

## 1) Write a C program to calculate sum of digits of a number.

The screenshot shows the Programiz Online Compiler interface. On the left, the code editor contains a C program named main.c. The code prompts the user to enter a number, checks if it's negative, and then uses a while loop to calculate the sum of its digits. On the right, the output window shows the result for the input 8976, which is 30. A success message "Code Execution Successful" is also displayed. The sidebar on the right features a banner for "Sugamya Bharat Abhiyan" and a "Programiz PRO" section.

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int number, sum = 0, digit;
5     printf("Enter a number: ");
6     scanf("%d", &number);
7     if (number < 0) {
8         printf("Please enter a positive number.\n");
9         return 1;
10    }
11    while (number > 0) {
12        digit = number % 10;
13        sum += digit;
14        number /= 10;
15    }
16    printf("The sum of the digits is: %d\n", sum);
17    return 0;
18}
19
```

Output  
Enter a number: 8976  
The sum of the digits is: 30  
== Code Execution Successful ==

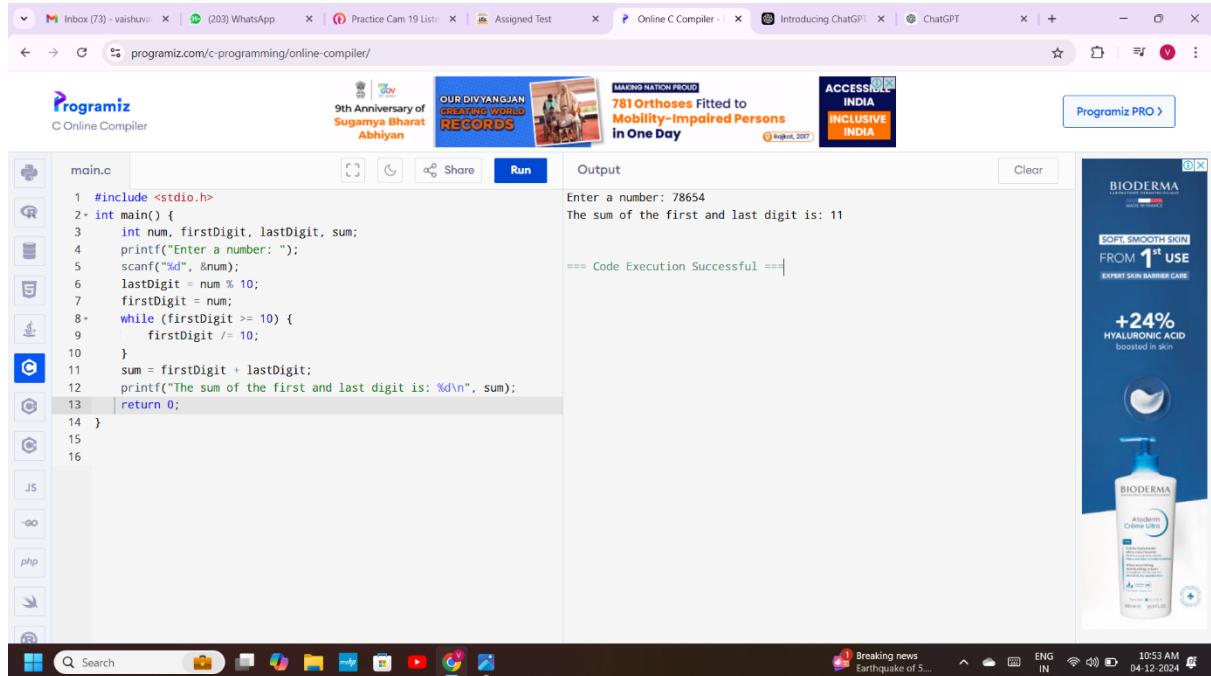
## 2) Write a C program to find first and last digit of a number.

The screenshot shows the Programiz Online Compiler interface. On the left, the code editor contains a C program named main.c. The code prompts the user to enter a number, handles negative numbers, and then uses a while loop to extract the first and last digits. On the right, the output window shows the result for the input 5768, where the first digit is 5 and the last digit is 8. A success message "Code Execution Successful" is also displayed. The sidebar on the right features a banner for "Sugamya Bharat Abhiyan" and a "Programiz PRO" section.

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int number, firstDigit, lastDigit;
5     printf("Enter a number: ");
6     scanf("%d", &number);
7     if (number < 0) {
8         number = -number;
8     }
9     lastDigit = number % 10;
10    firstDigit = number;
11    while (firstDigit >= 10) {
12        firstDigit /= 10;
13    }
14    printf("The first digit is: %d\n", firstDigit);
15    printf("The last digit is: %d\n", lastDigit);
16    return 0;
17}
18
19
```

Output  
Enter a number: 5768  
The first digit is: 5  
The last digit is: 8  
== Code Execution Successful ==

3) Write a C program to find sum of first and last digit of a number.

