**NAMA : SHABRINA QOTTRUNNADA**

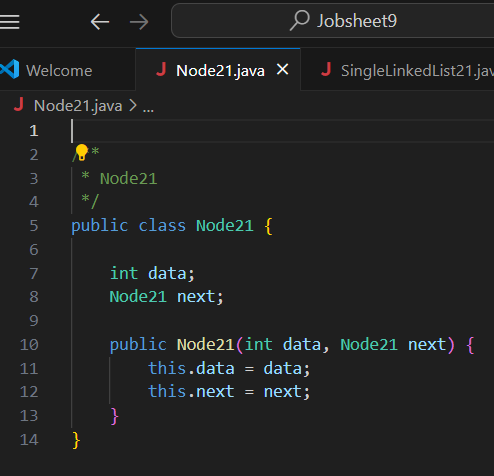
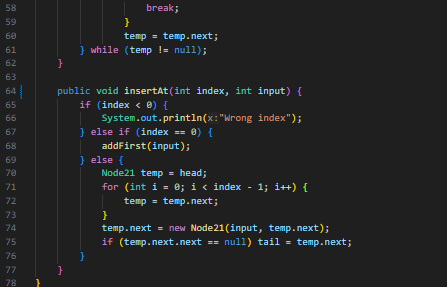
**KELAS : 1G**

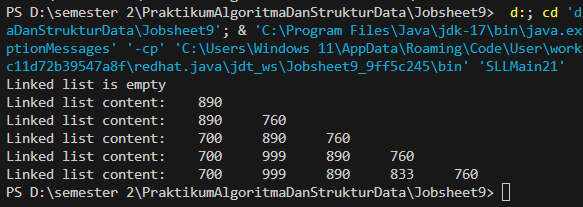
**NIM : 2341760160**

**MATKUL : ALGORITMA DAN STRUKTUR DATA (LINKED LIST)**

**JOBSHEET 9**

**Github :** [**https://github.com/shabrinaq/smstr2\_Jobsheet9/tree/main**](https://github.com/shabrinaq/smstr2_Jobsheet9/tree/main)





# Questions

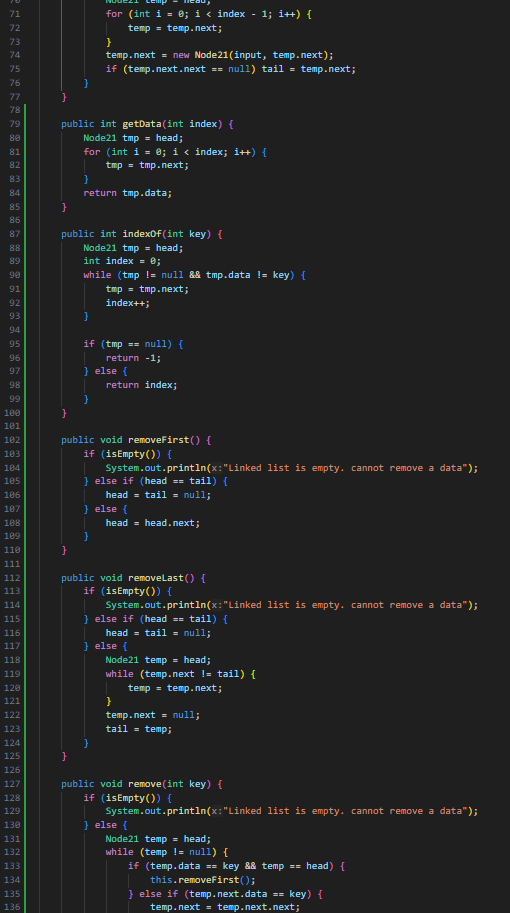
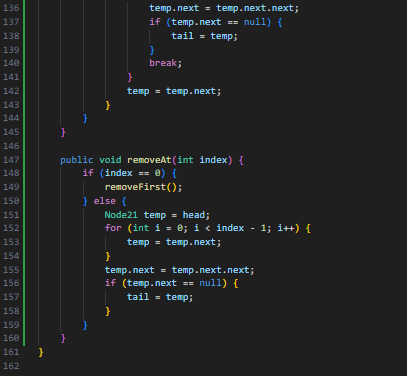
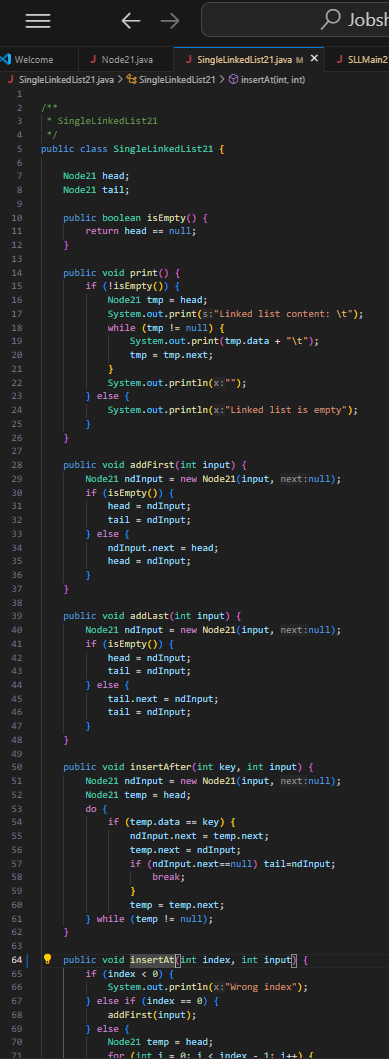
* + - 1. Why the output of the program in first line is “Linked list is empty”?
* Linked list is empty pada baris pertama itu menunjukkan bahwa linked list akan dimulai dan masih kosong. Hal ini menunjukkan bahwa program tersebut dalam kondisi awalan dan elemen akan ditambahkan dan menghasilkan output yang diinginkan.
  + - 1. Please explain the usage of these following codes in:

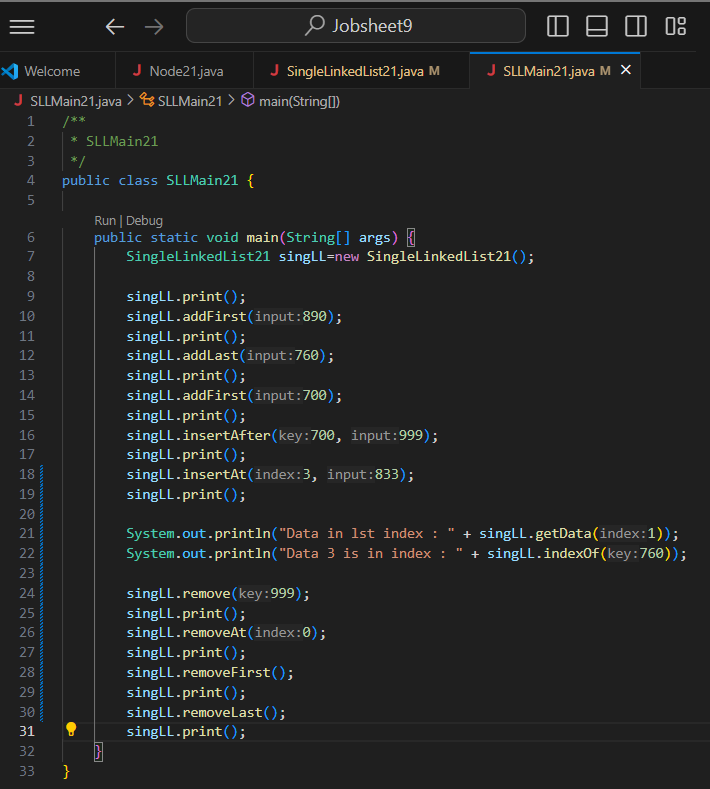
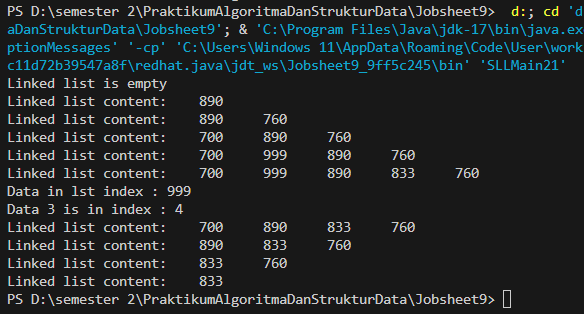


* Baris pertama itu digunakan untuk menunjukkan bahwa node ‘ndInput’ akan menunjuk ke ‘temp.next’.
* Baris kedua itu digunakan untuk menetapkan node ‘temp’ akan menunjukkan ke node baru yaitu ‘ndInput’.
  + - 1. In **SingleLinkedList,** what is the usage of this following code in **insertAt**?



