Problem 670: Maximum Swap

Problem Information

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

Problem Description

You are given an integer

num

. You can swap two digits at most once to get the maximum valued number.

Return

the maximum valued number you can get

.

Example 1:

Input:

num = 2736

Output:

7236

Explanation:

Swap the number 2 and the number 7.

Example 2:

Input:

num = 9973

Output:

9973

Explanation:

No swap.

Constraints:

0 <= num <= 10

8

Code Snippets

C++:

```
class Solution {
public:
int maximumSwap(int num) {

}
};
```

Java:

```
class Solution {
public int maximumSwap(int num) {

}
}
```

Python3:

```
class Solution:
def maximumSwap(self, num: int) -> int:
```

Python:

```
class Solution(object):
def maximumSwap(self, num):
"""
:type num: int
:rtype: int
"""
```

JavaScript:

```
/**
* @param {number} num
* @return {number}
*/
var maximumSwap = function(num) {

};
```

TypeScript:

```
function maximumSwap(num: number): number {

};
```

C#:

```
public class Solution {
public int MaximumSwap(int num) {

}
}
```

C:

```
int maximumSwap(int num) {

}
```

Go:

```
func maximumSwap(num int) int {

}
```

Kotlin:

```
class Solution {
fun maximumSwap(num: Int): Int {

}
}
```

Swift:

```
class Solution {
func maximumSwap(_ num: Int) -> Int {

}
}
```

Rust:

```
impl Solution {
pub fn maximum_swap(num: i32) -> i32 {

}
}
```

Ruby:

```
# @param {Integer} num
# @return {Integer}
def maximum_swap(num)

end
```

PHP:

```
class Solution {

/**
* @param Integer $num
* @return Integer
*/
function maximumSwap($num) {

}
}
```

Dart:

```
class Solution {
int maximumSwap(int num) {

}
}
```

Scala:

```
object Solution {
def maximumSwap(num: Int): Int = {

}
}
```

Elixir:

```
defmodule Solution do
@spec maximum_swap(num :: integer) :: integer
def maximum_swap(num) do

end
end
```

Erlang:

```
-spec maximum_swap(Num :: integer()) -> integer().
maximum_swap(Num) ->
.
```

Racket:

```
(define/contract (maximum-swap num)
(-> exact-integer? exact-integer?)
)
```

Solutions

C++ Solution:

```
/*
* Problem: Maximum Swap
* Difficulty: Medium
* Tags: greedy, math
*
* Approach: Greedy algorithm with local optimal choices
* Time Complexity: O(n) to O(n^2) depending on approach
* Space Complexity: O(1) to O(n) depending on approach
*/

class Solution {
public:
int maximumSwap(int num) {

}
};
```

Java Solution:

```
/**
* Problem: Maximum Swap
* Difficulty: Medium
* Tags: greedy, math
*
* Approach: Greedy algorithm with local optimal choices
* Time Complexity: O(n) to O(n^2) depending on approach
* Space Complexity: O(1) to O(n) depending on approach
*/

class Solution {
public int maximumSwap(int num) {

}
}
```

Python3 Solution:

```
"""
Problem: Maximum Swap
Difficulty: Medium
Tags: greedy, math

Approach: Greedy algorithm with local optimal choices
Time Complexity: O(n) to O(n^2) depending on approach
Space Complexity: O(1) to O(n) depending on approach
"""

class Solution:
def maximumSwap(self, num: int) -> int:
# TODO: Implement optimized solution
pass
```

Python Solution:

```
class Solution(object):
def maximumSwap(self, num):
"""
:type num: int
:rtype: int
"""
```

JavaScript Solution:

```
/**
* Problem: Maximum Swap
* Difficulty: Medium
* Tags: greedy, math
*
* Approach: Greedy algorithm with local optimal choices
* Time Complexity: O(n) to O(n^2) depending on approach
* Space Complexity: O(1) to O(n) depending on approach
*/

/**
```

```
* @param {number} num
* @return {number}
*/
var maximumSwap = function(num) {

};
```

TypeScript Solution:

```
/**
* Problem: Maximum Swap
* Difficulty: Medium
* Tags: greedy, math
*
* Approach: Greedy algorithm with local optimal choices
* Time Complexity: O(n) to O(n^2) depending on approach
* Space Complexity: O(1) to O(n) depending on approach
*/

function maximumSwap(num: number): number {

};
```

C# Solution:

```
/*
* Problem: Maximum Swap
* Difficulty: Medium
* Tags: greedy, math
*
* Approach: Greedy algorithm with local optimal choices
* Time Complexity: O(n) to O(n^2) depending on approach
* Space Complexity: O(1) to O(n) depending on approach
*/

public class Solution {
public int MaximumSwap(int num) {

}
}
```

C Solution:

```
/*
* Problem: Maximum Swap
* Difficulty: Medium
* Tags: greedy, math
*
* Approach: Greedy algorithm with local optimal choices
* Time Complexity: O(n) to O(n^2) depending on approach
* Space Complexity: O(1) to O(n) depending on approach
*/

int maximumSwap(int num) {

}
```

Go Solution:

```
// Problem: Maximum Swap
// Difficulty: Medium
// Tags: greedy, math
//
// Approach: Greedy algorithm with local optimal choices
// Time Complexity: O(n) to O(n^2) depending on approach
// Space Complexity: O(1) to O(n) depending on approach

func maximumSwap(num int) int {

}
```

Kotlin Solution:

```
class Solution {
fun maximumSwap(num: Int): Int {

}
}
```

Swift Solution:

```
class Solution {
func maximumSwap(_ num: Int) -> Int {
}
}
```

Rust Solution:

```
// Problem: Maximum Swap
// Difficulty: Medium
// Tags: greedy, math
//
// Approach: Greedy algorithm with local optimal choices
// Time Complexity: O(n) to O(n^2) depending on approach
// Space Complexity: O(1) to O(n) depending on approach

impl Solution {
pub fn maximum_swap(num: i32) -> i32 {

}
}
```

Ruby Solution:

```
# @param {Integer} num
# @return {Integer}
def maximum_swap(num)

end
```

PHP Solution:

```
class Solution {

/**
* @param Integer $num
* @return Integer
*/
function maximumSwap($num) {

}
}
```

Dart Solution:

```
class Solution {
int maximumSwap(int num) {

}
}
```

Scala Solution:

```
object Solution {
def maximumSwap(num: Int): Int = {

}
}
```

Elixir Solution:

```
defmodule Solution do
@spec maximum_swap(num :: integer) :: integer
def maximum_swap(num) do

end
end
```

Erlang Solution:

```
-spec maximum_swap(Num :: integer()) -> integer().
maximum_swap(Num) ->
.
```

Racket Solution:

```
(define/contract (maximum-swap num)
(-> exact-integer? exact-integer?)
)
```