Lecture #22: Application — Analysis of Algorithms
What Will Happen During the Lecture

Remember... You Had Homework!

Students were asked to work through the following set of lecture notes before this lecture.

- Lecture Notes — "Application: Analysis of Algorithms".

As always, you may attend the lecture presentation if you have not worked through this material ahead of time — but it will not be repeated for you, and you might get a little bit lost, during the presentation, if you haven't worked through this.

While some students have indicated that most of this material, in a previous section of the course, others stated that they had seen almost none of it before this

Problems To Be Solved

The lecture presentation included a ***randomized algorithm*** to check whether an input integer `key` was stored in an input integer array `A` — as given in Figure 1 on page 2. This algorithm calls a variant of a "Linear Search" algorithm as a subroutine — as shown in Figure 2 on page 2.

During the lecture, material from the preparatory reading will be used to analyze the running time of this algorithm.

The preparatory reading also included an analysis of another randomized algorithm ending with a complicated — and not very helpful — expression. If time permits, this expression will be simplified (and the techniques that were used to do this will be highlighted).

```
boolean rSearch3 (integer[] A, integer key) {
1.  integer n := A.length
2.  integer i := 0
3.  while (i < n) {
4.    Choose j uniformly from the set {0, 1, 2, ..., n − 1} — independently from any
      previous selections.
5.    if (A[j] == key) {
6.      return true
      }
7.    i := i + 1
    }
8.  return dSearch(A, key)
}
```

Figure 1: Randomized Algorithm for Searching in an Integer Array

```
integer dSearch (integer[] A, integer key) {
1.  integer n := A.length
2.  integer i := 0
3.  while (i < n) {
4.    if (A[i] == key) {
5.      return true
      }
6.    i := i + 1
    }
7.  return false
}
```

Figure 2: Subroutine Implementing Linear Search