

Beat the competition and land yourself a top job. [Register for Job-a-thon now!](#)

# Build Lowest Number by Removing n digits from a given number

Difficulty Level : Hard • Last Updated : 27 Jun, 2022



Given a string 'str' of digits and an integer 'n', build the lowest possible number by removing 'n' digits from the string and not changing the order of input digits.

## Examples:

**Input:** str = "4325043", n = 3

**Output:** "2043"

**Input:** str = "765028321", n = 5

**Output:** "0221"

**Input:** str = "121198", n = 2

**Output:** "1118"

Recommended: Please solve it on "[PRACTICE](#)" first, before moving on to the solution.

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

**Got It !**

of first (n+1) digits and put it in result, and recur for the remaining characters. Below is complete algorithm.

Initialize result as empty string

```
res = ""
```

**buildLowestNumber(str, n, res)**

1) If  $n == 0$ , then there is nothing to remove.

Append the whole 'str' to 'res' and return

2) Let 'len' be length of 'str'. If 'len' is smaller or equal to n, then everything can be removed

Append nothing to 'res' and return

3) Find the smallest character among first (n+1) characters of 'str'. Let the index of smallest character be minIndex. Append 'str[minIndex]' to 'res' and recur for substring after minIndex and for  $n = n - \text{minIndex}$

```
buildLowestNumber(str[minIndex+1..len-1], n-minIndex).
```

Below is the implementation of the above algorithm:

---

## C++

// C++ program for the above approach

```
#include <bits/stdc++.h>
```

```
using namespace std;
```

```
string removeKdigits(string num, int k)
```

```
{
```

```
    int n = num.size();
```

```
    stack<char> mystack;
```

```
    // Store the final string in stack
```

---

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

**Got It !**

```

        k -= 1;
    }

    if (!mystack.empty() || c != '0'){
        mystack.push(c);
    }
}

// Now remove the largest values from the top of the
// stack
while (!mystack.empty() && k-->0)
    mystack.pop();
if (mystack.empty())
    return "0";

// Now retrieve the number from stack into a string
// (reusing num)
while (!mystack.empty()) {
    num[n - 1] = mystack.top();
    mystack.pop();
    n -= 1;
}
return num.substr(n);
}

int main()
{
    string str = "765028321";
    int k = 5;
    cout << removeKdigits(str, k);
    return 0;
}

```

## Java

```

// Java program for the above approach
import java.io.*;
import java.util.*;

class GFG {

```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

**Got It !**

```

// We have to delete all digits
if (k >= num.length()) {
    return "0";
}
// Nothing to delete
if (k == 0) {
    return num;
}
Stack<Character> s = new Stack<Character>();

for (int i = 0; i < num.length(); i++) {
    char c = num.charAt(i);

    // Removing all digits in stack that are greater
    // than this digit(since they have higher
    // weightage)
    while (!s.isEmpty() && k > 0 && s.peek() > c) {
        s.pop();
        k--;
    }
    // ignore pushing 0
    if (!s.isEmpty() || c != '0')
        s.push(c);
}

// If our k isnt 0 yet then we keep popping out the
// stack until k becomes 0
while (!s.isEmpty() && k > 0) {
    k--;
    s.pop();
}
if (s.isEmpty())
    return "0";
while (!s.isEmpty()) {
    result.append(s.pop());
}
String str = result.reverse().toString();

return str;
}

```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

**Got It !**

```

        String s = "765028321";
        int k = 5;
        System.out.println(removeKdigits(s, 5));
    }
}
// this code is contributed by gireeshgudaparthi

```

## Python3

# Python program for the above approach

```

def removeKdigits(num, k):

    n = len(num)
    mystack = []

    # Store the final string in stack
    for c in num:
        while (len(mystack) > 0 and k > 0 and ord(mystack[len(mystack)-1]
            > ord(c)):
            mystack.pop()
            k -= 1

        if len(mystack) > 0 or c != '0':
            mystack.append(c)

    # Now remove the largest values from the top of the
    # stack
    while len(mystack) > 0 and k:
        mystack.pop()
        k -= 1
    if len(mystack) == 0:
        return "0"

    # Now retrieve the number from stack into a string
    # (reusing num)
    while(len(mystack) > 0):
        num = num.replace(num[n - 1],mystack[len(mystack) - 1])
        mystack.pop()
        n -= 1
    return num[n:]

