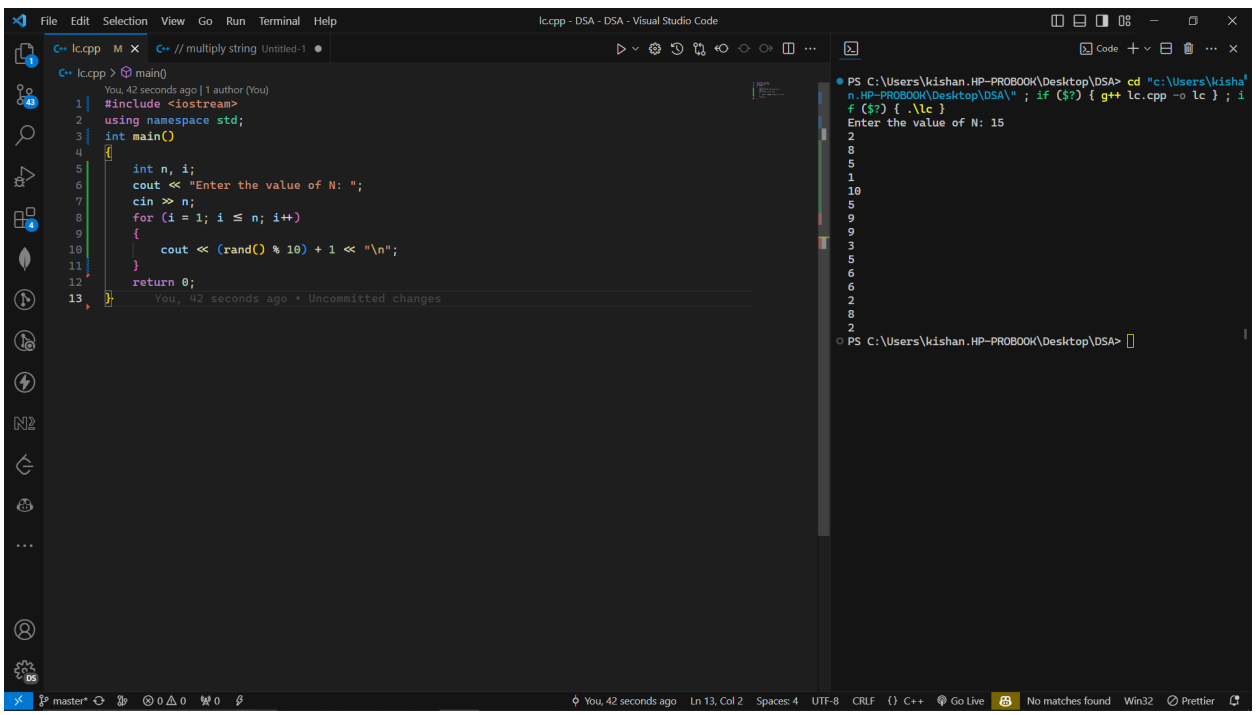
Kishan Vaghamashi
202312014

1)

```cpp
#include <iostream>
using namespace std;
int main()
{
    int n, i;
    cout << "Enter the value of N: ";
    cin >> n;
    for (i = 1; i <= n; i++)
    {
        cout << (rand() % 10) + 1 << "\n";
    }
    return 0;
}
```



2)

```cpp
#include <iostream>
#include <ctime>
#include <cstdlib>
using namespace std;

void insertionSort(int arr[], int n)
{
```

