ANALYZING NONRECURSIVE PROGRAMS: AN EXAMPLE

We analyze the (meaningless) program shown below.

```c
#include <stdio.h>

int bar(int x, int n);
int foo(int x, int n);

/* This line is left intentionally blank */
main()
{
  int a, n;
  scanf("%d", &n);
  a=foo(0, n);
  printf("%d ", bar(a, n));
}

/* This line is left intentionally blank */
int bar(int x, int n);
{
  int i;
  for i ← 1 to n do
    | x+ = i;
  end
  return x;
}

/* This line is left intentionally blank */
int foo(int x, int n);
{
  int i;
  for i ← 1 to n do
    | x+ = bar(i, n);
  end
  return x;
}
```

Note that this is a nonrecursive program. We first analyze the function bar, then the function foo, and finally the function main. It is clear that the function bar is $O(n)$. Now, if the function call in the body of a for-loop, we add its cost to the bound on the time for each iteration. It follows that the running time of a call to foo is $O(n^2)$. Next, when the function call is within a simple statement, we add its cost to the cost of that statement. Thus, the function main takes $O(n^2)$ times. Therefore, the running time of this program is $O(n^2)$.

Reference: http://infolab.stanford.edu/~ullman/focs/ch03.pdf