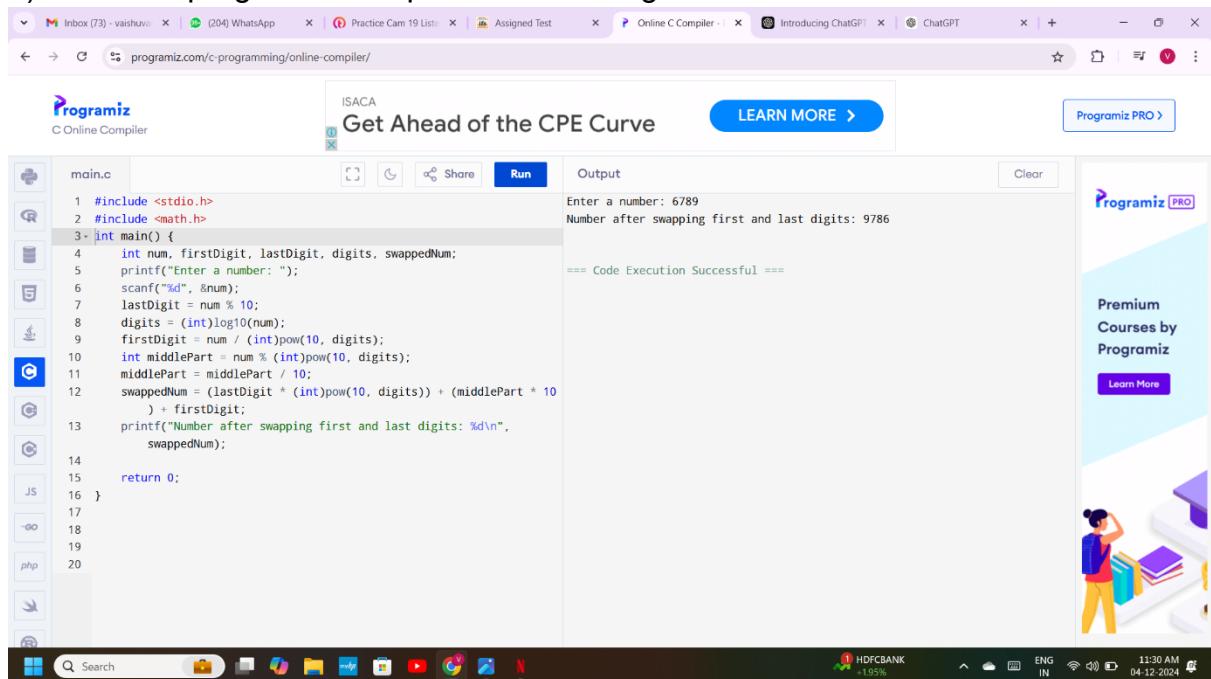


```
main.c
1 #include <stdio.h>
2 int main() {
3     int num, firstDigit, lastDigit, sum;
4     printf("Enter a number: ");
5     scanf("%d", &num);
6     lastDigit = num % 10;
7     firstDigit = num;
8     while (firstDigit >= 10) {
9         firstDigit /= 10;
10    }
11    sum = firstDigit + lastDigit;
12    printf("The sum of the first and last digit is: %d\n", sum);
13    return 0;
14 }
15
16
```

Output

```
Enter a number: 78654
The sum of the first and last digit is: 11
==== Code Execution Successful ====
```

4) Write a C program to swap first and last digits of a number



```
main.c
1 #include <stdio.h>
2 #include <math.h>
3 int main() {
4     int num, firstDigit, lastDigit, digits, swappedNum;
5     printf("Enter a number: ");
6     scanf("%d", &num);
7     lastDigit = num % 10;
8     digits = (int)log10(num);
9     firstDigit = num / (int)pow(10, digits);
10    int middlePart = num % (int)pow(10, digits);
11    middlePart = middlePart / 10;
12    swappedNum = (lastDigit * (int)pow(10, digits)) + (middlePart * 10
13        ) + firstDigit;
14    printf("Number after swapping first and last digits: %d\n",
15        swappedNum);
16 }
17
18
19
20
```

Output

```
Enter a number: 6789
Number after swapping first and last digits: 9786
==== Code Execution Successful ====
```

5) Write a C program to find frequency of each digit in a given integer.

```

main.c
9- while (number > 0) {
10-     int digit = number % 10;
11-     freq[digit]++;
12-     number /= 10;
13 }
14 }
15
16- int main() {
17     int number;
18     int freq[10];
19
20     printf("Enter an integer: ");
21     scanf("%d", &number);
22
23     findfrequency(number, freq);
24
25     printf("Digit frequencies:\n");
26-     for (int i = 0; i < 10; i++) {
27-         if (freq[i] > 0) {
28-             printf("Digit %d: %d times\n", i, freq[i]);
29-         }
30     }
31
32     return 0;
33 }
34

```

Output

```

Enter an integer: 653344456788
Digit frequencies:
Digit 0: 1 times
Digit 2: 1 times
Digit 4: 1 times
Digit 5: 1 times
Digit 6: 1 times
Digit 7: 2 times
Digit 9: 2 times

==== Code Execution Successful ====