* Kegunaannya di insertAt adalah untuk menyelesaikan masalah seperti node baru yang dimasukkan tepat sebelum node terakhir pada file SingleLinkedList21.





# Questions

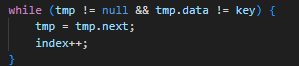
* + - 1. Why we use **break** keyword in remove function? Please explain
* ‘break’ digunakan pada remove itu karena untuk mengeluakan loop setelah menemukan dan menghapus node dengan kunci yang sudah ditentukan itu.
  + - 1. Please explain why we implement these following codes in method remove



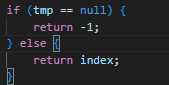
* Kode program diatas itu digunakan untuk mengecek dan mencari node yang akan kita hapus dengan kode program ‘temp.next’. lalu jika ‘temp.next.data == key’, maka kita bisa memperbarui next dari node ‘temp’ untuk melangkahinya ke node ‘temp.next’ yang akan kita hapus nanti.
  + - 1. What are the outputs of method indexOf? Please explain each of the output!



Digunakan untuk menginisialisasi temp ke head untuk mencari node pertama dan index = 0 untuk mengetahui posisi node sekarang ada di mana.



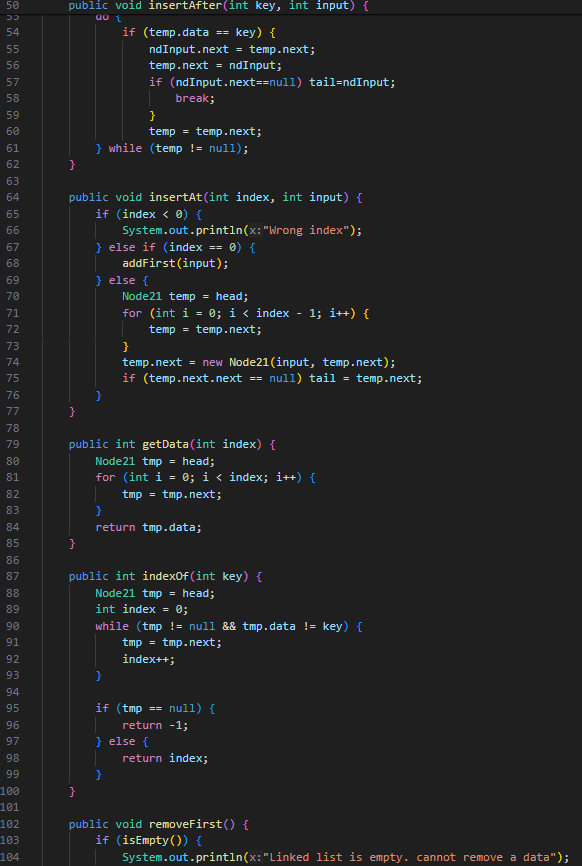
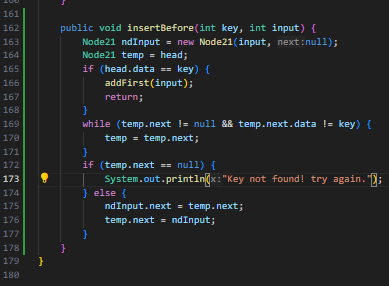
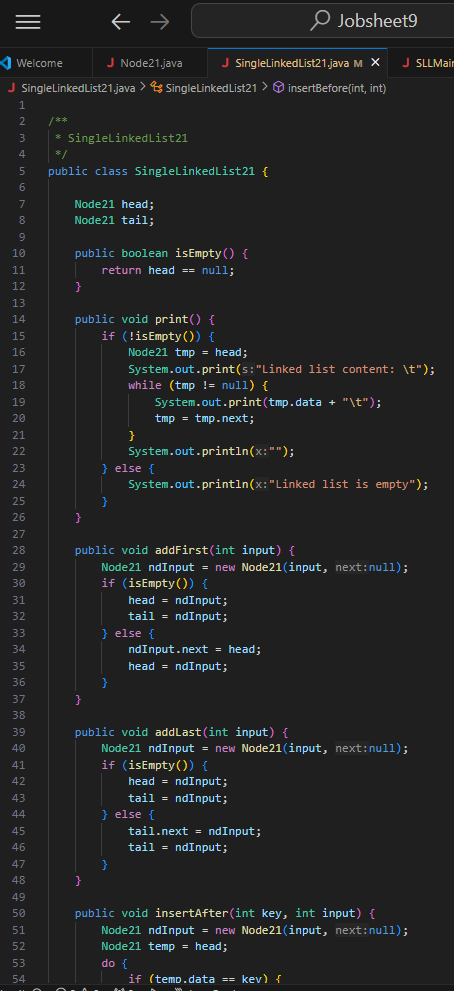
Digunakan untuk ‘tmp tidak sama dengan null’ serta data node sekarang itu ‘tmp.data tidak sama dengan key’. Lalu jika node tidak sama dengan key, maka akan berpindah ke node ‘tmp.next’.

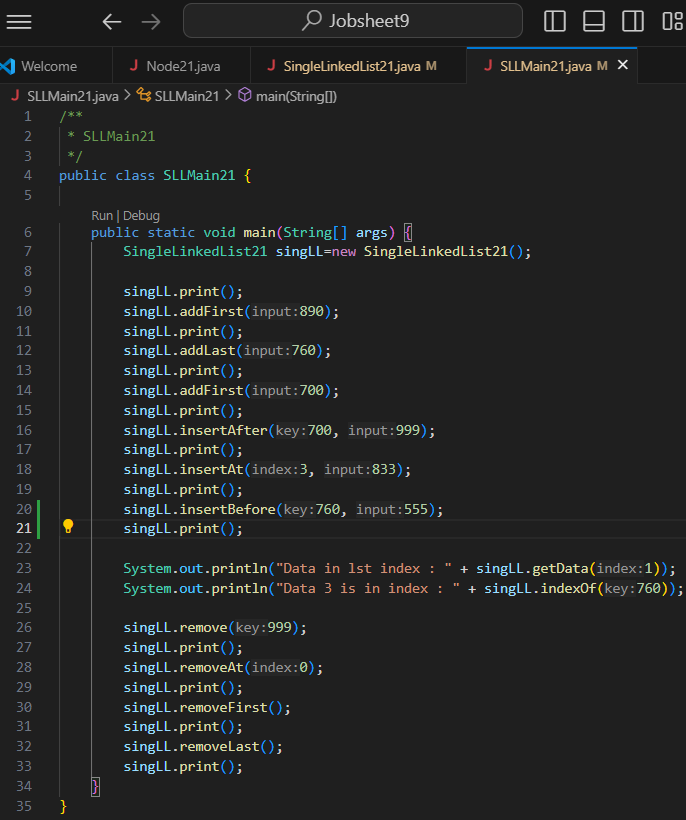
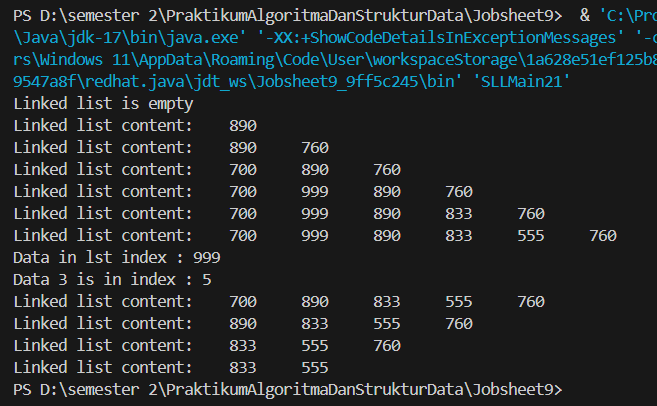


Digunakan untuk

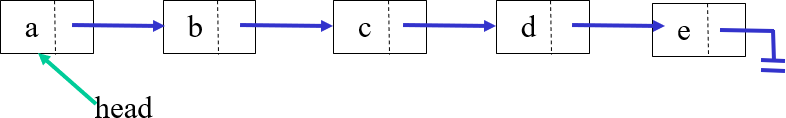
* Jika di baris if itu ‘temp sama dengan null’ maka ‘key’ tidak ditemukan oleh lindked list dan dikembalikan menjadi -1.
* Jika di baris else itu ‘tmp tidak null’ maka key itu nanti ditemuka dan pengembalikan nilai itu diposisi node dengan data yang valid.

