

# String

---

## Key Points about String

- **Immutable:** Once created, String content cannot be changed
  - **Reference type:** Stored in heap memory
  - **Special pool:** String literals stored in String pool (part of heap)
  - **Final class:** Cannot be extended
  - **Thread-safe:** Due to immutability
- 

## String Methods with Examples

**Input String:** "abcdE"

### 1. charAt(int index)

- **Definition:** Returns the character at specified index position
  - **Example:** s1.charAt(2)
  - **Output:** 'c'
- 

### 2. substring(int beginIndex)

- **Definition:** Returns substring from beginIndex to end of string
  - **Example:** s1.substring(2)
  - **Output:** "cdE"
- 

### 3. substring(int beginIndex, int endIndex)

- **Definition:** Returns substring from beginIndex to endIndex-1 (endIndex excluded)
  - **Example:** s1.substring(1, 4)
  - **Output:** "bcd"
- 

### 4. length()

- **Definition:** Returns total number of characters in string
  - **Example:** s1.length()
  - **Output:** 5
-

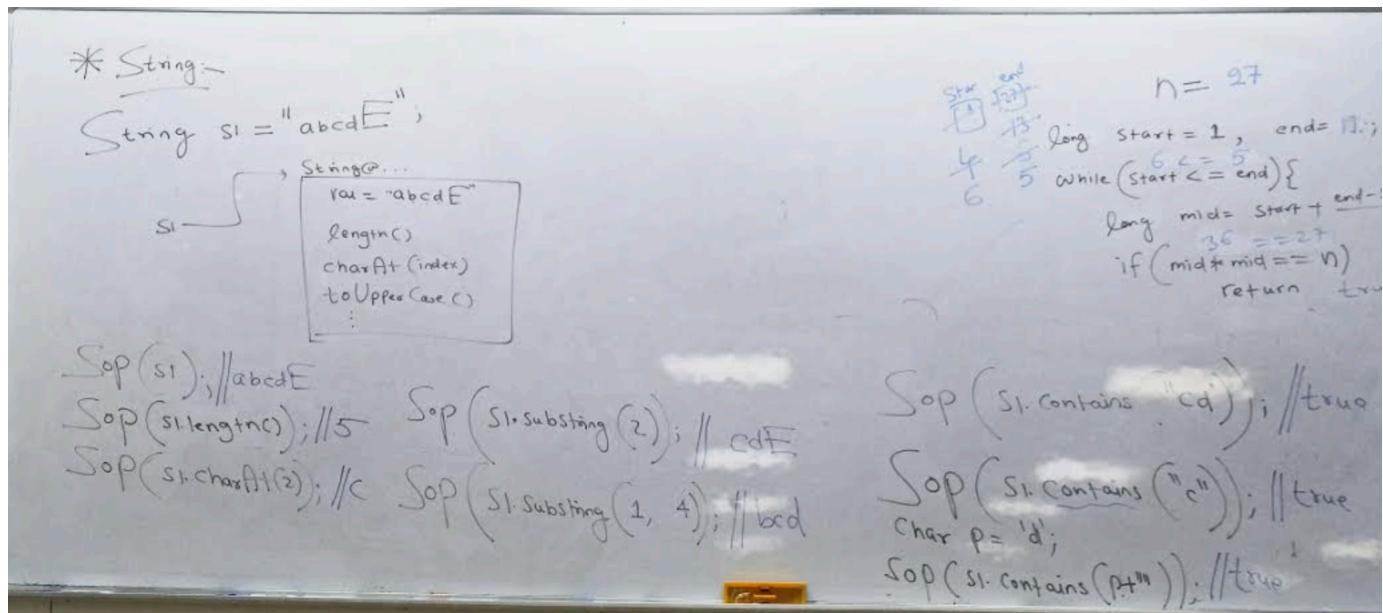
## 5. toUpperCase()

- **Definition:** Converts all characters to uppercase
- **Example:** `s1.toUpperCase()`
- **Output:** "ABCDE"