```

## 6) Write a C program to enter a number and print it in words.

```

main.c
27     printf("Minus ");
28     number = -number;
29 }
30- while (number > 0) {
31     digits[count++] = number % 10;
32     number /= 10;
33 }
34- for (int i = count - 1; i >= 0; i--) {
35     printDigitInWords(digits[i]);
36 }
37
38     printf("\n");
39 }
40
41- int main() {
42     int number;
43
44     printf("Enter a number: ");
45     scanf("%d", &number);
46
47     printf("Number in words: ");
48     printNumberInWords(number);
49
50     return 0;
51 }
52

```

Output

```

Enter a number: 897
Number in words: Eight Nine Seven

```

## 7) Write a C program to find one's complement of a binary number.

Screenshot of a web browser showing a C programming online compiler on programiz.com. The code is for finding One's Complement of a binary number. The output shows an invalid binary number error.

```

main.c
1 #include <stdio.h>
2 #include <string.h>
3 void findOnesComplement(char binary[]) {
4     printf("One's Complement: ");
5     for (int i = 0; i < strlen(binary); i++) {
6         if (binary[i] == '0') {
7             printf("1");
8         } else if (binary[i] == '1') {
9             printf("0");
10        } else {
11            printf("\nInvalid binary number.\n");
12            return;
13        }
14    }
15    printf("\n");
16}
17 int main() {
18     char binary[100];
19     printf("Enter a binary number: ");
20     scanf("%s", binary);
21     findOnesComplement(binary);
22     return 0;
23 }

```

Output:

```

Enter a binary number: 457
One's Complement:
Invalid binary number.

== Code Execution Successful ==

```

## 8) Write a C program to find two's complement of a binary number

Screenshot of a web browser showing a C programming online compiler on programiz.com. The code is for finding Two's Complement of a binary number. The output shows an invalid binary number error.

```

main.c
1 #include <stdio.h>
2 #include <string.h>
3 void findOnesComplement(char binary[], char onesComplement[]) {
4     for (int i = 0; i < strlen(binary); i++) {
5         if (binary[i] == '0') {
6             onesComplement[i] = '1';
7         } else if (binary[i] == '1') {
8             onesComplement[i] = '0';
9         } else {
10            printf("\nInvalid binary number.\n");
11            return;
12        }
13    }
14    onesComplement[strlen(binary)] = '\0';
15}
16 void findTwosComplement(char binary[]) {
17     char onesComplement[100];
18     findOnesComplement(binary, onesComplement);
19     int carry = 1;
20     int n = strlen(onesComplement);
21     char twosComplement[100];
22     for (int i = n - 1; i >= 0; i--) {
23         if ((onesComplement[i] == '1' && carry == 1) ||
24             (onesComplement[i] == '0' && carry == 0)) {
25             twosComplement[i] = '0';
26         } else if ((onesComplement[i] == '0' && carry == 1) ||
27             (onesComplement[i] == '1' && carry == 0)) {
28             twosComplement[i] = '1';
29         }
30     }
31     twosComplement[strlen(binary)] = '\0';
32}
33 int main() {
34     char binary[100];
35     printf("Enter a binary number: ");
36     scanf("%s", binary);
37     findTwosComplement(binary);
38     printf("Two's Complement: %s", twosComplement);
39     return 0;
40 }

```

Output:

```

Enter a binary number: 897
Invalid binary number.
Two's Complement: @

== Code Execution Successful ==

```

## 9) Write a C program to convert Decimal to Hexadecimal number system

Screenshot of a web browser showing the Programiz online compiler interface. The code editor contains C code for decimal-to-hexadecimal conversion. The output window shows the program's execution results.

**Code Editor:**

```
main.c
1 #include <stdio.h>
2 void decimalToHexadecimal(int decimal) {
3     char hex[100];
4     int index = 0;
5     if (decimal == 0) {
6         printf("Hexadecimal: 0\n");
7         return;
8     }
9     int isNegative = 0;
10    if (decimal < 0) {
11        isNegative = 1;
12        decimal = -decimal;
13    }
14    while (decimal > 0) {
15        int remainder = decimal % 16;
16        if (remainder < 10) {
17            hex[index++] = 48 + remainder; // Convert to character
18            '0'-'9';
19        } else {
20            hex[index++] = 55 + remainder; // Convert to character
21            'A'-'F';
22        }
23        decimal /= 16;
24    }
25    if (isNegative) {
26        printf("-");
27    }
28    printf("Hexadecimal: ");
29    for (int i = index - 1; i >= 0; i--) {
30        printf("%c", hex[i]);
31    }
32    printf("\n");
33}
34 int main() {
35     int decimal;
36     printf("Enter a decimal number: ");
37     scanf("%d", &decimal);
38     decimalToHexadecimal(decimal);
39     return 0;
40 }
```

**Output Window:**

```
Enter a decimal number: 8.967
Hexadecimal: 8

== Code Execution Successful ==
```

**Right Sidebar:**

Premium Courses by Programiz

Programiz PRO



Screenshot of a web browser showing the Programiz online compiler interface. The code editor contains C code for decimal-to-hexadecimal conversion. The output window shows the program's execution results.

