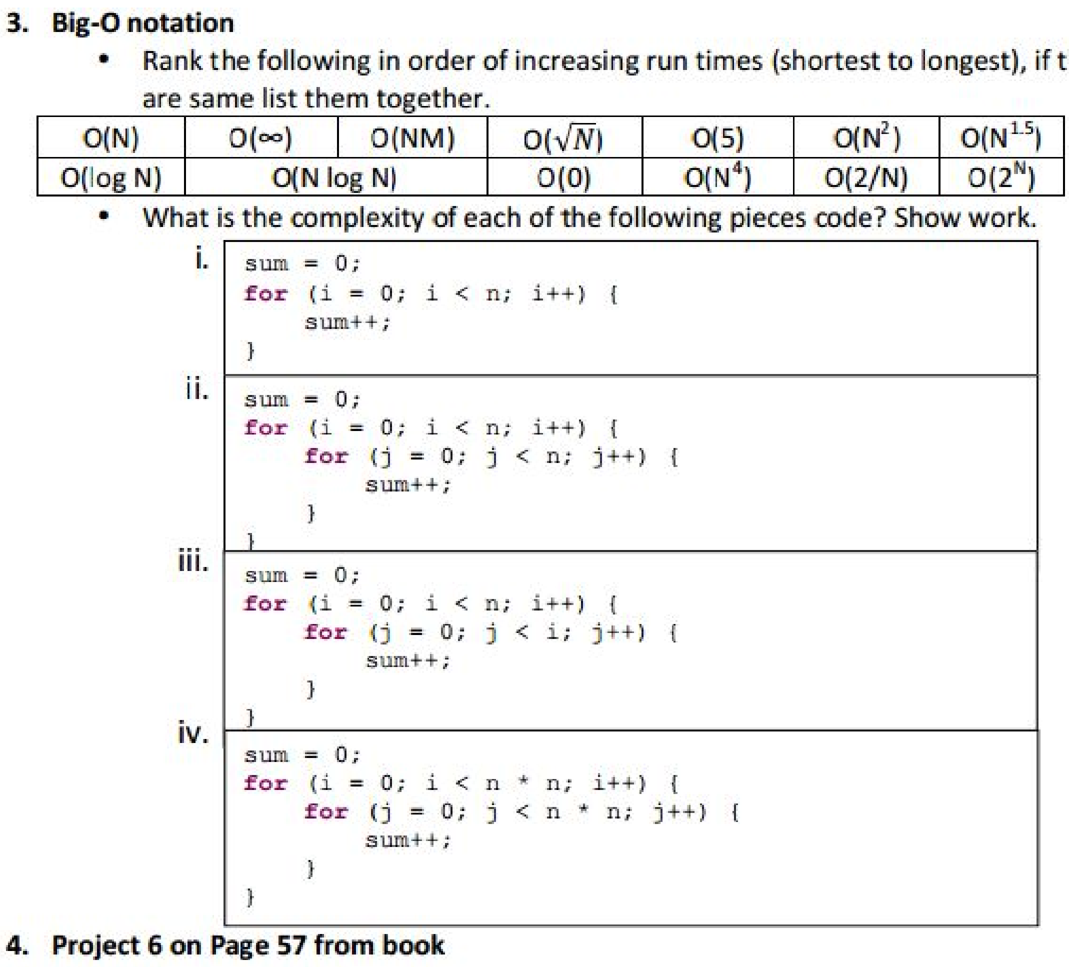

3. Big-O notation

- Rank the following in order of increasing run times (shortest to longest), if they are same list them together.

O(N)	O(∞)	O(NM)	O($\sqrt{N}$)	O(5)	O(N^2)	O($N^{1.5}$)
O(log N)	O(N log N)		O(0)	O(N^4)	O(2/N)	O(2^N)

- What is the complexity of each of the following pieces code? Show work.

i.

```
sum = 0;
for (i = 0; i < n; i++) {
    sum++;
}
```

ii.

```
sum = 0;
for (i = 0; i < n; i++) {
    for (j = 0; j < n; j++) {
        sum++;
    }
}
```

iii.

```
sum = 0;
for (i = 0; i < n; i++) {
    for (j = 0; j < i; j++) {
        sum++;
    }
}
```

iv.

```
sum = 0;
for (i = 0; i < n * n; i++) {
    for (j = 0; j < n * n; j++) {
        sum++;
    }
}
```

4. Project 6 on Page 57 from book

i. Most efficient, O(N) - the program will run data as efficient as the size of the data, the size and efficiency essentially go up at the same rate. One loop through data for each = n

ii. 3rd most efficient, O(N^2) - the program must run through the length of data twice, as a result the program must go through the loop n^2 times.

iii. 2nd most efficient O(NM) - the program must go through the loop through the entire data (n) for the first for loop and follows up to the current index for the second index (m). As a result the program goes through the loop nm times.

iv. Least efficient O(n^4)- the program must go through the data n^2 times for the first for loop and n^2 times for the second for loop resulting in a very inefficient loop traverse going through n^4 times.