# Problem 2496: Maximum Value of a String in an Array

## Problem Information

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

## Problem Description

The

value

of an alphanumeric string can be defined as:

The

numeric

representation of the string in base

10

, if it comprises of digits

only

.

The

length

of the string, otherwise.

Given an array

strs

of alphanumeric strings, return

the

maximum value

of any string in

strs

.

Example 1:

Input:

strs = ["alic3","bob","3","4","00000"]

Output:

5

Explanation:

- "alic3" consists of both letters and digits, so its value is its length, i.e. 5. - "bob" consists only of letters, so its value is also its length, i.e. 3. - "3" consists only of digits, so its value is its numeric equivalent, i.e. 3. - "4" also consists only of digits, so its value is 4. - "00000" consists only of digits, so its value is 0. Hence, the maximum value is 5, of "alic3".

Example 2:

Input:

strs = ["1","01","001","0001"]

Output:

1

Explanation:

Each string in the array has value 1. Hence, we return 1.

Constraints:

1 <= strs.length <= 100

1 <= strs[i].length <= 9

strs[i]

consists of only lowercase English letters and digits.

## Code Snippets

**C++:**

```
class Solution {
public:
int maximumValue(vector<string>& strs) {

}
};
```

**Java:**

```
class Solution {
public int maximumValue(String[] strs) {

}
}
```

**Python3:**

```
class Solution:
def maximumValue(self, strs: List[str]) -> int:
```

**Python:**

```
class Solution(object):
def maximumValue(self, strs):
"""
:type strs: List[str]
:rtype: int
"""
```

**JavaScript:**

```
/**
* @param {string[]} strs
* @return {number}
*/
var maximumValue = function(strs) {

};
```

**TypeScript:**

```
function maximumValue(strs: string[]): number {

};
```

**C#:**

```
public class Solution {
public int MaximumValue(string[] strs) {

}
}
```

**C:**

```
int maximumValue(char** strs, int strsSize) {

}
```

**Go:**

```
func maximumValue(strs []string) int {

}
```

**Kotlin:**

```
class Solution {
fun maximumValue(strs: Array<String>): Int {

}
}
```

**Swift:**

```
class Solution {
func maximumValue(_ strs: [String]) -> Int {

}
}
```

**Rust:**

```
impl Solution {
pub fn maximum_value(strs: Vec<String>) -> i32 {

}
}
```

**Ruby:**

```
# @param {String[]} strs
# @return {Integer}
def maximum_value(strs)

end
```

**PHP:**

```
class Solution {

/**
* @param String[] $strs
* @return Integer
```

```
*/
function maximumValue($strs) {

}
}
```

**Dart:**

```
class Solution {
int maximumValue(List<String> strs) {

}
}
```

**Scala:**

```
object Solution {
def maximumValue(strs: Array[String]): Int = {

}
}
```

**Elixir:**

```
defmodule Solution do
@spec maximum_value(strs :: [String.t]) :: integer
def maximum_value(strs) do

end
end
```

**Erlang:**

```
-spec maximum_value(Strs :: [unicode:unicode_binary()]) -> integer().
maximum_value(Strs) ->
.
```

**Racket:**

```
(define/contract (maximum-value strs)
(-> (listof string?) exact-integer?)
)
```

## Solutions

### C++ Solution:

```
/*
* Problem: Maximum Value of a String in an Array
* Difficulty: Easy
* Tags: array, string
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

class Solution {
public:
int maximumValue(vector<string>& strs) {

}
};
```

### Java Solution:

```
/**
* Problem: Maximum Value of a String in an Array
* Difficulty: Easy
* Tags: array, string
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

class Solution {
public int maximumValue(String[] strs) {

}
}
```

### Python3 Solution:

```
"""
Problem: Maximum Value of a String in an Array
Difficulty: Easy
Tags: array, string

Approach: Use two pointers or sliding window technique
Time Complexity: O(n) or O(n log n)
Space Complexity: O(1) to O(n) depending on approach
"""

class Solution:
def maximumValue(self, strs: List[str]) -> int:
# TODO: Implement optimized solution
pass
```

### Python Solution:

```
class Solution(object):
def maximumValue(self, strs):
"""
:type strs: List[str]
:rtype: int
"""
```

### JavaScript Solution:

```
/**
* Problem: Maximum Value of a String in an Array
* Difficulty: Easy
* Tags: array, string
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

/**
* @param {string[]} strs
* @return {number}
*/
var maximumValue = function(strs) {
```

```
};
```

### TypeScript Solution:

```
/**
* Problem: Maximum Value of a String in an Array
* Difficulty: Easy
* Tags: array, string
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

function maximumValue(strs: string[]): number {

};
```

### C# Solution:

```
/*
* Problem: Maximum Value of a String in an Array
* Difficulty: Easy
* Tags: array, string
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

public class Solution {
public int MaximumValue(string[] strs) {

}
}
```

### C Solution:

```
/*
* Problem: Maximum Value of a String in an Array
* Difficulty: Easy
```

```
* Tags: array, string
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

int maximumValue(char** strs, int strsSize) {

}
```

### Go Solution:

```
// Problem: Maximum Value of a String in an Array
// Difficulty: Easy
// Tags: array, string
//
// Approach: Use two pointers or sliding window technique
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(1) to O(n) depending on approach

func maximumValue(strs []string) int {

}
```

### Kotlin Solution:

```
class Solution {
fun maximumValue(strs: Array<String>): Int {

}
}
```

### Swift Solution:

```
class Solution {
func maximumValue(_ strs: [String]) -> Int {

}
}
```

### Rust Solution:

```
// Problem: Maximum Value of a String in an Array
// Difficulty: Easy
// Tags: array, string
//
// Approach: Use two pointers or sliding window technique
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(1) to O(n) depending on approach

impl Solution {
pub fn maximum_value(strs: Vec<String>) -> i32 {

}
}
```

### Ruby Solution:

```
# @param {String[]} strs
# @return {Integer}
def maximum_value(strs)

end
```

### PHP Solution:

```
class Solution {

/**
* @param String[] $strs
* @return Integer
*/
function maximumValue($strs) {

}
}
```

### Dart Solution:

```
class Solution {
int maximumValue(List<String> strs) {
```

```
}
}
```

### Scala Solution:

```
object Solution {
def maximumValue(strs: Array[String]): Int = {

}
}
```

### Elixir Solution:

```
defmodule Solution do
@spec maximum_value(strs :: [String.t]) :: integer
def maximum_value(strs) do

end
end
```

### Erlang Solution:

```
-spec maximum_value(Strs :: [unicode:unicode_binary()]) -> integer().
maximum_value(Strs) ->
.
```

### Racket Solution:

```
(define/contract (maximum-value strs)
(-> (listof string?) exact-integer?)
)
```