The three principles of algorithm analysis (stated for running time, although they apply to space):

1. Measure time as a function of input size
2. Ignore multiplicative constants and lower order terms
3. Look at time as input size goes to infinity

The five relations for comparing run times $A(n)$ and $B(n)$ according to these principles are:

- $A(n) = \Theta(B(n))$ if and only if $\lim_{n\to\infty} \frac{A(n)}{B(n)}$ is greater than 0 and less than $\infty$
- $A(n) = O(B(n))$ if and only if $\lim_{n\to\infty} \frac{A(n)}{B(n)}$ is less than $\infty$
- $A(n) = o(B(n))$ if and only if $\lim_{n\to\infty} \frac{A(n)}{B(n)}$ is 0
- $A(n) = \Omega(B(n))$ if and only if $\lim_{n\to\infty} \frac{A(n)}{B(n)}$ is greater than 0
- $A(n) = \omega(B(n))$ if and only if $\lim_{n\to\infty} \frac{A(n)}{B(n)}$ is $\infty$.

The *average running time* $A(n)$ of an algorithm $A$ on a particular probability distribution is

$$\sum_{I \in \mathcal{I}_n} (\text{Probability of } I \text{ given } I \in \mathcal{I}_n) \cdot (\text{Running time of } A \text{ on input } I\ )$$

Here $\mathcal{I}_n$ represents the collection of inputs of size $n$. The important point is that the average case case time depends on the input distribution. It is quite possible that the average case time for $A$ will be different for different input distributions.

The *expected running time* $A(n)$ of a randomized algorithm $A$

$$\max_{I \in \mathcal{I}_n} \text{Expected time of } A \text{ on } I$$

Here $\mathcal{I}_n$ represents the collection of inputs of size $n$. The important point is that you are taking the worst case input, and looking at the expectation over random events internal to the algorithm.