Analyse:

Alle metodene i alle klassene er merket med sin kjøretidskompleksitet under metode kommentarene i eclipse filene.

MyNode klassen sine metoder har kjøretidskompleksitet O(1), konstant tid. Denne klassen består hovedsakelig av set og get metoder. Set metodene setLeft, setParent, setRight typisk bare tilordner en variable til en annen. Get metodenen bare returnerer variabelen eller element.

MyBinaryTree klassen sine metoder har mye av det samme som MyNode, get og set som er merket med kjøretidskompleksitet under metode kommentaren i eclipse der også, altså 0(1), konstant tid. Men metoden containsElement kjører i O(N) tid fordi i verste fall så finnens ikke element i treet. Det samme gjelder for containsNode som også i verste fall må søke igjennom hele liste for å se om node finnes i listen , altså O(N). toString metoden kjører i O(N) fordi den skal printe ut alle elementene i listen.

MyStack klassene sine metoder kjører alle i konstant tid utenom toString metoden som må printe ut hvert element i stack`en , den kjører i linjær tid O(N). De er get og set her også som kjører i som sagt konstant tid. Pop og push kjører i O(1) tid for pop tar siste elementet ut og fjerner og push tar et nytt element og legger på toppen av stack`en, altså en konstant operasjon.

Converter klassen sine metoder er getPrecedOrder som kjører i konstant tid,O(1), den sjekker to input og leverer tilbake den med høyest presidens ut fra en boolsk betingelse. isThisOperator metoden kjører også i O(1) tid for den levere tilbake true eller false, etter input sjekk.

convertExp metoden bruker flere andre metoder, metodene er stacken sin isEmpty som er O(1), Converter klassene sine getPrecedorder, O(1) og isThisOperator som kjører i konstant tid, men selve convertExp metoden tar en streng som input og kjører på lengden av denne strengen så hele metoden tar linjær tid når vi ser vekk fra lavere orden kjøretid.

O(1) + O(1) + O(1) + O(N) = O(N).