**Opgave 1**

Bij het zoeken of de som bestaat hebben we te maken met meerder loops. Als eerste A[i], deze begint te tellen bij positie twee, A[2], van de array zodat we de eerste twee posities A[0] + A[1] kunnen optellen.

Als er geen som is moet als eerste de positie A[i] met één worden verhoogd, in mijn methode is dit *sumPos*. Nu kunnen moeten we weer vanaf 0 beginnen met het zoeken naar de som. In mijn methode is *start* het begin van de eerste som loop. Vervolgens zo is er een twee teller *loopCounter* die alle posities vanaf *start + 1* tot *sumPos -1* doorloopt om zo te bepalen of de som bestaat. Als dit geen succes oplevert kan de start positie met één worden verhoogd en kan de *loopCounter* opnieuw zijn werk doen. Bij het aanroepen moet: sumPos op twee, start op nul en loopCounter op éé’n staan

public static boolean sumFind(List<Integer> lijst, int sumPos, int start, int loopCounter) {

if (lijst.get(sumPos) == (lijst.get(start) + lijst.get(loopCounter))) {

return true;

}

if (loopCounter + 1 < sumPos) {

loopCounter++;

return sumFind(lijst, sumPos, start, loopCounter);

} else if (start + 1 < sumPos) {

start++;

loopCounter = start + 1;

return sumFind(lijst, sumPos, start, loopCounter);

} else {

if (sumPos + 1 < lijst.size()) {

sumPos++;

return sumFind(lijst, sumPos, 0, 1);

}

}

return false;

}