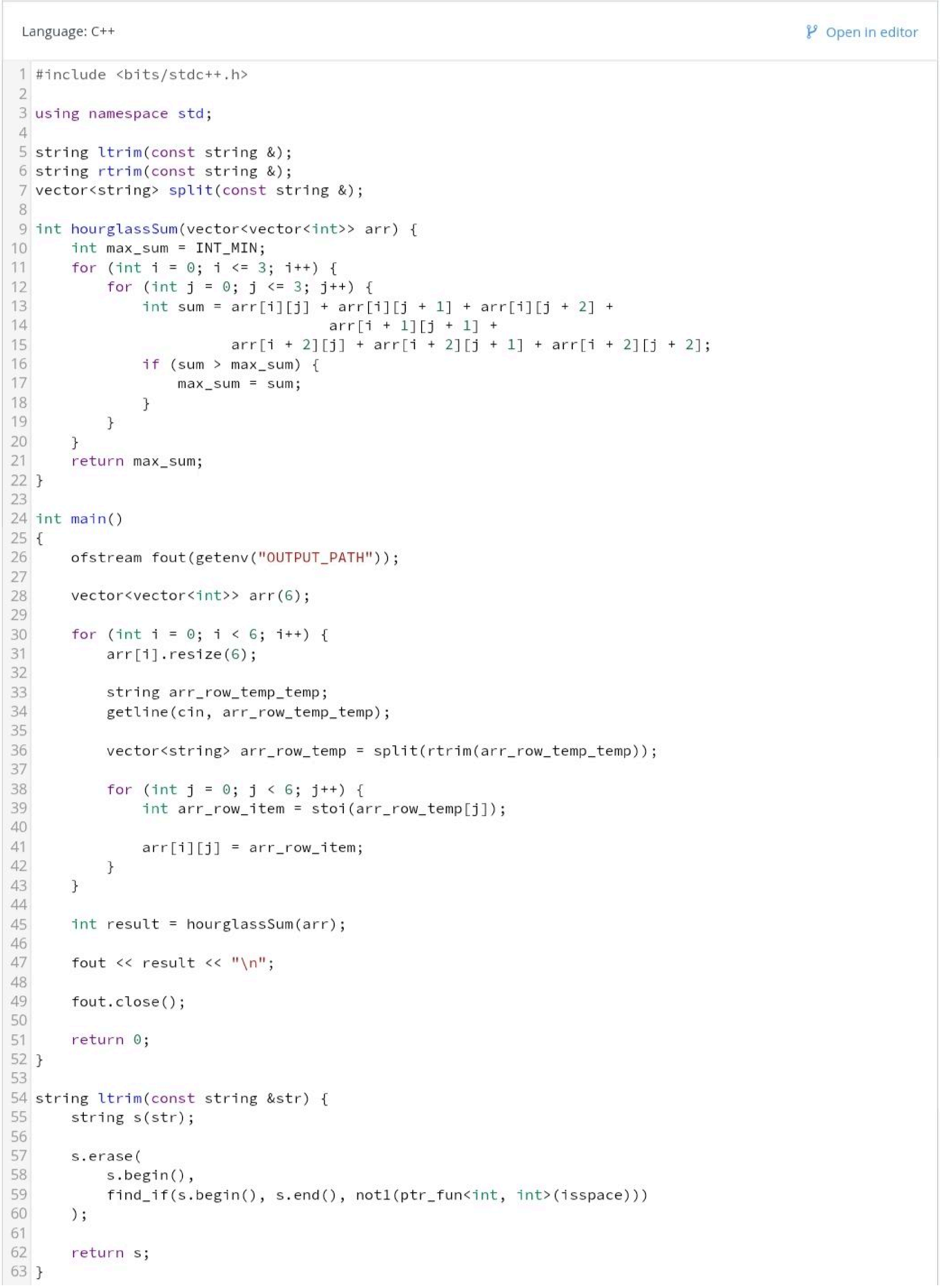
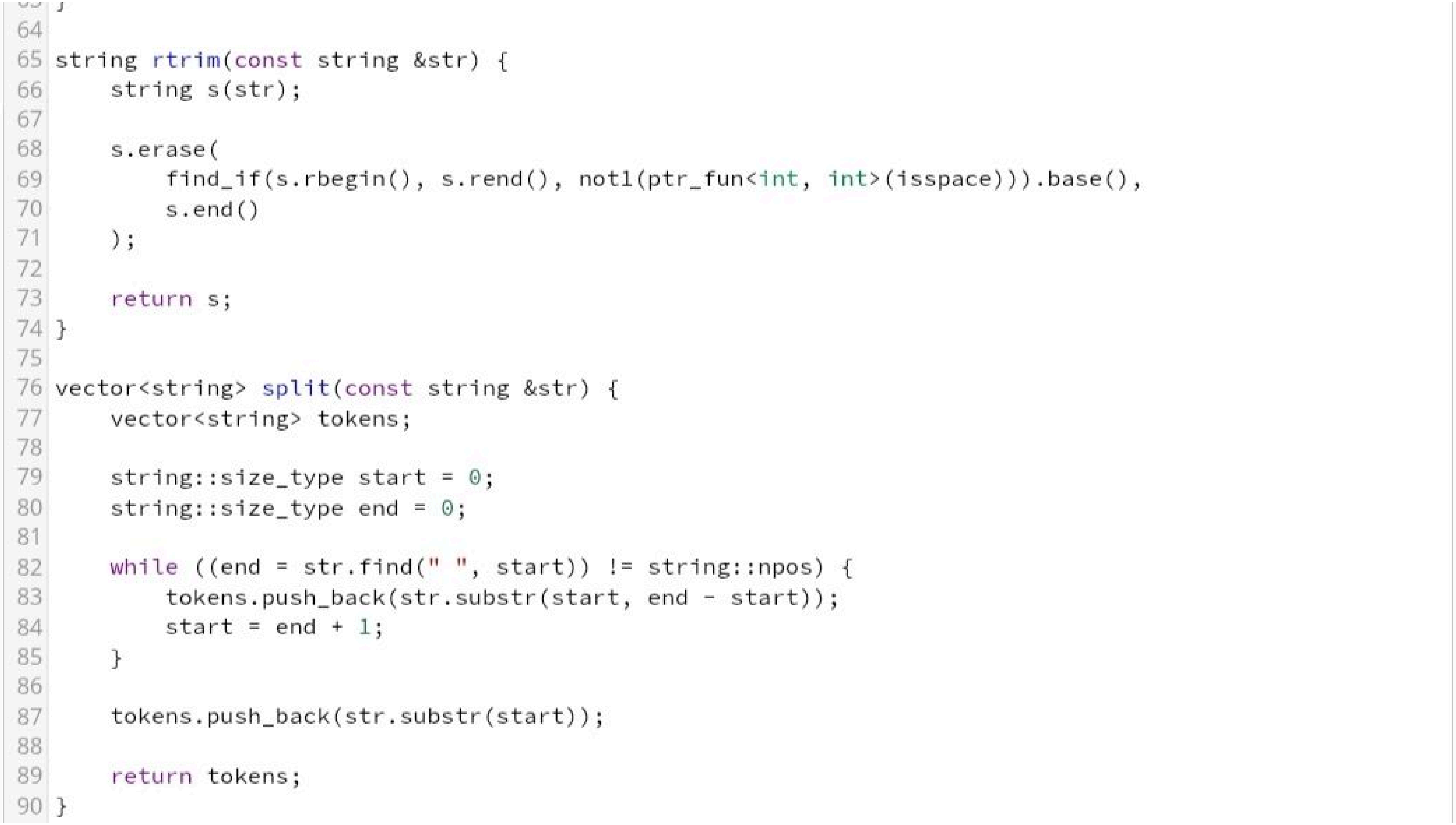


```cpp
#include <bits/stdc++.h>

using namespace std;

string ltrim(const string &);
string rtrim(const string &);
vector<string> split(const string &);

int hourglassSum(vector<vector<int>> arr) {
    int max_sum = INT_MIN;
    for (int i = 0; i <= 3; i++) {
        for (int j = 0; j <= 3; j++) {
            int sum = arr[i][j] + arr[i][j + 1] + arr[i][j + 2] +
                                 arr[i + 1][j + 1] +
                      arr[i + 2][j] + arr[i + 2][j + 1] + arr[i + 2][j + 2];
            if (sum > max_sum) {
                max_sum = sum;
            }
        }
    }
    return max_sum;
}

int main()
{
    ofstream fout(getenv("OUTPUT_PATH"));

    vector<vector<int>> arr(6);

    for (int i = 0; i < 6; i++) {
        arr[i].resize(6);

        string arr_row_temp_temp;
        getline(cin, arr_row_temp_temp);

        vector<string> arr_row_temp = split(rtrim(arr_row_temp_temp));

        for (int j = 0; j < 6; j++) {
            int arr_row_item = stoi(arr_row_temp[j]);

            arr[i][j] = arr_row_item;
        }
    }

    int result = hourglassSum(arr);

    fout << result << "\n";

    fout.close();

    return 0;
}

string ltrim(const string &str) {
    string s(str);

    s.erase(
        s.begin(),
        find_if(s.begin(), s.end(), not1(ptr_fun<int, int>(isspace)))
    );

    return s;
}
```



```
}

string rtrim(const string &str) {
    string s(str);

    s.erase(
        find_if(s.rbegin(), s.rend(), not1(ptr_fun<int, int>(isspace))).base(),
        s.end()
    );

    return s;
}

vector<string> split(const string &str) {
    vector<string> tokens;

    string::size_type start = 0;
    string::size_type end = 0;

    while ((end = str.find(" ", start)) != string::npos) {
        tokens.push_back(str.substr(start, end - start));
        start = end + 1;
    }

    tokens.push_back(str.substr(start));

    return tokens;
}
```