```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

**Got It !**

# This code is contributed by shinjanpatra

## C#

```
// C# program for the above approach
using System;
using System.Collections.Generic;
using System.Collections;
class HelloWorld {

    static string removeKdigits(string Num, int k)
    {
        char[] num = Num.ToCharArray();
        int n = num.Length;
        Stack<char> mystack = new Stack<char>();

        // Store the final string in stack
        for (int i = 0; i < num.Length; i++) {
            while (mystack.Count > 0 && k > 0
                && mystack.Peek() > num[i]) {
                mystack.Pop();
                k = k - 1;
            }

            if (mystack.Count > 0 || num[i] != '0') {
                mystack.Push(num[i]);
            }
        }

        // Now remove the largest values from the top of the
        // stack
        while (mystack.Count > 0 && k > 0) {
            mystack.Pop();
            k = k - 1;
        }

        if (mystack.Count == 0)
            return "0";
    }
}
```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

**Got It !**

```

        num[n - 1] = temp;
        mystack.Pop();
        n = n - 1;
    }

    return new string(num).Substring(n);
}

static void Main()
{
    string str = "765028321";
    int k = 5;
    Console.WriteLine(removeKdigits(str, k));
}
}

// The code is contributed by Nidhi goel

```

## JavaScript

<script>

```

// JavaScript program for the above approach
function removeKdigits(num,k)
{
    let n = num.length;
    let mystack = [];

    // Store the final string in stack
    for (let c of num)
    {
        while (mystack.length>0 && k > 0 &&
            mystack[mystack.length-1].charCodeAt(0) > c.charCodeAt(0))
        {
            mystack.pop();
            k -= 1;
        }

        if(mystack.length > 0 || c !== '0')

```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

**Got It !**

```

// Now remove the largest values from the top of the
// stack
while(mystack.length > 0 && k--){
    mystack.pop();
}
if (mystack.length == 0)
    return "0";

// Now retrieve the number from stack into a string
// (reusing num)
while(mystack.length > 0)
{
    num = num.split('');
    num[n - 1] = mystack[mystack.length - 1];
    num = num.join('');
    mystack.pop();
    n -= 1;
}
return num.substr(n);
}

// Driver code
let str = "765028321"
let k = 5
document.write(removeKdigits(str, k))

// This code is contributed by shinjanpatra

</script>

```

## Output

221

Below is an optimized code in C++ contributed by Gaurav Mamgain

---

## C++14

```

// C++14 implementation of the above program

```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

**Got It !**



```

void insertInNonDecOrder(deque<char>& dq, char ch)
{

    // If container is empty , insert the current digit
    if (dq.empty())
        dq.push_back(ch);

    else {
        char temp = dq.back();

        // Keep removing digits larger than current digit
        // from the back side of deque
        while (temp > ch && !dq.empty()) {
            dq.pop_back();
            if (!dq.empty())
                temp = dq.back();
        }

        // Insert the current digit
        dq.push_back(ch);
    }
    return;
}

string buildLowestNumber(string str, int n)
{
    int len = str.length();

    // Deleting n digits means we need to print k digits
    int k = len - n;

    deque<char> dq;
    string res = "";

    // Leaving rightmost k-1 digits we need to choose
    // minimum digit from rest of the string and print it
    int i;
    for (i = 0; i <= len - k; i++)

        // Insert new digit from the back side in
        // appropriate position and/ keep removing

```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

**Got It !**

```

while (i < len) {

    // keep the minimum digit in output string
    res += dq.front();

    // remove minimum digit
    dq.pop_front();

    // Again insert new digit from the back
    // side in appropriate position and keep
    // removing digits larger than current digit
    insertInNonDecOrder(dq, str[i]);
    i++;
}

// Now only one element will be there in the deque
res += dq.front();
dq.pop_front();
return res;
}

string lowestNumber(string str, int n)
{
    string res = buildLowestNumber(str, n);

    // Remove all the leading zeroes
    string ans = "";
    int flag = 0;
    for (int i = 0; i < res.length(); i++) {
        if (res[i] != '0' || flag == 1) {
            flag = 1;
            ans += res[i];
        }
    }

    if (ans.length() == 0)
        return "0";
    else
        return ans;
}

```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

**Got It !**

```
int n = 5;
cout <<lowestNumber(str, n) << endl;
return 0;
}
// This code is contributed by Gaurav Mangain
```

## Output

221

***Time Complexity:***  $O(N)$

***Space Complexity:***  $O(N)$

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

**Got It !**

### **Approach-2:**

*Let's suppose the length of the given string num be  $n$ . so the result string will contain the length of  $n-k$ .*

*As we proceed to solve this problem we should make sure that the output string contains minimum values at their high weightage positions. so we ensure that by using a stack.*

- 1. Return 0 if  $k \geq n$ . and return num if  $k=0$ .*
- 2. Create a stack and iterate through num string and push the value at that position if it is greater than the top element of the stack.*
- 3. Iterate through the num string and if the integer value at that position is less than the top of the stack we will pop the stack and decrement  $k$  until we reach the condition where the top of the stack is less than the value we are looking at (while  $k > 0$ ) (by this we are making sure that most significant positions of the result are filled with minimum values).*
- 4. If the  $k$  is still greater than 0 we will pop stack until  $k$  becomes 0.*
- 5. Append the elements in the stack to the result string.*
- 6. Delete leading zeroes from the result string.*

Below is the implementation of the above approach:

---

### **C++**

```
// C++ program for the above approach
```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

**Got It !**

```

string removeKdigits(string num, int k)
{
    int n = num.size();
    stack<char> mystack;
    // Store the final string in stack
    for (char c : num) {
        while (!mystack.empty() && k > 0
                && mystack.top() > c) {
            mystack.pop();
            k -= 1;
        }

        if (!mystack.empty() || c != '0')
            mystack.push(c);
    }

    // Now remove the largest values from the top of the
    // stack
    while (!mystack.empty() && k--)
        mystack.pop();
    if (mystack.empty())
        return "0";

    // Now retrieve the number from stack into a string
    // (reusing num)
    while (!mystack.empty()) {
        num[n - 1] = mystack.top();
        mystack.pop();
        n -= 1;
    }
    return num.substr(n);
}

int main()
{
    string str = "765028321";
    int k = 5;
    cout << removeKdigits(str, k);
    return 0;
}

```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

**Got It !**

```

import java.io.*;
import java.util.*;

class GFG {

    public static String removeKdigits(String num, int k)
    {
        StringBuilder result = new StringBuilder();

        // We have to delete all digits
        if (k >= num.length()) {
            return "0";
        }
        // Nothing to delete
        if (k == 0) {
            return num;
        }
        Stack<Character> s = new Stack<Character>();

        for (int i = 0; i < num.length(); i++) {
            char c = num.charAt(i);

            // Removing all digits in stack that are greater
            // than this digit(since they have higher
            // weightage)
            while (!s.isEmpty() && k > 0 && s.peek() > c) {
                s.pop();
                k--;
            }
            // ignore pushing 0
            if (!s.isEmpty() || c != '0')
                s.push(c);
        }

        // If our k isnt 0 yet then we keep popping out the
        // stack until k becomes 0
        while (!s.isEmpty() && k > 0) {
            k--;
            s.pop();
        }
        if (s.isEmpty())

```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

**Got It !**

```

        String str = result.reverse().toString();

        return str;
    }

    // Driver Code
    public static void main(String[] args)
    {
        String s = "765028321";
        int k = 5;
        System.out.println(removeKdigits(s, 5));
    }
}
// this code is contributed by gireeshgudaparthi

```

## Python3

```

# Python program for the above approach
def removeKdigits(num, k):

    n = len(num)
    mystack = []

    # Store the final string in stack
    for c in num:
        while (len(mystack) > 0 and k > 0
               and mystack[-1] > c):
            mystack.pop()
            k -= 1

        if (len(mystack) > 0 or c != '0'):
            mystack.append(c)

    # Now remove the largest values from the top of the
    # stack
    while (len(mystack) > 0 and k):
        k -= 1
        mystack.pop()
    if (len(mystack) == 0):
        return "0"

```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

**Got It !**

```

        num = num.replace(num[n - 1] , mystack[-1])
        mystack.pop()
        n -= 1

    return num[n:]

# driver code
Str = "765028321"
k = 5
print(removeKdigits(Str, k))

# This code is contributed by shinjanpatra

