Review Activity 8

More on Algorithmic Analysis

1) Explain how the **flip_bits** function is mapped to the recurrence relationship given below.

```
void flip_bits(bool* A, int first, int last) {
   cout << "flip_bits called  with [" << first << ", " << last << "]" << endl;

   int partition_size = (last - first) / 3;

   if (last <= first + 1) {
         A[first] = 1;
         return;
   }

   flip_bits(A, first + 1, first + partition_size);
   flip_bits(A, first + partition_size + 1, first + partition_size * 2);
   flip_bits(A, first + partition_size * 2 + 1, last);
}
```

Let n = last - first. Recurrence Relationship: $T(1) = a$, $T(n) = 3T\left(\frac{n}{3}\right) + b$ for n > 1

Explanation:

```
int main() {
   bool A[100] = {0};
   flip_bits(A, 25, 28);
   cout << A[25] << A[26] << A[27] << A[28] << endl;
   return 0;
}
```

Draw the call tree for **flip_bits(A, 25, 28);**

Call Tree:

2) Solve the recurrence relationship given above by unrolling the recurrence (use back substitution), and give T(n)'s order of growth in terms of the Big-O notation as a function of n. Show all steps in deriving your solution. Note that $k^{\log_k n} = n$.

$\mathrm{T}(1) = \mathrm{a}$, $\mathrm{T}(\mathrm{n}) = 3\mathrm{T}\left(\frac{\mathrm{n}}{3}\right) + \mathrm{b}$ for n > 1

Solution: