Oefeningen OO structuren:

De voorbereiding hiervoor hebben we gedaan tijdens de theorie sessie van dit onderwerp. Het aanmaken van de Node en List, Queue en Stack klassen. Heb je die nog niet dan kan je meer info terugvinden in de video opname hiervan.

Opmerking: ik heb in mijn oplossingen de klassen hernoemt naar NodeInt.cs, ListInt.cs, QueueInt.cs, StackInt.cs ,… omdat het allen datastructuren zijn die met type "int" werken.

(dit is voorlopig de beste oplossing, verder dit semester zien we nog dat er zeker een betere oplossing hiervoor bestaat)

A. Singly Linked list

1. Bouw de SLL. Hiervoor heb je de 2 klassen nodig **Node** en **List**.
    a. Een Node bevat de '**value'** en de referentie naar de volgende node.
    b. De List klasse is verantwoordelijk voor het **beheer van de lijst**. Dus ervoor zorgen dat er:
        i. Een verwijziging wordt bijgehouden naar de First en Last Node
        ii. Elke node ten allen tijde correct verwijst naar de node die hierop volgt in de lijst.
        iii. Dat het mogelijk is om elementen (nodes) toe te voegen in de lijst
        iv. Dat het mogelijk is om elementen te verwijderen uit de lijst
        v. Dat het mogelijk is om elementen in te voegen in de lijst, dit kan na een bepaalde node, of voor een bepaalde node zijn.
        vi. Dat het mogelijk is om een element op te zoeken in de lijst
        vii. Dat het mogelijk is om de lijst leeg te maken.
        viii. Dat het mogelijk is om te weten of de lijst al dan niet leeg is.
        ix. Dat het (eventueel) mogelijk is om de lengte te kennen van de lijst ? (hoe zou je dit kunnen realiseren ?)
2. Test de SLL uit vanuit een client applicatie waarmee je alle functies van de list kan testen (toevoegen, verwijderen, …).
3. Om de inhoud van de lijst weer te geven kan je
    a. met een while lus door de lijst lopen startende vanaf de first node.
    b. ofwel node.ToString()overschrijven en daarin ook ineens de "next" node zichzelf laten weergeven. Op die manier kan je de lijst heel eenvoudig tonen door enkel op de eerste node ToString() aan te roepen….
4. Bedenking 1: Iemand die een referentie naar een Node doorkrijgt, mag die zomaar de Next node kunnen aanpassen ? Zo neen, waarom niet en hoe zouden we dat eventueel kunnen verhinderen ?
5. Bedenking 2: Mag iemand de First en Last node van de List klasse kunnen instellen en aanpassen ? (had ik reeds aangehaald in de theorie)
6. Bedenking 3: Stel, ik heb 2 SLL lijsten, ik zoek een node op in lijst A en vraag om deze te verwijderen uit lijst 2. Wat zou er dan gebeuren ? En wat zou er moeten gebeuren ?

Kijk na of de time complexity van jouw implementatie overeenstemt met de overzichtstabel in de cursus.

B. **Doubly Linked List**:
    a. Voor de DLL heb je een nieuwe Node klasse nodig. Deze heeft immers bijkomend ook nog een referentie naar de vorige node in de lijst. Kan je hier eventueel denken aan overerving ? Wat zou de beste base node kunnen zijn ?
    b. De functies van de DLL klasse zelf zijn identiek aan de SLL klasse. Uiteraard is de implementatie niet helemaal hetzelfde aangezien er nu 2 referenties up-to-date gehouden moeten worden.
        i. Is dat laatste een verantwoordelijkheid van de List klasse zelf ?
        ii. Is dat eerder een verantwoordelijkheid van Node klasse ?
    c. Kijk ook na of de 3 bedenkingen bij de SLL ook hier geldig zijn.
    d. Test de DLL (je kan dezelfde client test code gebruiken als bij de SLL).

C. Queue op basis van een Linked List
    a. Bouw een Queue die getallen (int) kan bevatten en werkt met behulp van een Linked List. Je kan zelf bepalen of je een SLL of DLL gebruikt hiervoor.
    b. Test deze nieuwe queue. Je kan hiervoor ook de client app gebruiken die je reeds had gemaakt voor de Queue op basis van een array of zelfs in het autofabriekje.

