to come out of the Stack. This functionality makes Stacks practical for recursive algorithms.

# Is it possible to add nodes to the beginning of a LinkedList? If so, how? What about adding a node to the end of a LinkedList? If this can be done, what method would be used?

A node can be added to the beginning of a LinkedList with the following code snippet:



A node can be added to end of a LinkedList with the following code snippet:



# What is the purpose of implementing the Comparable interface?

Implementing the Comparable interface to a class allow the class to use many inbuilt comparison methods. This allows the class to use sorting features much more effectively and saves the programmer from writing superfluous code. Methods such as the compareTo() method allows to programmer to set a specific return value that can be tailored to fit the requirements of the algorithm. This increases code reuse, allows for more efficient sorting and helps the programmer to assign specificity in greater detail

# Course Codes HashMap Collection

