LAPORAN TUGAS KECIL 1

IF2211 Strategi Algoritma

Penyelesaian *Word Search Puzzle* dengan Algoritma *Brute Force*

Disusun oleh:

Nama : Johannes Winson Sukiatmodjo

NIM : 13520123

PROGRAM STUDI TEKNIK INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

BANDUNG

2022

Algoritma *Brute Force*

Berdasarkan program yang telah saya buat, langkah-langkah pencarian solusinya adalah sebagai berikut:

1. Program akan mengecek setiap kata sampai ketemu. Jika kata sudah ketemu ke arah tertentu, proses pencarian dengan arah lain tidak akan dilakukan karena looping akan langsung berhenti dan akan mengecek kata-kata berikutnya.

2. Program akan mengecek setiap kata dengan mencocokkannya pada setiap karakter pada *puzzle* dengan urutan arah yang akan dicek terlebih dahulu adalah vertikal ke atas, vertikal ke bawah, horizontal ke kanan, horizontal ke kiri, diagonal ke kanan atas, diagonal ke kanan bawah, diagonal ke kiri atas, dan diagonal ke kiri bawah.

3. Sebelum masuk ke dalam proses pencocokan karakter, program akan mengecek terlebih dahalu apakah panjang kata tersebut melebihi batas *puzzle*. Jika panjang kata tersebut melebihi batas *puzzle*, pencocokan karakter melalui arah tersebut tidak akan dilakukan dan dilanjutkan dengan pencocokan ke arah yang lain.

Source Program

```
#include <iostream>
#include <fstream>
#include <chrono>

using namespace std;

int main() {
    string txt;
    int baris, kolom, keywords, check, i, j, k, l, m, a, b, c;
    int checkall = 0;
    bool found;
    cout << "Masukkan nama file : ";
    cin >> txt;
    cout << "Masukkan jumlah baris : ";
    cin >> baris;
    cout << "Masukkan jumlah kolom : ";
    cin >> kolom;
    cout << "Masukkan jumlah kata kunci : ";
    cin >> keywords;
    char character[baris][kolom];
    string word[keywords];
    string line;
    ifstream file;
    file.open("../test/" + txt);
    for (i = 0; i < baris; i++) {
        for (j = 0; j < kolom; j++) {
            file >> character[i][j];
        }
    }
    int idx = 0;
    while (getline(file, line)) {
        file >> word[idx];
        idx++;
    }
    file.close();
    auto started = std::chrono::high_resolution_clock::now();
    cout <<
"================================================================================
===" << endl;
    for (i = 0; i < keywords; i++) {
        check = 0;
        found = false;
        for (j = 0; j < baris; j++) {
            for (k = 0; k < kolom; k++) {
```

```
                a = j;
                b = k;
                c = 0;
                if (a - word[i].length() + 1 >= 0) { // vertikal ke atas
                    while (character[a][b] == word[i][c]) {
                        a--;
                        c++;
                        check++;
                        checkall++;
                        if (word[i].length() == c) {
                            found = true;
                            a = j - word[i].length() + 1;
                            b = k;
                            c = 0;
                            for (l = 0; l < baris; l++) {
                                for (m = 0; m < kolom; m++) {
                                    if ((l == a) && (m == b) && (c <
word[i].length())) {
                                        cout << character[l][m] << ' ';
                                        a++;
                                        c++;
                                    } else {
                                        cout << '-' << ' ';
                                    }
                                }
                                cout << endl;
                            }
                            break;
                        }
                    }
                    if (found) {
                        break;
                    }
                    check += 1;
                    checkall += 1;
                    a = j;
                    b = k;
                    c = 0;
                }
                if (a + word[i].length() - 1 < baris) { // vertikal ke bawah
                    while (character[a][b] == word[i][c]) {
                        a++;
                        c++;
                        check++;
                        checkall++;
                        if (word[i].length() == c) {
                            found = true;
                            a = j;
```

```
                            b = k;
                            c = 0;
                            for (l = 0; l < baris; l++) {
                                for (m = 0; m < kolom; m++) {
                                    if ((l == a) && (m == b) && (c <
word[i].length())) {
                                        cout << character[l][m] << ' ';
                                        a++;
                                        c++;
                                    } else {
                                        cout << '-' << ' ';
                                    }
                                }
                                cout << endl;
                            }
                            break;
                        }
                    }
                    if (found) {
                        break;
                    }
                    check += 1;
                    checkall += 1;
                    a = j;
                    b = k;
                    c = 0;
                }
                if (b + word[i].length() - 1 < kolom) { // horizontal ke kanan
                    while (character[a][b] == word[i][c]) {
                        b++;
                        c++;
                        check++;
                        checkall++;
                        if (word[i].length() == c) {
                            found = true;
                            a = j;
                            b = k;
                            c = 0;
                            for (l = 0; l < baris; l++) {
                                for (m = 0; m < kolom; m++) {
                                    if ((l == a) && (m == b) && (c <
word[i].length())) {
                                        cout << character[l][m] << ' ';
                                        b++;
                                        c++;
                                    } else {
                                        cout << '-' << ' ';
                                    }
```

```
                            }
                            cout << endl;
                        }
                        break;
                    }
                }
                if (found) {
                    break;
                }
                check += 1;
                checkall += 1;
                a = j;
                b = k;
                c = 0;
            }
            if (b - word[i].length() + 1 >= 0) { // horizontal ke kiri
                while (character[a][b] == word[i][c]) {
                    b--;
                    c++;
                    check++;
                    checkall++;
                    if (word[i].length() == c) {
                        found = true;
                        a = j;
                        b = k - word[i].length() + 1;
                        c = 0;
                        for (l = 0; l < baris; l++) {
                            for (m = 0; m < kolom; m++) {
                                if ((l == a) && (m == b) && (c <
word[i].length())) {
                                    cout << character[l][m] << ' ';
                                    b++;
                                    c++;
                                } else {
                                    cout << '-' << ' ';
                                }
                            }
                            cout << endl;
                        }
                        break;
                    }
                }
                if (found) {
                    break;
                }
                check += 1;
                checkall += 1;
                a = j;
```

```
                    b = k;
                    c = 0;
                }
                if ((a - word[i].length() + 1 >= 0) && (b + word[i].length() -
1 < kolom)) { // diagonal ke kanan atas
                    while (character[a][b] == word[i][c]) {
                        a--;
                        b++;
                        c++;
                        check++;
                        checkall++;
                        if (word[i].length() == c) {
                            found = true;
                            a = j - word[i].length() + 1;
                            b = k + word[i].length() - 1;
                            c = 0;
                            for (l = 0; l < baris; l++) {
                                for (m = 0; m < kolom; m++) {
                                    if ((l == a) && (m == b) && (c <
word[i].length())) {
                                        cout << character[l][m] << ' ';
                                        a++;
                                        b--;
                                        c++;
                                    } else {
                                        cout << '-' << ' ';
                                    }
                                }
                                cout << endl;
                            }
                            break;
                        }
                    }
                    if (found) {
                        break;
                    }
                    check += 1;
                    checkall += 1;
                    a = j;
                    b = k;
                    c = 0;
                }
                if ((a + word[i].length() - 1 < baris) && (b +
word[i].length() - 1 < kolom)) { // diagonal ke kanan bawah
                    while (character[a][b] == word[i][c]) {
                        a++;
                        b++;
                        c++;
```

