LAB 2

Task 1

Quick Sorting Algorithm

```
#include <iostream>
#include <vector>
using namespace std;

int partition(vector<int>& arr,int low , int high,int& count,int& loop){
    int pivot = arr[high];
    int i=low-1;

    for (int j=low ; j <= high-1;j++){
        if(arr[j]<pivot){
            i++;
            swap(arr[i],arr[j]);
            count ++;
        }
        loop++;
    }

    swap(arr[i+1],arr[high]);
    return i+1;
}

void quicksort(vector<int>& arr,int low , int high,int& count,int& loop){
    if (low<high){
        int pi = partition(arr,low,high,count,loop);
        quicksort(arr,low,pi-1,count,loop);
        quicksort(arr,pi+1,high,count,loop);
    }
}

int main (){
    vector<int> arr = {10, 7, 8, 9, 1, 5, 2, 6, 3, 4};
    int n = arr.size();
    int count = 0;
    int loop = 0;
    quicksort(arr,0,n-1,count,loop);
    cout<<"Number of Swaps: "<<count<<endl;
    cout<<"Number of loops: "<<loop<<endl;
    cout<<"Sorted array: ";
    for (int i=0;i<n;i++){
        cout<<arr[i]<<" ";
    }

```

```
PS C:\Users\mohit\Desktop\STUDY\DAA LAB\30 JAN> cd "c:\U
Number of Swaps: 45
Number of loops: 45
Sorted array: 1 2 3 4 5 6 7 8 9 10
PS C:\Users\mohit\Desktop\STUDY\DAA LAB\30 JAN> cd "c:\U
Number of Swaps: 20
Number of loops: 45
Sorted array: 1 2 3 4 5 6 7 8 9 10
PS C:\Users\mohit\Desktop\STUDY\DAA LAB\30 JAN> cd "c:\U
Number of Swaps: 14
Number of loops: 22
Sorted array: 1 2 3 4 5 6 7 8 9 10
PS C:\Users\mohit\Desktop\STUDY\DAA LAB\30 JAN>
```

	LOOPS	SWAPS
BEST	45	45
WORSE	45	20
AVG	22	14

PIVOT ELEMENT AS FIRST

```cpp
#include <iostream>
#include <vector>
using namespace std;

int partition(vector<int>& arr, int low, int high, int& count, int& loop) {
    int pivot = arr[low];
    int i = low + 1;

    for (int j = low + 1; j <= high; j++) {
        if (arr[j] < pivot) {
            swap(arr[i], arr[j]);
            i++;
            count++;
        }
        loop++;
    }
    cout<<pivot<<endl;
    swap(arr[low], arr[i - 1]);
    return i - 1;
}

void quicksort(vector<int>& arr, int low, int high, int& count, int& loop) {
    if (low < high) {
        int pi = partition(arr, low, high, count, loop);
        quicksort(arr, low, pi - 1, count, loop);
        quicksort(arr, pi + 1, high, count, loop);
    }
}

int main() {
    vector<int> arr = {10,40,80,70,50};
    int n = arr.size();
    int count = 0;
    int loop = 0;
    quicksort(arr, 0, n - 1, count, loop);

    cout << "Number of Swaps: " << count << endl;
    cout << "Number of loops: " << loop << endl;
    cout << "Sorted array: ";
    for (int i = 0; i < n; i++) {
        cout << arr[i] << " ";
    }
}

```

```
PS C:\Users\mohit\Desktop\STUDY\DAA LAB\30 JAN> cd "c:
pivot element 10
pivot element 40
pivot element 80
pivot element 50
Number of Swaps: 2
Number of loops: 10
Sorted array: 10 40 50 70 80
PS C:\Users\mohit\Desktop\STUDY\DAA LAB\30 JAN>
```

	LOOPS	SWAPS
BEST	45	45
WORSE	45	20
AVG	22	14

TASK 2

Decimal to Binary (Iterative)

```
#include <iostream>
using namespace std;

int main() {
    int decimal, binary = 0, place = 1;

    cout << "Enter a decimal number: ";
    cin >> decimal;

    int originalDecimal = decimal;

    while (decimal > 0) {
        int remainder = decimal % 2;
        binary += remainder * place;
        place *= 10;
        decimal /= 2;
    }

    cout << "The binary of " << originalDecimal << " is: " << binary << endl;

    return 0;
}

```

```
PS C:\Users\mohit\Desktop\STUDY\DAA LAB\30 JAN>
Enter a decimal number: 25
The binary of 25 is: 11001
PS C:\Users\mohit\Desktop\STUDY\DAA LAB\30 JAN>
```

Decimal to Binary Recursive

```
#include <iostream>
using namespace std;

void decimalToBinary(int decimal) {
    if (decimal == 0) {
        return;
    }
    decimalToBinary(decimal / 2);
    cout << decimal % 2;
}

int main() {
    int decimal;
    cout << "Enter a decimal number: ";
    cin >> decimal;

    if (decimal == 0) {
        cout << "The binary of 0 is: 0" << endl;
    } else {
        cout << "The binary of " << decimal << " is: ";
        decimalToBinary(decimal);
        cout << endl;
    }

    return 0;
```

LEETCODE PROBLEM

Given an m x n 2D binary grid grid which represents a map of '1's (land) and '0's (water), return *the number of islands*.

An island is surrounded by water and is formed by connecting adjacent lands horizontally or v class Solution {

```
 1.  public:
 2.   int numIslands(vector<vector<char>>& grid) {
 3.     constexpr int dirs[4][2] = {{0, 1}, {1, 0}, {0, -1}, {-1, 0}};
 4.     const int m = grid.size();
 5.     const int n = grid[0].size();
 6.     int ans = 0;
 7.
 8.     auto bfs = [&](int r, int c) {
 9.       queue<pair<int, int>> q{{{r, c}}};
10.       grid[r][c] = '2';  // Mark '2' as visited.
11.       while (!q.empty()) {
12.         const auto [i, j] = q.front();
13.         q.pop();
14.         for (const auto& [dx, dy] : dirs) {
15.           const int x = i + dx;
16.           const int y = j + dy;
17.           if (x < 0 || x == m || y < 0 || y == n)
18.             continue;
19.           if (grid[x][y] != '1')
20.             continue;
21.           q.emplace(x, y);
22.           grid[x][y] = '2';  // Mark '2' as visited.
23.         }
24.       }
25.     };
26.
27.     for (int i = 0; i < m; ++i)
28.       for (int j = 0; j < n; ++j)
29.         if (grid[i][j] == '1') {
30.           bfs(i, j);
31.           ++ans;
32.         }
33.
34.     return ans;
35.   }
36. };
37.
38.
39
40.
```