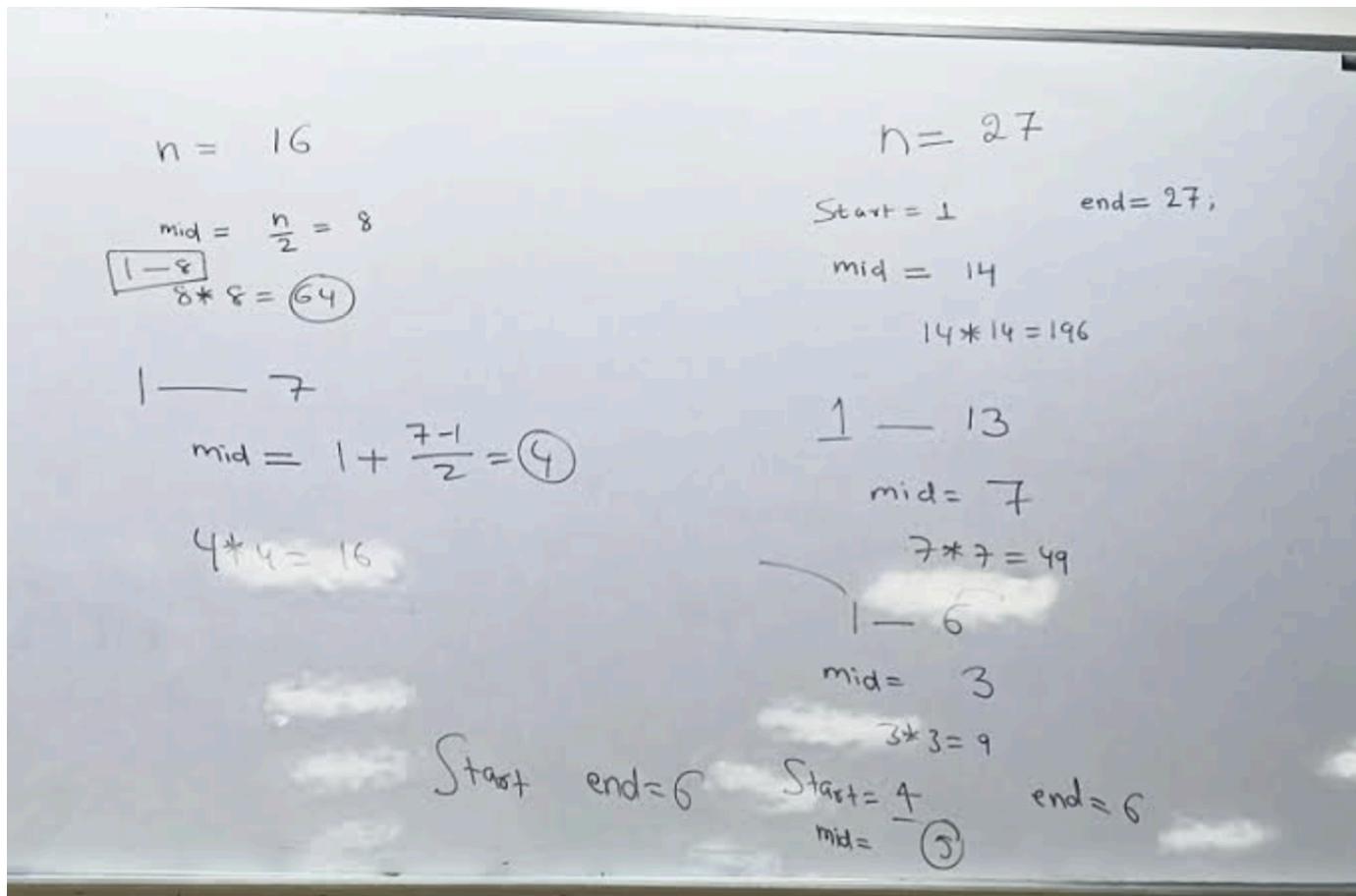
## 6. contains(CharSequence sequence)

- **Definition:** Checks if string contains specified sequence of characters
- **Example:** `s1.contains("cd")`
- **Output:** `true`
- **Example:** `s1.contains("xy")`
- **Output:** `false`

**Note:** String class has both instance methods (like above) and static methods (like `valueOf()`, `format()`)



# Leetcode code Explanation



Brave File Edit View History Bookmarks Profiles Tab Window Help Tue 2 Dec 5:38 PM

Sq(x) - LeetCode Valid Perfect Square - LeetCode avinash-01 - LeetCode Profile +

leetcode.com/problems/valid-perfect-square/description/ Problem List Submit Premium

**367. Valid Perfect Square** Solved

Easy Topics Companies

Given a positive integer num, return true if num is a perfect square or false otherwise.

A **perfect square** is an integer that is the square of an integer. In other words, it is the product of some integer with itself.

You must not use any built-in library function, such as `sqrt`.

**Example 1:**

**Input:** num = 16  
**Output:** true  
**Explanation:** We return true because  $4 * 4 = 16$  and 4 is an integer.

**Example 2:**

**Input:** num = 14  
**Output:** false  
**Explanation:** We return false because  $3.742 * 3.742 = 14$  and 3.742 is not an integer.

4.6K 107 • 34 Online

**Code**

```
Java Auto
1 class Solution {
2     public boolean isPerfectSquare(long x) {
3         long start = 1;
4         long end = x;
5
6         while (start <= end) {
7             long mid = start + (end - start) / 2;
8
9             if (mid * mid == x)
10                return true;
11            else if (mid * mid > x)
12                end = mid - 1;
13            else
14                start = mid + 1;
15        }
16        return false;
17    }
18 }
```

Ln 4, Col 20

Testcase Test Result

Accepted Runtime: 0 ms

Case 1 Case 2 Case 3

Input

The screenshot shows a LeetCode problem page for "Sqrt(x)". The problem description asks for the square root of a non-negative integer  $x$ , rounded down to the nearest integer. It specifies that the returned integer should be non-negative as well. A note states that built-in exponent functions or operators must not be used.

**Example 1:**

- Input:**  $x = 4$
- Output:** 2
- Explanation:** The square root of 4 is 2, so we return 2.

**Example 2:**

- Input:**  $x = 8$
- Output:** 2
- Explanation:** The square root of 8 is 2.82842..., and since we round it down to the nearest integer, 2 is returned.

Below the examples, there are statistics: 9.4K views, 392 likes, and 175 online users. The code editor shows a Java solution using binary search:

```

1 class Solution {
2     public int mySqrt(int x) {
3         long start = 1;
4         long end = x;
5
6         while (start <= end) {
7             long mid = start + (end - start) / 2;
8
9             if (mid * mid == x)
10                return (int)mid;
11             else if (mid * mid > x)
12                end = mid - 1;
13             else
14                start = mid + 1;
15         }
16         return (int)(start - 1);
17     }
18 }
19

```

The code is saved, and the test result shows it is accepted with a runtime of 0 ms. All three test cases (Case 1, Case 2, Case 3) are checked.