1.7

1. Het ophalen van een item in een Python list door middel van zijn index: O(1). De tijd dat het mijn laptop kost om list[1] en list[999] op te halen blijft gelijk. Dat de grootte van de lijst (of index) toeneemt heeft geen invloed op hoe lang het duurt om een element op te halen via een index.

2. Het op de juiste plek invoegen van een waarde in een gesorteerde Python lijst: O(n). Stel we hebben een lijst [1, 2, 4, 3, 5, 6]. We willen het getal 7 toevoegen. De functie moet opzoek naar het punt waarin het element 7 groter is dan het vorige element en kleiner is dan het volgende element. Dus dan krijg je 7>1, 7>2, 7>3, 7>4, 7>5, 7>6. 7 wordt toegevoegd achteraan de lijst, omdat 7 op dit punt groter is dan het vorige element, en er is geen volgend element. Dit zijn dus 6 vergelijkingen, even groot als de lijst zelf. Als je een lijst van 1 tot 100 hebt en daarin het getal 101 aan wilt toevoegen, dan krijg je 100 vergelijkingen. Dus O(n).

3. Het Shell sorteren van een array: O(n2). Shell sort heeft een eerste for loop met een complexiteit van O(n). Vervolgens zit hier weer een for loop in, dit zorgt voor een complexiteit van O(n2). De derde for loop die hier in zit zorgt voor een complexiteit van O(n3) echter het verschil tussen de twee complexiteiten is zo klein dat men kan zeggen dat het een complexiteit van O(n2) heeft.

4. Het Bubble sorteren van een array: O(n2). Bubble sort heeft een oneindige while loop. Hierin wordt iedere keer alle n-elementen met elkaar vergelijken en omgewisseld indien nodig. Dit houd in dat de gehele lijst meerdere keren wordt doorlopen, totdat deze volledig gesorteerd is.

5. Het oplossen van de Torens van Hanoi voor n schijven: O(2n-1). n = 1, 2, 3, 4, 5 en 6 is het aantal operaties, respectievelijk, 1, 3, 7, 15, 31 en 63. Het aantal operaties neemt bij 2 met 2 operaties toe. Bij 3 met 4, bij 4 met 8, bij 5 met 16 etc. 2^2 = 4, 3^2 = 8, etc. Dus een complexiteit van O(2n-1).

6. Het vinden van alle permutaties: O(k\*n!). Als je 3 **verschillende** getallen hebt (3, 2, 1) dan heb je 1x2x3 mogelijke combinaties. Als je 4 verschillende getallen hebt (4, 3, 2, 1), dan krijg je: 1x2x3x4 = 24 mogelijke combinaties. Bij 5 verschillende getallen (1, 2, 3, 4, 5) is dat al 120. Dus is dit een complexiteit van O(k\*n!).