Space Complexity

Space complexity is a measure of the amount of memory an algorithm uses, in terms of the sizes of the input. Like time complexity, it is often expressed using big O notation.

Need for Space Complexity

The amount of space a system has can be limited and therefore we need to optimize the memory/space taken by the algorithm to execute on that particular system with bounded space limits.

Let's understand space complexity with different examples:

Example 1:

Code:

```
void printHello() {
    String hello = "Hello, World!";
    System.out.println(hello);
}
```

Explanation:
The above code has constant space complexity because the amount of memory required by the code does not depend on the size of the input. So, This is O(1) space complexity code.
Example 2:

Code:

```
int[] copyArray(int[] arr) {
    int[] copy = new int[arr.length];
    for (int i = 0; i < arr.length; i++) {
        copy[i] = arr[i];
    }
    return copy;
```

Explanation:
The above code has O(n) space complexity i.e linear space complexity because we are creating an array of similar size as taken as a parameter in the function. So, more the length of the parameter array means more will be the length of the copy array resulting in direct relation. That's why O(n) space complexity.

Example 3:

Code:

```
int addUpto(int n){
    if (n ⩽ 0){
        return 0;
    }
    return n + addUpto(n-1);
}
```