**Code Editor:**

```
main.c
1 #include <stdio.h>
2 void decimalToHexadecimal(int decimal) {
3     char hex[100];
4     int index = 0;
5     if (decimal == 0) {
6         printf("Hexadecimal: 0\n");
7         return;
8     }
9     int isNegative = 0;
10    if (decimal < 0) {
11        isNegative = 1;
12        decimal = -decimal;
13    }
14    while (decimal > 0) {
15        int remainder = decimal % 16;
16        if (remainder < 10) {
17            hex[index++] = 48 + remainder; // Convert to character
18            '0'-'9';
19        } else {
20            hex[index++] = 55 + remainder; // Convert to character
21            'A'-'F';
22        }
23        decimal /= 16;
24    }
25    if (isNegative) {
26        printf("-");
27    }
28    printf("Hexadecimal: ");
29    for (int i = index - 1; i >= 0; i--) {
30        printf("%c", hex[i]);
31    }
32    printf("\n");
33}
34 int main() {
35     int decimal;
36     printf("Enter a decimal number: ");
37     scanf("%d", &decimal);
38     decimalToHexadecimal(decimal);
39     return 0;
40 }
```

**Output Window:**

```
Enter a decimal number: 8.967
Hexadecimal: 8

== Code Execution Successful ==
```

**Right Sidebar:**

Premium Courses by Programiz

Programiz PRO



## Trapping rain water

The screenshot shows a browser window with multiple tabs open. The active tab is 'Trapping Rain Water' on leetcode.com. The code editor contains a Python3 solution for the problem. The code uses a two-pointer approach to calculate trapped water. It initializes left and right pointers at the start and end of the height array, respectively. It also initializes left\_max and right\_max to 0. A variable water\_trapped is used to store the total amount of trapped water. The algorithm iterates while left is less than right. Inside the loop, it checks if height[left] is less than height[right]. If true, it checks if height[left] is greater than or equal to left\_max. If true, it updates left\_max to height[left]. Otherwise, it adds left\_max - height[left] to water\_trapped and increments left by 1. If height[right] is greater than or equal to right\_max, it updates right\_max to height[right]. Otherwise, it adds right\_max - height[right] to water\_trapped and decrements right by 1. Finally, it returns water\_trapped. The code includes a main function call to trap with an example input [0,1,0,2,1,0,1,3,2,1,2,1].

```
1 class Solution:
2     def trap(self, height):
3         if not height or len(height) < 3:
4             return 0
5
6         left, right = 0, len(height) - 1
7         left_max, right_max = 0, 0
8         water_trapped = 0
9
10        while left < right:
11            if height[left] < height[right]:
12                if height[left] >= left_max:
13                    left_max = height[left]
14                else:
15                    water_trapped += left_max - height[left]
16                left += 1
17            else:
18                if height[right] >= right_max:
19                    right_max = height[right]
20                else:
21                    water_trapped += right_max - height[right]
22                right -= 1
23
24        return water_trapped
25
26 # Example for Testing
27 if __name__ == "__main__":
28     solution = Solution()
29     param_1 = [0,1,0,2,1,0,1,3,2,1,2,1] # Example input
30     print(solution.trap(param_1)) # Output: 6
31
```

Ln 1, Col 1 Saved

The screenshot shows a browser window with multiple tabs open. The active tab is 'Trapping Rain Water' on leetcode.com. The code editor contains the same Python3 solution as the previous screenshot. The status bar indicates 'Solved'. The test case section shows 'Accepted' with a runtime of 0 ms. The input is height = [0,1,0,2,1,0,1,3,2,1,2,1]. The output is 6. The explanation states that the elevation map is represented by the array [0,1,0,2,1,0,1,3,2,1,2,1]. In this case, 6 units of rain water are being trapped.

**Example 1:**

**Input:** height = [0,1,0,2,1,0,1,3,2,1,2,1]  
**Output:** 6  
**Explanation:** The above elevation map (black section) is represented by array [0,1,0,2,1,0,1,3,2,1,2,1]. In this case, 6 units of rain water (blue section) are being trapped.

**Example 2:**

**Input:** height = [4,2,0,3,2,5]

33.1K 326 433 Online

23°C Mostly sunny ENG IN 10:43 AM 08-01-2025

## Flatten binary tree to linked list

leetcode.com/problems/flatten-binary-tree-to-linked-list/

Problem List | Premium

Code

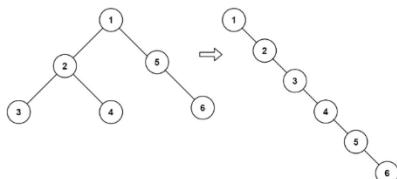
Python3 Auto

```
1 class TreeNode:
2     def __init__(self, val=0, left=None, right=None):
3         self.val = val
4         self.left = left
5         self.right = right
6
7
8 class Solution:
9     def flatten(self, root):
10         if not root:
11             return
12
13         # Flatten the left and right subtrees
14         self.flatten(root.left)
15         self.flatten(root.right)
16
17         # Store the original right subtree
18         temp_right = root.right
19
20         # Move the flattened left subtree to the right
21         root.right = root.left
22         root.left = None
23
24         # Attach the original right subtree to the end of the new right subtree
25         current = root
26         while current.right:
27             current = current.right
28         current.right = temp_right
29
```

In 1, Col 1 Saved

Run Submit

Example 1:



Input: root = [1,2,5,3,4,null,6]  
Output: [1,null,2,null,3,null,4,null,5,null,6]

Testcase > Test Result

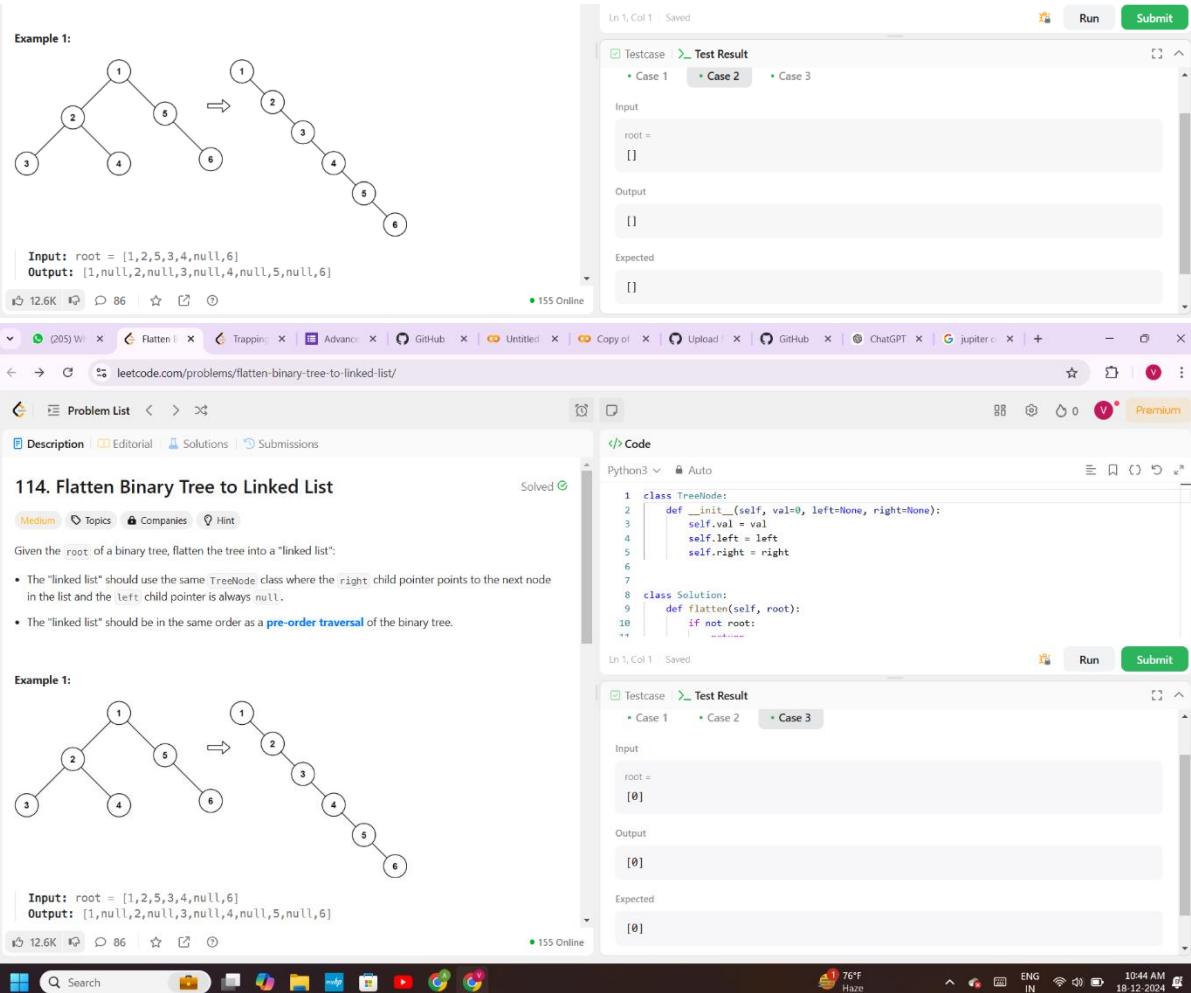
Case 1 Case 2 Case 3

Input  
root = [1,2,5,3,4,null,6]

Output  
[1,null,2,null,3,null,4,null,5,null,6]

Expected  
[1,null,2,null,3,null,4,null,5,null,6]

12.6K 86 155 Online



## Largest elements in an array

Problem List < > ✎

Code Python3 v Auto

```

1 import random
2
3 class Solution:
4     def findKthLargest(self, nums, k):
5         def quickselect(left, right, k_smallest):
6             if left == right:
7                 return nums[left]
8
9             pivot_index = random.randint(left, right)
10            pivot_index = partition(left, right, pivot_index)
11
12            if k_smallest == pivot_index:
13                return nums[k_smallest]
14            elif k_smallest < pivot_index:
15                return quickselect(left, pivot_index - 1, k_smallest)
16            else:
17                return quickselect(pivot_index + 1, right, k_smallest)
18
19        def partition(left, right, pivot_index):
20            pivot = nums[pivot_index]
21            nums[pivot_index], nums[right] = nums[right], nums[pivot_index]
22            store_index = left
23
24            for i in range(left, right):
25                if nums[i] < pivot:
26                    nums[i], nums[store_index] = nums[store_index], nums[i]
27                    store_index += 1
28
29            nums[store_index], nums[right] = nums[right], nums[store_index]
30            return store_index
31
32        return quickselect(0, len(nums) - 1, len(nums) - k)
33
34
35

```

Ln 33, Col 1 Saved

Run Submit

Problem List < > ✎

Description | Editorial | Solutions | Submissions

## 215. Kth Largest Element in an Array

Medium Topics Companies

Given an integer array `nums`, and an integer `k`, return the `kth` largest element in the array.