```

## Javascript

```

<script>

// JavaScript program for the above approach

function removeKdigits(num,k){

    let n = num.length
    let mystack = []
    // Store the final string in stack
    for(let c of num){
        while (mystack.length>0 && k > 0 && mystack[mystack.length - 1] > c){
            mystack.pop()
            k -= 1
        }

        if (mystack.length>0 || c != '0')
            mystack.push(c)
    }

    // Now remove the largest values from the top of the
    // stack
    while (mystack.length>0 && k){
        k -= 1
        mystack.pop()
    }

    return mystack.join('')
}

```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

**Got It !**



```

// Now retrieve the number from stack into a string
// (reusing num)
while (mystack.length>0){
    num = num.replace(num[n - 1] , mystack[mystack.length-1])
    mystack.pop()
    n -= 1
}

return num.substring(n,)
}

// driver code

let Str = "765028321"
let k = 5
document.write(removeKdigits(Str, k))

// code is contributed by shinjanpatra

</script>

```

## Output

221

***Time complexity:***  $O(N)$

***Space complexity:***  $O(N)$

This article is contributed by **Pallav Gurha**. Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

### AMAZON TEST SERIES

To Help Crack Your SDE Interview

[Enrol Now](#)



GeeksforGeeks

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

**Got It !**

Liked 30

Next

Calculate the angle between  
hour hand and minute hand

## RECOMMENDED ARTICLES

Page : 1 2 3

- 01

**Reduce the fraction to its lowest form**  
03, Apr 19
- 02

**Generate a sequence from first X natural numbers which adds up to S on raising 2 to the power of their lowest set bits**  
03, Jan 21
- 03

**Python program to print words from a sentence with highest and lowest ASCII value of characters**  
15, Mar 21
- 04

**Lowest Common Ancestor in a Binary Search Tree.**  
09, Aug 09
- 05

**Lowest Common Ancestor in a Binary Tree | Set 1**  
22, Feb 14
- 06

**Minimum digits to be removed to make either all digits or alternating digits same**  
15, Aug 20
- 07

**Numbers of Length N having digits A and B and whose sum of digits contain only digits A and B**  
31, Jan 19
- 08

**Find smallest number with given number of digits and sum of digits under given constraints**  
26, Jan 20

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

Got It !

## Article Contributed By :



GeeksforGeeks

## Vote for difficulty

Current difficulty : [Hard](#)

Easy

Normal

Medium

Hard

Expert

**Improved By :** [sanjeev2552](#), [Rajnis09](#), [gireeshgudaparthi](#),  
[UtkarshPandey6](#), [vaishaligoyal878](#), [sooda367](#),  
[arpitsrivastava467](#), [prasanna1995](#), [abhishek0719kadiyan](#),  
[shinjanpatra](#), [classroompxico](#), [hardikkoriintern](#)

**Article Tags :** [Amazon](#), [FactSet](#), [number-digits](#), [Mathematical](#), [Strings](#)

**Practice Tags :** [Amazon](#), [FactSet](#), [Strings](#), [Mathematical](#)

Improve Article

Report Issue

Writing code in comment? Please use [ide.geeksforgeeks.org](https://ide.geeksforgeeks.org), generate link and share the link here.

Load Comments

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

Got It !



A-143, 9th Floor, Sovereign Corporate Tower,  
Sector-136, Noida, Uttar Pradesh - 201305

[feedback@geeksforgeeks.org](mailto:feedback@geeksforgeeks.org)

## Company

[About Us](#)  
[Careers](#)  
[In Media](#)  
[Contact Us](#)  
[Privacy Policy](#)  
[Copyright Policy](#)

## Learn

[Algorithms](#)  
[Data Structures](#)  
[SDE Cheat Sheet](#)  
[Machine learning](#)  
[CS Subjects](#)  
[Video Tutorials](#)  
[Courses](#)

## News

[Top News](#)  
[Technology](#)  
[Work & Career](#)  
[Business](#)  
[Finance](#)  
[Lifestyle](#)  
[Knowledge](#)

## Languages

[Python](#)  
[Java](#)  
[CPP](#)  
[Golang](#)  
[C#](#)  
[SQL](#)  
[Kotlin](#)

## Web Development

## Contribute

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

**Got It !**

---

NodeJS

@geeksforgeeks , Some rights reserved

Do Not Sell My Personal Information

---

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

**Got It !**