# Assignments

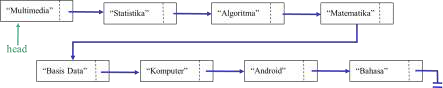
1. Create a method **insertBefore()** to add node before the desired keyword



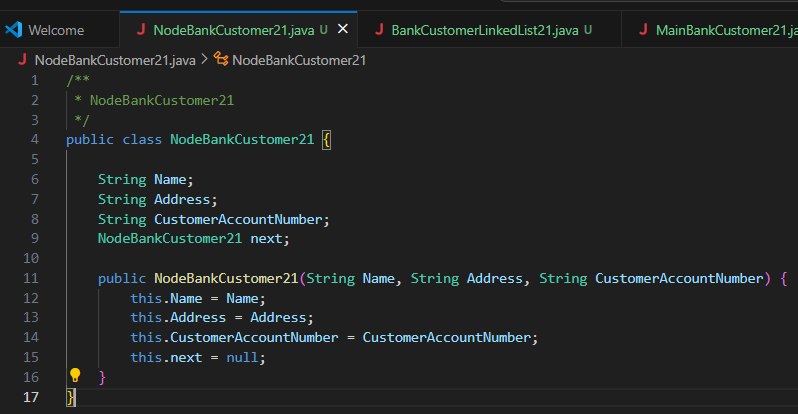
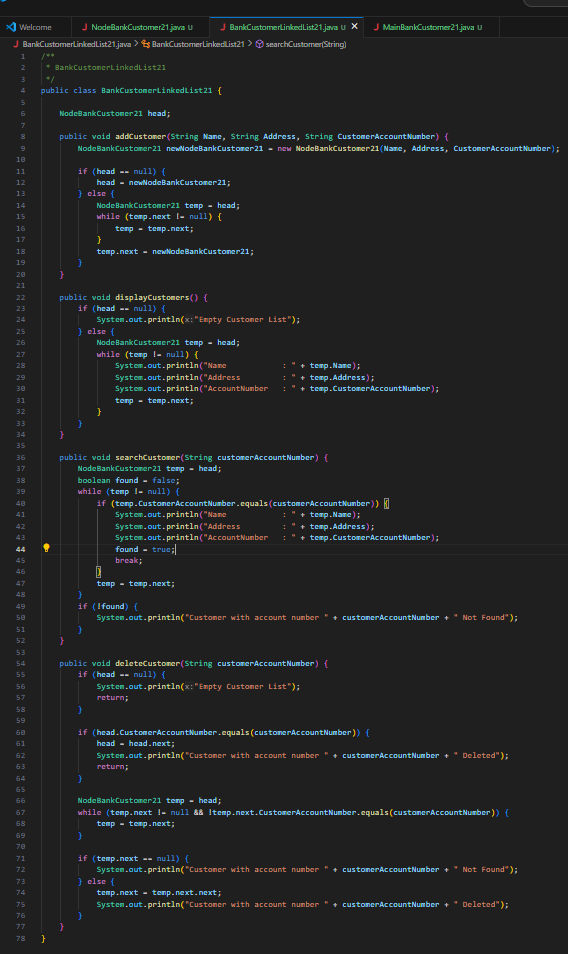
1. Implement the linked list from this following image. You may use 4 method of adding data we’ve learnt