Note that it is the `kth` largest element in the sorted order, not the `kth` distinct element.

Can you solve it without sorting?

**Example 1:**

Input: `nums = [3,2,1,5,6,4]`, `k = 2`  
Output: 5

**Example 2:**

Input: `nums = [3,2,3,1,2,4,5,5,6]`, `k = 4`  
Output: 4

**Constraints:**

- $1 \leq k \leq \text{nums.length} \leq 10^5$
- $-10^4 \leq \text{nums}[i] \leq 10^4$

Eco Retreat Hirakud Open

Colaba Tourism

Seen this question in a real interview before? 1/5

17.5K 263 192 Online

Testcase > Test Result

Case 1 Case 2

Input

```
nums =
[3,2,1,5,6,4]
```

k = 2

Output

5

Expected

5

**215. Kth Largest Element in an Array**

Given an integer array `nums` and an integer `k`, return the `kth` largest element in the array.

Note that it is the `kth` largest element in the sorted order, not the `kth` distinct element.

Can you solve it without sorting?

**Example 1:**

```
Input: nums = [3,2,1,5,6,4], k = 2
Output: 5
```

**Example 2:**

```
Input: nums = [3,2,3,1,2,4,5,5,6], k = 4
Output: 4
```

**Constraints:**

- $1 \leq k \leq \text{nums.length} \leq 10^5$
- $-10^4 \leq \text{nums}[i] \leq 10^4$

Odisha Tourism  
Eco Retreat Hirakud

Seen this question in a real interview before? 1/5

17.5K 263 193 Online

Python3

```
import random

class Solution:
    def findKthLargest(self, nums, k):
        def quickselect(left, right, k_smallest):
            if left == right:
                return nums[left]
            pivot_index = random.randint(left, right)
            pivot_index = partition(left, right, pivot_index)

            if k_smallest == pivot_index:
                return nums[k_smallest]
            elif k_smallest < pivot_index:
                return quickselect(left, pivot_index - 1, k_smallest)
            else:
                return quickselect(pivot_index + 1, right, k_smallest)

        return quickselect(0, len(nums) - 1, k)
```

Ln 33, Col 1 Saved

Testcase Test Result

Case 1 Case 2

Input

```
nums =
[3,2,3,1,2,4,5,5,6]
```

k =

Output

4

Expected

4

Run Submit

## Merge k sorted lists

The screenshot shows the LeetCode platform. At the top, there's a navigation bar with various tabs like WhatsApp, Merge k Sort, Trapping Rain, Stamping The, Maximum Subarray, G-Learn, Introducing C++, ChatGPT, and YouTube. Below the navigation bar is the URL leetcode.com/problems/merge-k-sorted-lists/. The main area has a title "Problem List" and a "Code" section. The code is written in Python3 and uses a heap-based approach to merge k sorted linked lists. It defines a ListNode class and a Solution class with a mergeKLists method. The code is saved and ready to run. Below the code editor is a browser window showing the same page, with a "Testcase" tab open. The test case input is [[1,4,5],[1,3,4],[2,6]] and the output is [1,1,2,3,4,4,5,6]. The status is "Accepted" with a runtime of 0 ms. The browser also shows system information like 25°C Haze, ENG IN, and 11:00 AM 08-01-2025.

```

</> Code
Python3 v Auto
1 import heapq
2
3 class ListNode:
4     def __init__(self, val=0, next=None):
5         self.val = val
6         self.next = next
7
8 class Solution:
9     def mergeKLists(self, lists):
10        # Define a heap
11        heap = []
12
13        # Initialize the heap with the head of each linked list
14        for i, lst in enumerate(lists):
15            if lst:
16                heapq.heappush(heap, (lst.val, i, lst))
17
18        # Dummy node to build the result
19        dummy = ListNode(0)
20        current = dummy
21
22        # Process the heap
23        while heap:
24            val, i, node = heapq.heappop(heap)
25            current.next = node
26            current = current.next
27            if node.next:
28                heapq.heappush(heap, (node.next.val, i, node.next))
29
30        # Return the merged list
31        return dummy.next
32
33 Ln 1, Col 1 Saved
34 Run Submit

```

**Description | Editorial | Solutions | Submissions**

### 23. Merge k Sorted Lists

**Hard** **Topics** **Companies**

You are given an array of  $k$  linked-lists `lists`, each linked-list is sorted in ascending order. Merge all the linked-lists into one sorted linked-list and return it.

