Date: 05/05/2021

Practical 9:

2CSDE75 - Advanced Data Structures

Name: Shrey Viradiya

Roll No: 18BCE259

Aim:

Suffix arrays are pre-processed structures that can be used to solve the classical substring matching problem. Implement suffix arrays for a long string sequence and demonstrate the matching operation.

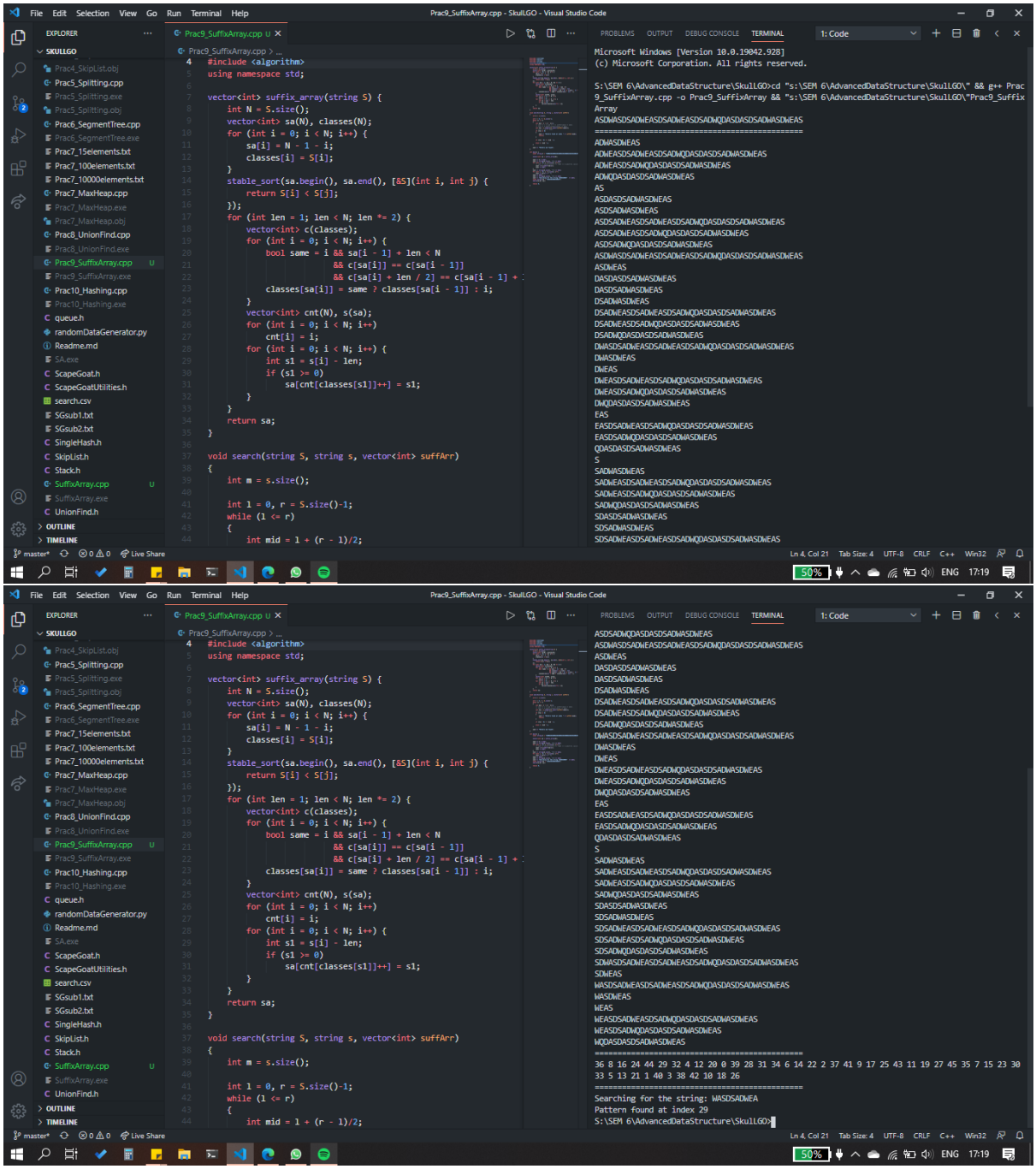
Code:

Prac9_SuffixArray.cpp

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

vector<int> suffix_array(string S) {
    int N = S.size();
    vector<int> sa(N), classes(N);
    for (int i = 0; i < N; i++) {
        sa[i] = N - 1 - i;
        classes[i] = S[i];
    }
    stable_sort(sa.begin(), sa.end(), [&S](int i, int j) {
        return S[i] < S[j];
    });
    for (int len = 1; len < N; len *= 2) {
        vector<int> c(classes);
        for (int i = 0; i < N; i++) {
            bool same = i && sa[i - 1] + len < N
                          && c[sa[i]] == c[sa[i - 1]]
                          && c[sa[i] + len / 2] == c[sa[i - 1] + len / 2];
            classes[sa[i]] = same ? classes[sa[i - 1]] : i;
        }
        vector<int> cnt(N), s(sa);
        for (int i = 0; i < N; i++)
            cnt[i] = i;
        for (int i = 0; i < N; i++) {
            int s1 = s[i] - len;
            if (s1 >= 0)
                sa[cnt[classes[s1]]++] = s1;
        }
    }
    return sa;
}

void search(string S, string s, vector<int> suffArr)
{
    int m = s.size();

    int l = 0, r = S.size()-1;
    while (l <= r)
    {
        int mid = l + (r - l)/2;
        // cout << l << mid << r << suffArr[mid] << endl;
```

```
        // cout << S.substr(suffArr[mid]) << endl;
        int res = s.compare(S.substr(suffArr[mid]));
        // cout << res << endl;
        if (res == 0)
        {
            cout << "Pattern found at index " << suffArr[mid];
            return;
        }

        if (res < 0) r = mid - 1;

        else l = mid + 1;
    }

    cout << "Pattern not found";
}

int main() {
    const string S = "ASDWASDSADWEASDSADWEASDSADWQDASDASDSADWASDWEAS";

    vector<int> sa = suffix_array(S);

    cout << S << endl;
    cout << string(S.size(), '=') << endl;
    for (int i = 0; i < S.size(); i++) {
        // cout << setw(S.size()) << right << S.substr(0, sa[i]) << '|';
        cout << S.substr(sa[i]);
        cout << endl;
    }
    cout << string(S.size(), '=') << endl;
    for (int i = 0; i < S.size(); i++)
        cout << sa[i] << " ";
    cout << endl;
    cout << string(S.size(), '=') << endl;
    cout << "Searching for the string: WASDSADWEA"  << endl;
    const string S2 = "ASDASDSADWASDWEAS";
    search(S,S2, sa);

    return 0;
}
```

Snapshot of the output:



Conclusion:

Suffix arrays are widely used and largely interchangeable index structures on strings and sequences.