

LAB PRACTICALS (JAN-JUN, 2025)

S. No.	WEEK No.	NAME OF PRACTICAL	Date	Signature
1.	WEEK 1			
2.	WEEK 2			
3.	WEEK 3			
4.	WEEK 4			
5.	WEEK 5			
6.	WEEK 6			
7.	WEEK 7			
8.	WEEK 8			
9.	WEEK 9			
10.	WEEK 10			
11.	WEEK 11			
12.	WEEK 12			
13.	WEEK 13			
14.	WEEK 14			

LAB 01

TASK 01:

BUBBLE SORT

CODE:

```
#include <bits/stdc++.h>
using namespace std;
void bubblesort(int arr[] , int n){
    int count=0;
    for(int i=0;i<n-1;i++){
        bool swapped = false;

        for(int j=0;j<n-i-1;j++){
            if(arr[j] > arr[j+1]){
                swap(arr[j],arr[j+1]);
                swapped = true;
            }
            count++;
        }
        if(!swapped){break;}
    }
    cout<<count<<endl;
}
void print(int arr[] ,int n){
    for(int i=0;i<n;i++){
        cout<<arr[i]<<" ";
    }
    cout<<endl;
}
int main(){
    int arr[7] = {7,6,5,4,3,2,1};
    int n = 7;
    print(arr,n);
    bubblesort(arr,n);
    print(arr,n);
}
```

OUTPUT:

```
PS C:\Users\Ritik gupta\Desktop\Lab\DAA\Lab 1> cd "c:\Users\Ritik gupta"
}
7 6 5 4 3 2 1
21
1 2 3 4 5 6 7
PS C:\Users\Ritik gupta\Desktop\Lab\DAA\Lab 1> █
```

SELECTION SORT:

```
#include <bits/stdc++.h>
using namespace std;
void selectionsort(int arr[] , int n){
    int count =0;
    for(int i =0;i<n-1;i++){
        int min_idx = i;
        for(int j = i+1;j<n;j++){
            if(arr[j] < arr[min_idx]){
                min_idx = j;
            }
        }
        count++;
        swap(arr[min_idx],arr[i]);
    }
    cout<<count<<endl;
}
void print(int arr[] ,int n){
    for(int i=0;i<n;i++){
        cout<<arr[i]<<" ";
    }
    cout<<endl;
}
int main(){
    int arr[5] = {3,8,2,5,1};
    int n = 5;
    print(arr,n);
    selectionsort(arr,n);
    print(arr,n);
}
```

OUTPUT:

```
PS C:\Users\Ritik gupta\Desktop\Lab\DAA\Lab 1> cd "c:\Users\Ritik gupta\Desktop\
"
3 8 2 5 1
10
1 2 3 5 8
PS C:\Users\Ritik gupta\Desktop\Lab\DAA\Lab 1> █
```

Time Complexities:

	Bubble sort	Selection sort
Best case	$n.(n-1)/2$	$n.(n-1)/2$
Worst case	$n-1$	$n.(n-1)/2$

TASK-02:

1.Add two numbers

Description

Editorial

Solutions

Submissions

2. Add Two Numbers

Medium

Topics

Companies

You are given two **non-empty** linked lists representing two non-negative integers. The digits are stored in **reverse order**, and each of their nodes contains a single digit. Add the two numbers and return the sum as a linked list.

You may assume the two numbers do not contain any leading zero, except the number 0 itself.

Example 1:

2 → 4 → 3

5 → 6 → 4

Solved

✓

</> Code

C++

Auto

```
12 public:
13     ListNode* addTwoNumbers(ListNode* l1, ListNode* l2) {
14         // int sum = l1->val+l2->val;
15         int c=0;
16         int a,b;
17         ListNode* ans=new ListNode(0);
18         ListNode* head = ans;
19         while(l1!=NULL || l2!=NULL || c!=0){
20
21             int a = (l1 != nullptr) ? l1->val : 0;
22             int b = (l2 != nullptr) ? l2->val : 0;
23
24             int sum =a+b+c;
25             ListNode* temp=new ListNode(sum%10);
26             c = sum/10;
27             ans->next =temp;
28
29             if(l1!=NULL){
30                 l1 = l1->next; }
31             if(l2!=NULL){
```

Saved

Testcase

Test Result

</> Code

Accepted

1569 / 1569 testcases passed

RITIK GUPTA submitted at Jan 26, 2025 15:35

Editorial

Solution

Runtime

0 ms

Beats 100.00%

Memory

76.98 MB

Beats 91.65%

Analyze Complexity

75%

50%

25%

0%

1ms

2ms

3ms

4ms

5ms

6ms

7ms

Testcase

Test Result

Source

2.Reverse integer

Description

Editorial

Solutions

Submissions

7. Reverse Integer

Medium

Topics

Companies

Given a signed 32-bit integer x , return x with its digits reversed. If reversing x causes the value to go outside the signed 32-bit integer range $[-2^{31}, 2^{31} - 1]$, then return 0.

Assume the environment does not allow you to store 64-bit integers (signed or unsigned).

Example 1:

Input: $x = 123$
Output: 321

Example 2:

Input: $x = -123$
Output: -321

Example 3:

Input: $x = 120$
Output: 21

Solved

✓

</> Code

C++

Auto

```
1 class solution {
2 public:
3     int reverse(int x) {
4         int temp = x;
5         int num=0;
6         for(int i=0;temp!=0;i++){
7             int digit = temp%10;
8             if(num>INT_MAX/10 || num<INT_MIN/10){
9                 return 0;
10            }
11            num = digit + num*10;
12            temp /= 10;
13        }
14        return num;
15    }
16};
```

Saved

Testcase

Test Result

</> Code

Accepted

1045 / 1045 testcases passed

RITIK GUPTA submitted at Jan 22, 2025 10:57

Editorial

Solution

Runtime

5 ms

Beats 19.42%

Memory

8.66 MB

Beats 20.62%

Analyze Complexity

75%

50%

25%

0%

1ms

2ms

3ms

4ms

5ms

6ms

7ms

Testcase

Test Result

Source

3.Dutch flag

75. Sort Colors

Given an array `nums` with n objects colored red, white, or blue, sort them **in-place** so that objects of the same color are adjacent, with the colors in the order red, white, and blue.

We will use the integers 0, 1, and 2 to represent the color red, white, and blue, respectively.

You must solve this problem without using the library's sort function.

Example 1:

Input: `nums = [2,0,2,1,1,0]`
Output: `[0,0,1,1,2,2]`

Example 2:

Input: `nums = [2,0,1]`
Output: `[0,1,2]`

</> Code

C++

Auto

```
1 class Solution {
2 public:
3     void sortColors(vector<int>& nums) {
4         int left=0;
5         int right=nums.size()-1;
6         int i=0;
7         while(i<=right){
8             if(nums[i] == 0){
9                 swap(nums[i],nums[left]);
10                i++;
11                left++;
12            }
13            else if(nums[i] == 2){
14                swap(nums[i],nums[right]);
15                right--;
16            }
17            else{
18                i++;
19            }
20        }
21    }
```

Saved

Testcase

Test Result

</> Code

Accepted

88 / 88 testcases passed

RITIK GUPTA submitted at Jan 22, 2025 18:52

Editorial

Solution

Runtime

0 ms

Beats 100.00%

Memory

11.65 MB

Beats 31.54%

Analyze Complexity

100%

100%

50%

0%

1ms

2ms

3ms

4ms

5ms

6ms

7ms

Testcase

Test Result

Source