**Example 1:**

```

Input: lists = [[1,4,5],[1,3,4],[2,6]]
Output: [1,1,2,3,4,4,5,6]
Explanation: The linked-lists are:
[
    1->4->5,
    1->3->4,
    2->6
]
merging them into one sorted list:
1->1->2->3->4->4->5->6

```

**Example 2:**

```

Input: lists = []
Output: []

```

**Example 3:**

19.9K 226

262 Online

25°C Haze

ENG IN

11:10 AM 08-01-2025

## Design circular deque

The screenshot shows a browser window with multiple tabs open at [leetcode.com/problems/design-circular-deque/](https://leetcode.com/problems/design-circular-deque/). The main content area displays two Python3 code snippets for a circular deque implementation.

**Code Snippet 1 (Insert Operations):**

```
1 class MyCircularDeque:
2     def __init__(self, k: int):
3         self.k = k # Maximum size of the deque
4         self.deque = [0] * k # Initialize deque with fixed size
5         self.front = -1 # Front pointer
6         self.rear = -1 # Rear pointer
7         self.size = 0 # Current size of deque
8
9     def insertFront(self, value: int) -> bool:
10        if self.isFull():
11            return False
12        if self.isEmpty():
13            self.front = 0
14            self.rear = 0
15        else:
16            self.front = (self.front - 1 + self.k) % self.k
17            self.deque[self.front] = value
18        self.size += 1
19        return True
20
21     def insertLast(self, value: int) -> bool:
22        if self.isFull():
23            return False
24        if self.isEmpty():
25            self.front = 0
26            self.rear = 0
27        else:
28            self.rear = (self.rear + 1) % self.k
29            self.deque[self.rear] = value
30        self.size += 1
31
32
33 Ln 69, Col 35 Saved
```

**Code Snippet 2 (Delete Operations):**

```
44     def deleteLast(self) -> bool:
45        if self.isEmpty():
46            return False
47        if self.front == self.rear:
48            self.front = -1
49            self.rear = -1
50        else:
51            self.rear = (self.rear - 1 + self.k) % self.k
52        self.size -= 1
53        return True
54
55     def getFront(self) -> int:
56        if self.isEmpty():
57            return -1
58        return self.deque[self.front]
59
60     def getRear(self) -> int:
61        if self.isEmpty():
62            return -1
63        return self.deque[self.rear]
64
65     def isEmpty(self) -> bool:
66        return self.size == 0
67
68     def isFull(self) -> bool:
69        return self.size == self.k
69
69 Ln 69, Col 35 Saved
```

**Example 1:**

**Input**  
["MyCircularDeque", "insertLast", "insertLast", "insertFront", "insertFront", "getRear", "isFull", "deleteLast", "insertFront", "getFront"]  
[[3], [1], [2], [3], [4], [], [], [], [4], []]  
**Output**  
[null, true, true, true, false, 2, true, true, true, 4]

**Explanation**  
MyCircularDeque myCircularDeque = new MyCircularDeque(3);  
myCircularDeque.insertLast(1); // return True  
myCircularDeque.insertLast(2); // return True  
myCircularDeque.insertFront(3); // return True  
myCircularDeque.insertFront(4); // return False, the queue is full.  
myCircularDeque.getRear(); // return 2  
myCircularDeque.isFull(); // return True  
myCircularDeque.deleteLast(); // return True  
myCircularDeque.insertFront(4); // return True  
myCircularDeque.getFront(); // return 4

**Constraints:**

- $1 \leq k \leq 1000$
- $0 \leq \text{value} \leq 1000$
- At most  $1000$  calls will be made to `insertFront`, `insertLast`, `deleteFront`, `deleteLast`, `getFront`, and `getRear`.

Code:

```
def __init__(self, k: int):
    self.queue = [None] * k
    self.front = -1
    self.rear = -1
    self.size = 0

def insertFront(self, value: int) -> bool:
    if self.isFull():
        return False
    if self.front == -1:
        self.front = 0
    else:
        self.front -= 1
    self.queue[self.front] = value
    self.size += 1
    return True

def insertLast(self, value: int) -> bool:
    if self.isFull():
        return False
    if self.rear == -1:
        self.rear = 0
    else:
        self.rear += 1
    self.queue[self.rear] = value
    self.size += 1
    return True

def deleteFront(self) -> int:
    if self.isEmpty():
        return -1
    value = self.queue[self.front]
    if self.front == self.rear:
        self.front = -1
        self.rear = -1
    else:
        self.front += 1
    self.size -= 1
    return value

def deleteLast(self) -> int:
    if self.isEmpty():
        return -1
    value = self.queue[self.rear]
    if self.front == self.rear:
        self.front = -1
        self.rear = -1
    else:
        self.rear -= 1
    self.size -= 1
    return value

def getFront(self) -> int:
    if self.isEmpty():
        return -1
    return self.queue[self.front]

def getRear(self) -> int:
    if self.isEmpty():
        return -1
    return self.queue[self.rear]

def isEmpty(self) -> bool:
    return self.size == 0

def isFull(self) -> bool:
    return self.size == len(self.queue)
```

Testcase: **Accepted** Runtime: 0 ms

Case 1

Input: ["MyCircularDeque", "insertLast", "insertLast", "insertFront", "insertFront", "getRear", "isFull", "deleteLast", "insertFront", "getFront"]  
[[3], [1], [2], [3], [4], [], [], [], [4], []]

Output: [null, true, true, true, false, 2, true, true, true, 4]

## Maximum sum circular subarray

Screenshot of a browser window showing a LeetCode problem: Maximum Sum Circular Subarray.