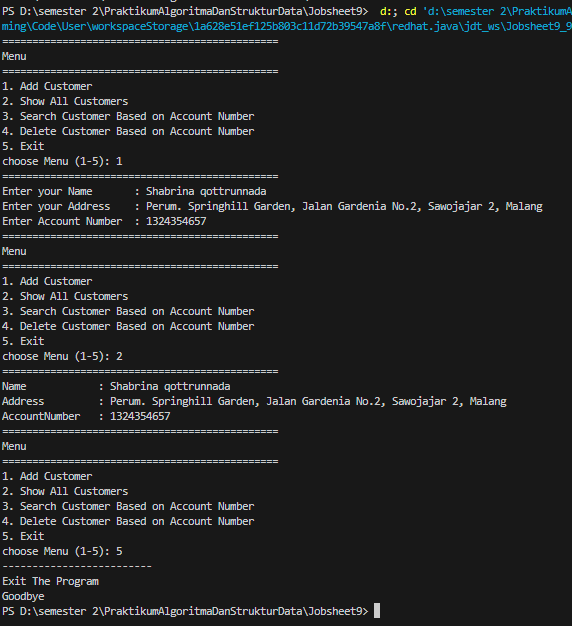


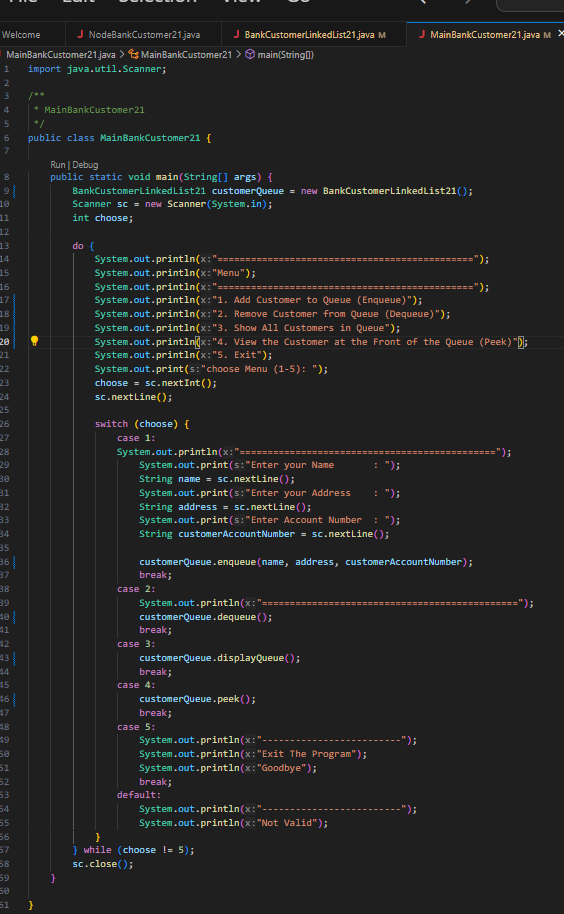
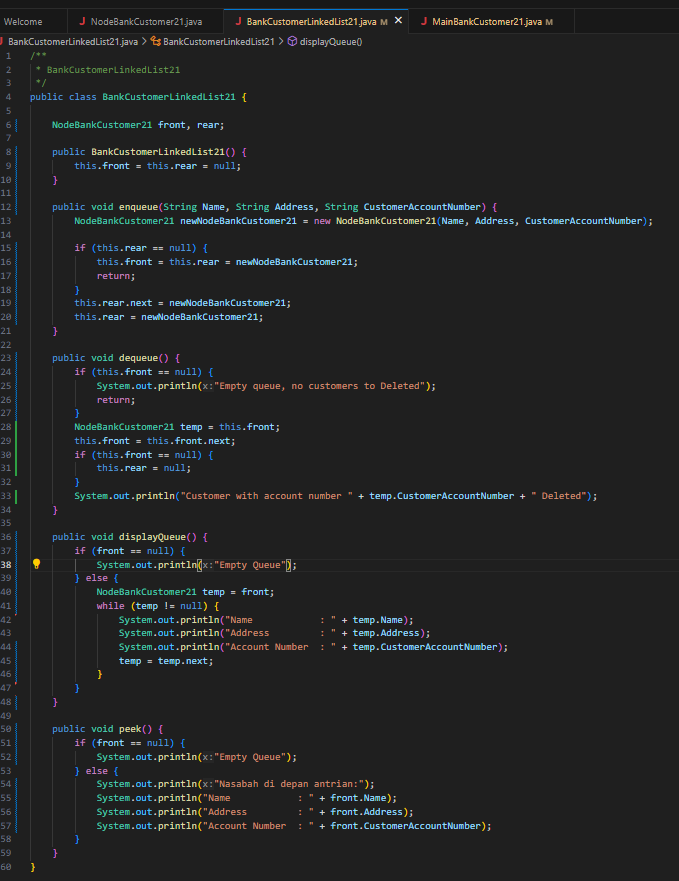
1. Create this following **Stack** implementation using Linked List implementation



1. Create a program that helps bank customer using linked list with data are as follows: Name,address, and customerAccountNumber





1. Implement **Queue** in previous number with **linked list** concept

