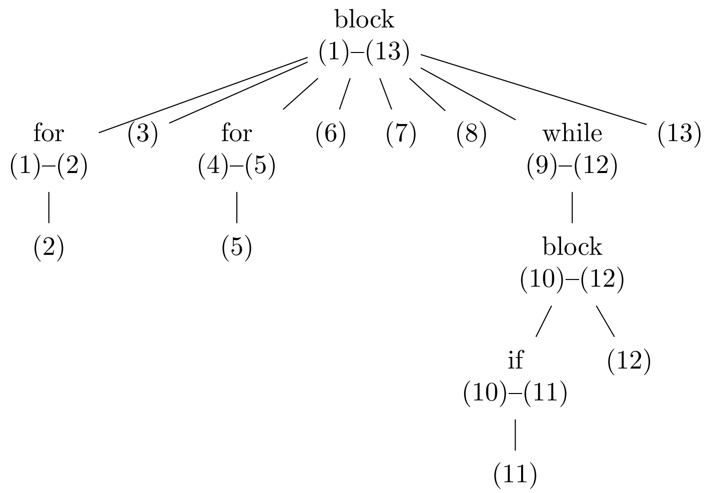
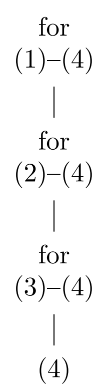
1. The running time of the body of the for-loop on lines (1) and (2) is 0. The loop goes around at most 100 times. Hence the for-loop takes $O(100)$ time, which is $O(1)$. Line (3) takes $O(1)$ time. Lines (4) and (5) take $O(n)$ time. Lines (6) to (8) take $O(1)$ time each. The while-loop on lines (9) to (12) goes around $n-i$ times. But we know from line (8) that $i=1$. Thus it goes around $n-1$ times. The body of the loop is a block. Line (10) is an if-statement that takes $O(1)$ time with no else-part. The if-part takes $O(1)$ time. Line (12) also takes $O(1)$ time. Thus the while-loop takes $O(1+(3+1)(n-1))$ time which is $O(n)$. Line (13) takes $O(1)$ time. Using the summation rule, we determine that the program takes $O(n)$ time.



2.

a) Line (3) goes around $n-i$ times. Line (2) goes around $n-i-1$ times. Line (4) takes $O(1)$ time. Thus lines (2) to (4) take $O((n-i)(n-i-1))$ time. We can neglect the lower order terms and say it is $O((n-i)^2)$.

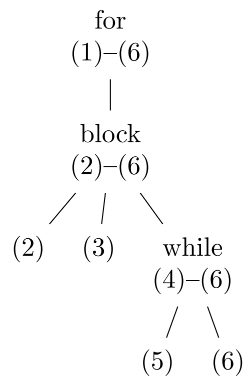
b) The loop starting on line (1) goes around $n-1$ times. Multiply this with $(n-i)^2$ to get $O(n^3)$ after dropping the lower order terms.



3.

a) The while-loop goes around $\log i$ times. The running time of the body of the while-loop is $O(1)$. Hence the while-loop takes $O(\log i)$ time.

b) The for-loop runs $((n+1)-1)$ times and the body takes $O(\log i)$ time. But the upper bound of the for-loop has the while loop take $O(\log n)$ time. Therefore the entire program takes $O(n \log n)$ time.



4. Lines (1), (4), (5), and (6) are $O(1)$. Line (3) is $O(1)$ and both the if-part and else-part are $O(1)$ hence the entire if-statement is $O(1)$, which is the body of the loop. The loop goes around $(n+1) - i^2$ times, but we can also say it stops when $i^2 > n$, or $i > \sqrt{n}$. Hence the number of times the loop goes around has a limit of $\sqrt{n}$. Thus the function takes $O(\sqrt{n})$ time.