The code implements a solution using Kadane's algorithm twice. It first finds the maximum sum of a subarray in the original array. Then it finds the minimum sum of a subarray in the inverted array (arr[i] = -arr[i]). The total sum of the array is then subtracted by the minimum sum of the inverted array to get the maximum sum of a circular subarray.

```

1 class Solution:
2     def maxSubarraySumCircular(self, nums):
3         def kadane(arr):
4             max_sum = arr[0]
5             current_sum = arr[0]
6             for i in range(1, len(arr)):
7                 current_sum = max(arr[i], current_sum + arr[i])
8                 max_sum = max(max_sum, current_sum)
9             return max_sum
10
11         total_sum = sum(nums)
12         max_kadane = kadane(nums)
13         inverted_nums = [-x for x in nums]
14         min_kadane = kadane(inverted_nums)
15         max_circular = total_sum + min_kadane
16
17         if max_circular == 0:
18             return max_kadane
19
20         return max(max_kadane, max_circular)
21
22 # Example call using the Solution class
23 param_1 = [5, -3, 5]
24 result = Solution().maxSubarraySumCircular(param_1)
25 print(result) # Output: 10
26
27
28

```

Ln 20, Col 45 Saved

Run Submit

Description Editorial Solutions Submissions Testcase Test Result Accepted Runtime: 0 ms Case 1 Case 2 Case 3 Input nums = [1,-2,3,-2] Stdout 10 Output

Accepted 306.2K Submissions 659.5K Acceptance Rate 46.4% 6.8K 79 123 Online

## Stamping the sequence

The screenshot shows a browser window with multiple tabs open. The active tab is for the LeetCode problem 'Stamping The Sequence'. The code editor contains the following Python solution:

```
1  class Solution:
2      def movesToStamp(self, stamp, target):
3          n, m = len(target), len(stamp)
4          target = list(target)
5          stamp = list(stamp)
6          result = []
7          replaced = [False] * n
8
9          # Helper function to check if we can stamp at index i
10         def canStamp(i):
11             stamped = False
12             for j in range(m):
13                 if target[i + j] != '?' and target[i + j] != stamp[j]:
14                     return False
15                 if target[i + j] == stamp[j]:
16                     stamped = True
17             return stamped
18
19         # Helper function to stamp at index i
20         def doStamp(i):
21             for j in range(m):
22                 target[i + j] = '?'
23             result.append(i)
24
25         # Try to stamp from right to left
26         changed = True
27         while changed:
28             changed = False
29             for i in range(n - m + 1):
30                 if canStamp(i) and not replaced[i]:
31                     ...
32
33
34
35
36
37
38
39
40
```

The code implements a backtracking algorithm to find the minimum number of stamp moves required to transform the target sequence into the stamp sequence. It uses helper functions to check if a stamp can be applied at a given index and to actually apply the stamp. The main loop iterates from right to left, applying stamps where possible and marking indices as replaced.

leetcode.com/problems/stamping-the-sequence/

Description | Editorial | Solutions | Submissions

Return an array of the index of the left-most letter being stamped at each turn. If we cannot obtain target from s within  $10 * \text{target.length}$  turns, return an empty array.

**Example 1:**

```
Input: stamp = "abc", target = "ababc"
Output: [0,2]
Explanation: Initially s = "??????".
- Place stamp at index 0 to get "abc??".
- Place stamp at index 2 to get "ababc".
[1,0,2] would also be accepted as an answer, as well as some other answers.
```

**Example 2:**

```
Input: stamp = "abca", target = "aabcaca"
Output: [3,0,1]
Explanation: Initially s = "????????".
- Place stamp at index 3 to get "????bca".
- Place stamp at index 0 to get "abca".
- Place stamp at index 1 to get "aabcaca".
```

**Constraints:**

- $1 \leq \text{stamp.length} \leq \text{target.length} \leq 1000$
- stamp and target consist of lowercase English letters.

1.6K | 13 | ⭐ | 🎁 | 29 Online | Run | Submit | Testcase | Test Result | Accepted | Runtime: 0 ms | Case 1 | Case 2 | Input | stamp = "abc" | target = "ababc" | Output | Nifty midcap -1.14% | ENG IN | 10:31 AM | 08-01-2025

```
Python3
# Python3
# Helper function to stamp at index i
def doStamp(i):
    for j in range(m):
        if target[i + j] == '?':
            result.append(i)
            break
    else:
        stamped = True
        for j in range(m):
            if target[i + j] != stamp[j]:
                stamped = False
                break
        if stamped:
            return stamped
        else:
            result.append(i)
            break
    return stamped

# Main function
def stampingTheSequence(stamp, target):
    m = len(stamp)
    n = len(target)
    result = []
    for i in range(n - m + 1):
        if doStamp(i):
            result.append(i)
    return result
```