figure 5.9

Meanings of the various growth functions

Mathematical Expression	Relative Rates of Growth
$T(N) = O(F(N))$	Growth of $T(N)$ is $\leq$ growth of $F(N)$.
$T(N) = \Omega(F(N))$	Growth of $T(N)$ is $\geq$ growth of $F(N)$.
$T(N) = \Theta(F(N))$	Growth of $T(N)$ is $=$ growth of $F(N)$.
$T(N) = o(F(N))$	Growth of $T(N)$ is $<$ growth of $F(N)$.

To be more precise

- T(N) is O(F(N)) if there are c and N_0 such that
  for all N>=N_0 : T(N) <= c*F(N)

Binary Search

```
public static int binarySearch(int[]a, int e){
  int low = 0;
  int high = a.length-1;
  int mid;
  while (low <= high) {
     mid = (low + high) / 2;
     if (a[mid] < e) low = mid + 1;
     else if (a[mid] > e) high = mid - 1;
     else return mid;
     }
  return -1;
}
```


Binary Search – Best Case Analysis

- Size of array is of length *n*.
- In **best** case, the element we are looking for is in the center position of the array.
- In this case, we have one comparison.
- O(1)

Binary Search – Worst Case Analysis

- Size of array is of length *n*.
- What is the **worst** case scenario?
- What if n = 1?
- What if n = 2, 3, 4, 5, 6, 7, 8, 15, 16?

Binary Search – Worst Case Analysis

N	Comparisons
1	1
2	2
3	2
4	3
5	3
6	3
7	3
8	4
15	4
16	5

Binary Search – Worst Case Analysis

- Comparisons: floor($\log_2$ n) + 1
- O(log n)

Binary Search – Average Case Analysis

Index:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Comps:	4	3	4	2	4	3	4	1	4	3	4	2	4	3	4

Sum: 8/15 * 4 + 4/15 * 3 + 2/15 *2 + 1/15 * 1
<= 8/15 * 4 + 4/15 * 4 + 2/15 *4 + 1/15 * 4
= (8/15 + 4/15 + 2/15 + 1/15) * 4
= (8/15 + 4/15 + 2/15 + 1/15) * floor(log(n)+1)
= 1 * floor(log(n)+1)
= O(log(n))