D. Stack op basis van een Linked List
    a. Bouw een Stack die getallen (int) kan bevatten en werk met behulp van een SLL. Let op het juiste gebruik van de SLL zodat je de meest performante oplossing bekomt !
    b. Test met 1 van de Stack Test clients die je reeds had gemaakt.

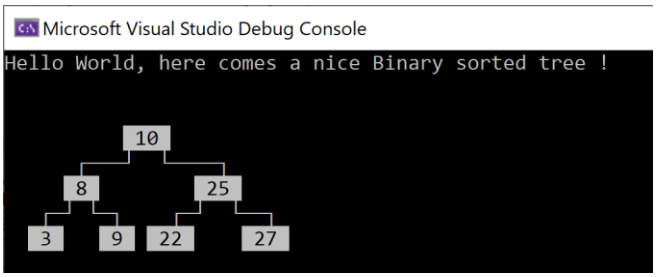
E. Bubble sort op basis van een Linked List
    a. Maak een nieuwe variant van de Bubble Sort waaraan je een SLL meegeeft.
        i. Hoe gaan we om met de swap operatie ? Wisselen we dan de node of enkel de value ?

F. Binary Tree

a. De Sorted Binary Tree (of Binary Search Tree, kortweg BST) is een binaire boomstructuur waarbij er **extra** regels gelden tov. een gewone tree. Dus:

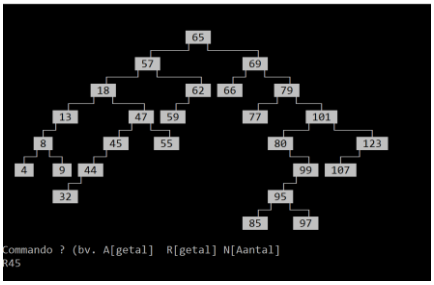
i. Elke node heeft **max. 2 children** (genaamd : Left en Right)
ii. De value van de **left** child moet **kleiner** zijn dan die van zijn **parent** Node
iii. De value van de **right** child moet **groter** zijn dan die van zijn **parent** Node.

b. Met deze info kan je reeds een nieuwe Node klasse maken en alle properties voorzien.
c. Daarnaast Hebben we ook nog een 2e klasse nodig (BST.Tree) die de volledige tree beheert. Deze zorgt ervoor dat:
   i. Er een referentie wordt bijgehouden naar de **Root** node van de tree
   ii. Er een value kan worden **toegevoegd** aan de tree (dus aanmaken van een nieuwe node, value instellen en de node op de juiste plaats in de tree onderbrengen rekening houdende met de extra regels voor een BST.
   iii. Er een value kan worden **verwijderd** uit de tree. Dwz. dat de node zal moeten worden opgezocht, en verwijderd worden. Hier geldt echter ook extra aandacht want onze BST moet steeds blijven voldoen aan de speciale regels van hierboven die bepalen welke node de verwijderde node zal vervangen. Er zijn 3 mogelijke scenario's, zie hiervoor de cursus.
   iv. De tree kan worden leeggemaakt
   v. De hoogte van de tree kan worden opgevraagd.
d. Je kan deze tree vervolgens testen en weergeven vanuit een console applicatie.



Ik heb een stukje code gevonden op stackoverflow dat een binary tree vrij netjes kan weergeven in console (zie screenshot hierboven). Je kan deze klasse gewoon overnemen vanuit mijn repo. Je vindt de klasse (genaamd **BinaryTreePrinter**) terug in mijn console project '**TreeTester'**. In dat project (main) zit ook demo code en kan je ook zien hoe je die klasse moet gebruiken om jouw tree te 'Printen' in de console.

Je kan vanuit je console applicatie de verschillende functies van jouw Tree testen (dus toevoegen, verwijderen, leegmaken van de tree) en je kan deze misschien ook eens laten opvullen met een willekeurig aantal waarden (die je kan laten genereren met onze RandomGenerator)



Succes !