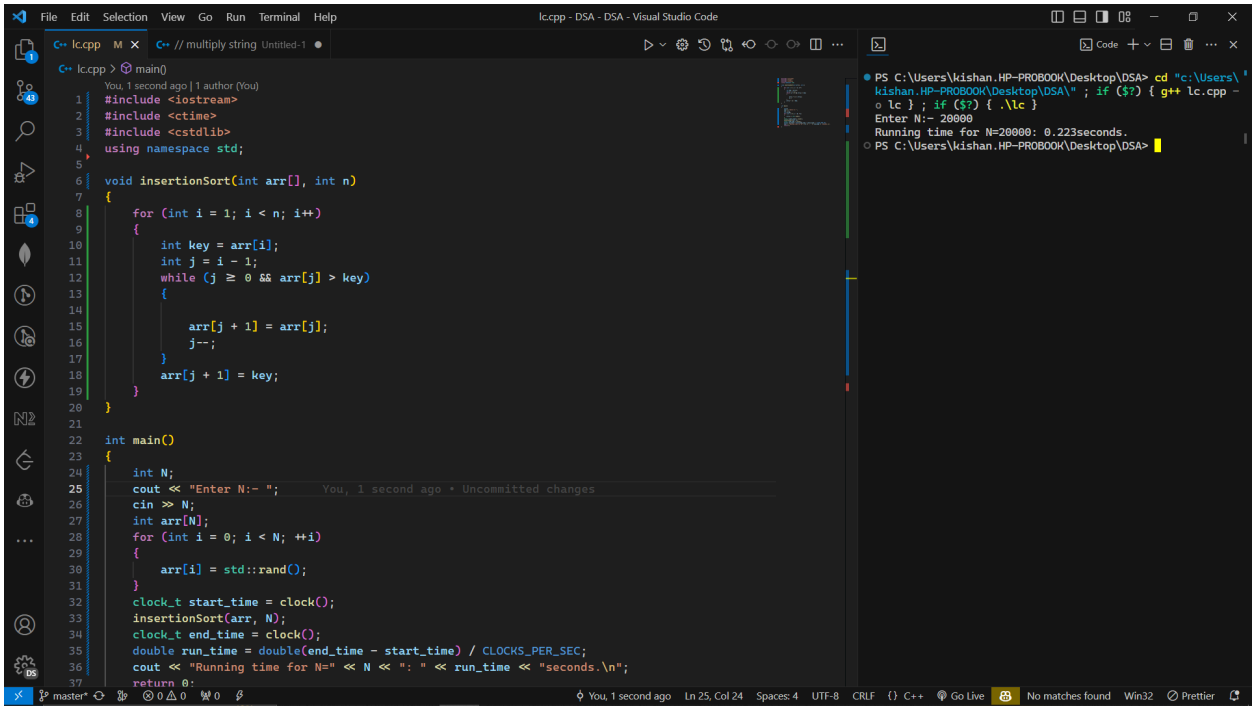
Kishan Vaghamashi
202312014

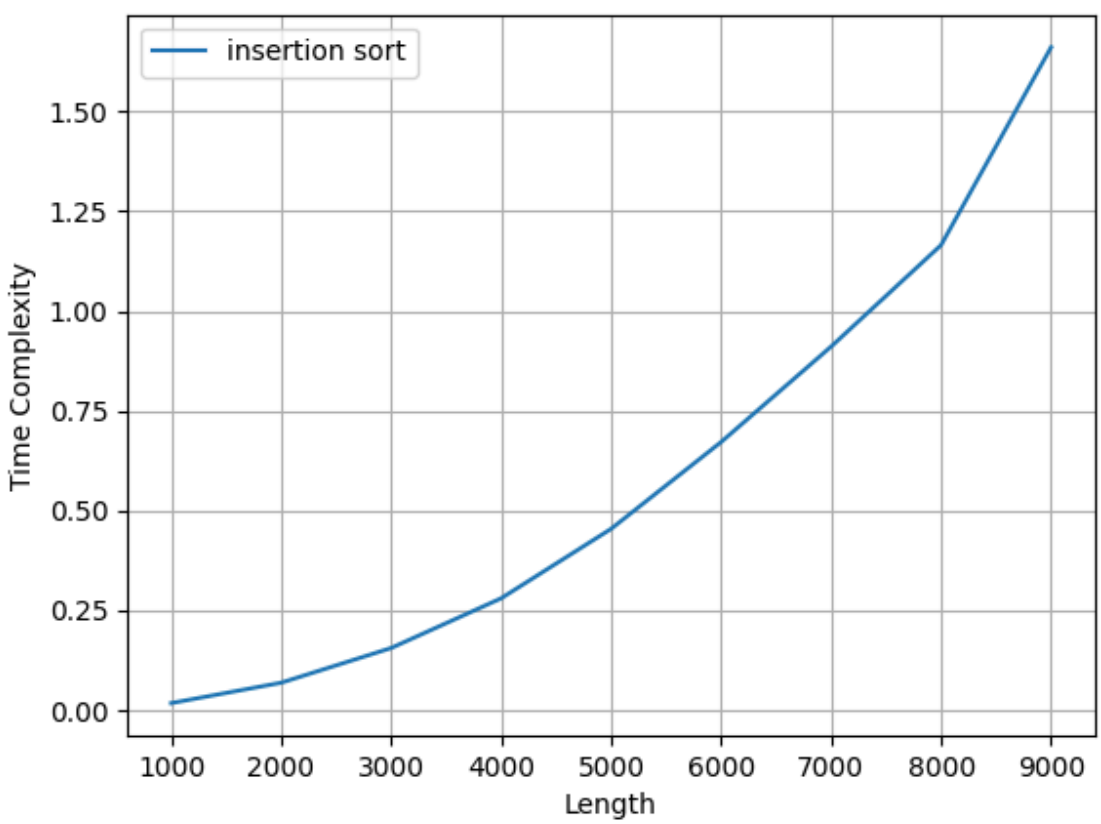
```
    for (int i = 1; i < n; i++)
    {
        int key = arr[i];
        int j = i - 1;
        while (j >= 0 && arr[j] > key)
        {

            arr[j + 1] = arr[j];
            j--;
        }
        arr[j + 1] = key;
    }
}

int main()
{
    int N;
    cout << "Enter N:- ";
    cin >> N;
    int arr[N];
    for (int i = 0; i < N; ++i)
    {
        arr[i] = std::rand();
    }
    clock_t start_time = clock();
    insertionSort(arr, N);
    clock_t end_time = clock();
    double run_time = double(end_time - start_time) / CLOCKS_PER_SEC;
    cout << "Running time for N=" << N << ": " << run_time <<
"seconds.\n";
    return 0;
}
```

Kishan Vaghamashi
202312014





Kishan Vaghamashi
202312014

3)

```
import time
import random
import matplotlib.pyplot as plt
random.seed(202312058)

def merge(arr, l, m, r):
    n1 = m - l + 1
    n2 = r- m
    L = [0] * (n1)
    R = [0] * (n2)
    for i in range(0 , n1):
        L[i] = arr[l + i]
    for j in range(0 , n2):
        R[j] = arr[m + 1 + j]
    i = 0
    j = 0
    k = l
    while i < n1 and j < n2 :
    if L[i] <= R[j]:
        arr[k] = L[i]
        i += 1
    else:
        arr[k] = R[j]
        j += 1
    k += 1

    while i < n1:
    arr[k] = L[i]
    i += 1
    k += 1

    while j < n2:
    arr[k] = R[j]
    j += 1
    k += 1

def merge_sort(arr, l , r):
    if l < r:
        m = l + (r-l)//2
```

Kishan Vaghamashi
202312014

```
    merge_sort(arr, l, m)
    merge_sort(arr, m+1, r)
    merge(arr, l, m, r)

def generate_numbers_file(n, file_name="data/numbers.txt"):
    f = open(file_name, "w")
    for i in range(1, n+1):
        r_num = random.randrange(1, n+1)
        f.write(str(r_num) + "\n")
    f.close()

def read_numbers_file(file_name="data/numbers/txt"):
    data = []
    f = open("data/numbers.txt", "r")
    for line in f:
        data.append(int(line))
    f.close()
    return data

def calculate_time(fn, arr):
    start = time.time()
    n = len(arr)
    fn(arr, 0, n-1)
    end = time.time()
    return end - start

x = [i*1000 for i in range(1, 100)]
y = []
for n in x:
    generate_numbers_file(n)
    arr = read_numbers_file()
    y.append(calculate_time(merge_sort, arr))

def plot_graph():
    plt.xlabel('Length')
    plt.ylabel('Time Complexity')
    plt.plot(x, y, label = "merge sort")
    plt.grid()
    plt.legend()
    plt.show()
```

Kishan Vaghamashi
202312014

```
plot_graph()
```