Checkpoint 0

A water main break in GHC has, confusingly, broken the C0 compiler’s -d option! C0 contracts are now being treated as comments, and the only way to generate assertion failures is with the assert() statements.

Insert assert() statements into the code below so that, when the code runs, all operations (C0 statements, conditional checks, and assertions) are performed at runtime in the exact same sequence that would have occurred if we compiled with -d. Not all of the blanks need to be filled in.

```c
int mult(int x, int y)
{ /* 1 */
  int k = x; int n = y;
  int res = 0;
  /* 2 */
  while (n != 0)
  { /* 3 */
    if ((k & 1) == 1) res = res + n;
    k = k >> 1;
    n = n << 1;
  } /* 4 */
  return res;
}

int main() {
  int a;
  /* 8 */
  a = mult(3,4);
  /* 9 */
  return a;
}
```
Checkpoint 1
Rank these big-O sets from left to right such that every big-O is a subset of everything to the right of it. (For instance, $O(n)$ goes farther to the left than $O(n!)$ because $O(n) \subset O(n!)$. If two sets are the same, put them on top of each other.

$O(n!)$  $O(n)$  $O(4)$  $O(n \log(n))$  $O(4n + 3)$  $O(n^2 + 2000n + 3)$  $O(1)$  $O(n^2)$  $O(2^n)$  $O(\log(n))$  $O(\log^2(n))$  $O(\log(\log(n)))$

Checkpoint 2
Using the formal definition of big-O, prove that $n^3 + 300n^2 \in O(n^3)$.

Checkpoint 3
Using the formal definition of big-O, prove that if $f(n) \in O(g(n))$, then $k \times f(n) \in O(g(n))$ for $k > 0$.

One interesting consequence of this is that $O(\log_i(n)) = O(\log_j(n))$ for all $i$ and $j$ (as long as they’re both greater than 1), because of the change of base formula:

$$\log_i(n) = \frac{\log_j(n)}{\log_j(i)}$$

But $\frac{1}{\log_j(i)}$ is just a constant! So, it doesn’t matter what base we use for logarithms in big-O notation.

Checkpoint 4
Simplify the following big-O bounds without changing the sets the represent:

$O(3n + 2)$, $O(n^{2.5} + \log_2(n))$, $O(\log_{10}(n) + \log_2(7n))$. 

2