# Problem 3744: Find Kth Character in Expanded String

## Problem Information

**Difficulty:** Medium
**Acceptance Rate:** 0.00%
**Paid Only:** No

## Problem Description

You are given a string

s

consisting of one or more words separated by single spaces. Each word in

s

consists of lowercase English letters.

We obtain the

expanded

string

t

from

s

as follows:

For each

word

in

s

, repeat its first character once, then its second character twice, and so on.

For example, if

s = "hello world"

, then

t = "heelllllllooooo woorrrllllddddd"

.

You are also given an integer

k

, representing a

valid

index of the string

t

.

Return the

k

th

character of the string

t

.

Example 1:

Input:

s = "hello world", k = 0

Output:

"h"

Explanation:

t = "heellllllooooo woorrrllllddddd"

. Therefore, the answer is

t[0] = "h"

.

Example 2:

Input:

s = "hello world", k = 15

Output:

" "

Explanation:

t = "heellllllooooo woorrrllllddddd"

. Therefore, the answer is

t[15] = " "

.

Constraints:

1 <= s.length <= 10

5

s

contains only lowercase English letters and spaces

' '

.

s

does not contain

any leading or trailing spaces.

All the words in

s

are separated by a

single space

.

0 <= k < t.length

. That is,

k

is a

valid

index of

t

.

## Code Snippets

**C++:**

```
class Solution {
public:
char kthCharacter(string s, long long k) {

}
};
```

**Java:**

```
class Solution {
public char kthCharacter(String s, long k) {

}
}
```

**Python3:**

```
class Solution:
def kthCharacter(self, s: str, k: int) -> str:
```

**Python:**

```
class Solution(object):
def kthCharacter(self, s, k):
"""
:type s: str
:type k: int
:rtype: str
"""
```

**JavaScript:**

```
/**
* @param {string} s
* @param {number} k
* @return {character}
*/
var kthCharacter = function(s, k) {

};
```

**TypeScript:**

```
function kthCharacter(s: string, k: number): string {

};
```

**C#:**

```
public class Solution {
public char KthCharacter(string s, long k) {

}
}
```

**C:**

```
char kthCharacter(char* s, long long k) {

}
```

**Go:**

```
func kthCharacter(s string, k int64) byte {
```

```
}
```

**Kotlin:**

```
class Solution {
fun kthCharacter(s: String, k: Long): Char {

}
}
```

**Swift:**

```
class Solution {
func kthCharacter(_ s: String, _ k: Int) -> Character {

}
}
```

**Rust:**

```
impl Solution {
pub fn kth_character(s: String, k: i64) -> char {

}
}
```

**Ruby:**

```
# @param {String} s
# @param {Integer} k
# @return {Character}
def kth_character(s, k)

end
```

**PHP:**

```
class Solution {

/**
* @param String $s
* @param Integer $k
```

```
* @return String
*/
function kthCharacter($s, $k) {

}
}
```

**Dart:**

```
class Solution {
String kthCharacter(String s, int k) {

}
}
```

**Scala:**

```
object Solution {
def kthCharacter(s: String, k: Long): Char = {

}
}
```

**Elixir:**

```
defmodule Solution do
@spec kth_character(s :: String.t, k :: integer) :: char
def kth_character(s, k) do

end
end
```

**Erlang:**

```
-spec kth_character(S :: unicode:unicode_binary(), K :: integer()) -> char().
kth_character(S, K) ->
.
```

**Racket:**

```
(define/contract (kth-character s k)
(-> string? exact-integer? char?)
```

```
)
```

## Solutions

### C++ Solution:

```
/*
* Problem: Find Kth Character in Expanded String
* Difficulty: Medium
* Tags: string
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

class Solution {
public:
char kthCharacter(string s, long long k) {

}
};
```

### Java Solution:

```
/**
* Problem: Find Kth Character in Expanded String
* Difficulty: Medium
* Tags: string
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

class Solution {
public char kthCharacter(String s, long k) {

}
}
```

### Python3 Solution:

```
"""
Problem: Find Kth Character in Expanded String
Difficulty: Medium
Tags: string

Approach: String manipulation with hash map or two pointers
Time Complexity: O(n) or O(n log n)
Space Complexity: O(1) to O(n) depending on approach
"""

class Solution:
def kthCharacter(self, s: str, k: int) -> str:
# TODO: Implement optimized solution
pass
```

### Python Solution:

```
class Solution(object):
def kthCharacter(self, s, k):
"""
:type s: str
:type k: int
:rtype: str
"""
```

### JavaScript Solution:

```
/**
* Problem: Find Kth Character in Expanded String
* Difficulty: Medium
* Tags: string
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

/**
* @param {string} s
* @param {number} k
```

```
* @return {character}
*/
var kthCharacter = function(s, k) {

};
```

**TypeScript Solution:**

```
/**
* Problem: Find Kth Character in Expanded String
* Difficulty: Medium
* Tags: string
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

function kthCharacter(s: string, k: number): string {

};
```

**C# Solution:**

```
/*
* Problem: Find Kth Character in Expanded String
* Difficulty: Medium
* Tags: string
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

public class Solution {
public char KthCharacter(string s, long k) {

}
}
```

**C Solution:**

```c
/*
* Problem: Find Kth Character in Expanded String
* Difficulty: Medium
* Tags: string
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

char kthCharacter(char* s, long long k) {

}
```

### Go Solution:

```go
// Problem: Find Kth Character in Expanded String
// Difficulty: Medium
// Tags: string
//
// Approach: String manipulation with hash map or two pointers
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(1) to O(n) depending on approach

func kthCharacter(s string, k int64) byte {

}
```

### Kotlin Solution:

```kotlin
class Solution {
fun kthCharacter(s: String, k: Long): Char {

}
}
```

### Swift Solution:

```swift
class Solution {
func kthCharacter(_ s: String, _ k: Int) -> Character {

}
```

```
}
```

### Rust Solution:

```
// Problem: Find Kth Character in Expanded String
// Difficulty: Medium
// Tags: string
//
// Approach: String manipulation with hash map or two pointers
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(1) to O(n) depending on approach

impl Solution {
pub fn kth_character(s: String, k: i64) -> char {

}
}
```

### Ruby Solution:

```
# @param {String} s
# @param {Integer} k
# @return {Character}
def kth_character(s, k)

end
```

### PHP Solution:

```
class Solution {

/**
* @param String $s
* @param Integer $k
* @return String
*/
function kthCharacter($s, $k) {

}
}
```

### Dart Solution:

```
class Solution {
String kthCharacter(String s, int k) {

}
}
```

### Scala Solution:

```
object Solution {
def kthCharacter(s: String, k: Long): Char = {

}
}
```

### Elixir Solution:

```
defmodule Solution do
@spec kth_character(s :: String.t, k :: integer) :: char
def kth_character(s, k) do

end
end
```

### Erlang Solution:

```
-spec kth_character(S :: unicode:unicode_binary(), K :: integer()) -> char().
kth_character(S, K) ->
.
```

### Racket Solution:

```
(define/contract (kth-character s k)
(-> string? exact-integer? char?)
)
```