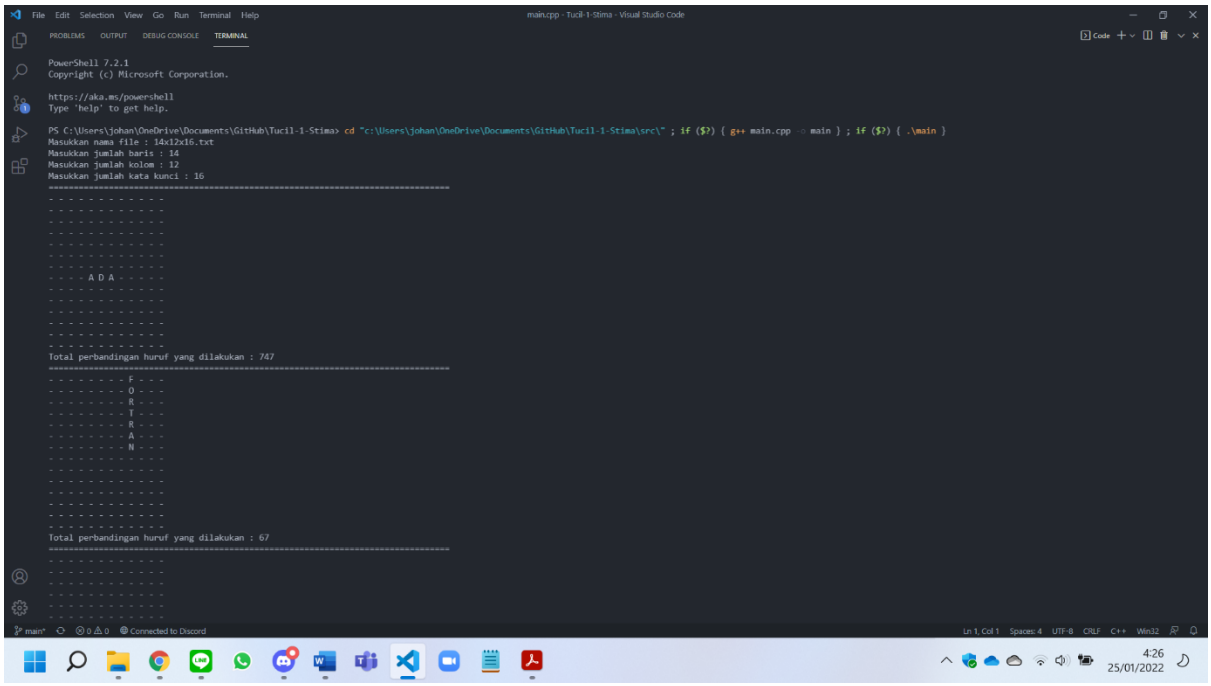
```
                        check++;
                        checkall++;
                        if (word[i].length() == c) {
                            found = true;
                            a = j;
                            b = k;
                            c = 0;
                            for (l = 0; l < baris; l++) {
                                for (m = 0; m < kolom; m++) {
                                    if ((l == a) && (m == b) && (c <
word[i].length())) {
                                        cout << character[l][m] << ' ';
                                        a++;
                                        b++;
                                        c++;
                                    } else {
                                        cout << '-' << ' ';
                                    }
                                }
                                cout << endl;
                            }
                            break;
                        }
                    }
                    if (found) {
                        break;
                    }
                    check += 1;
                    checkall += 1;
                    a = j;
                    b = k;
                    c = 0;
                }
                if ((a - word[i].length() + 1 >= 0) && (b - word[i].length() +
1 >= 0)) { // diagonal ke kiri atas
                    while (character[a][b] == word[i][c]) {
                        a--;
                        b--;
                        c++;
                        check++;
                        checkall++;
                        if (word[i].length() == c) {
                            found = true;
                            a = j - word[i].length() + 1;
                            b = k - word[i].length() + 1;
                            c = 0;
                            for (l = 0; l < baris; l++) {
                                for (m = 0; m < kolom; m++) {
```

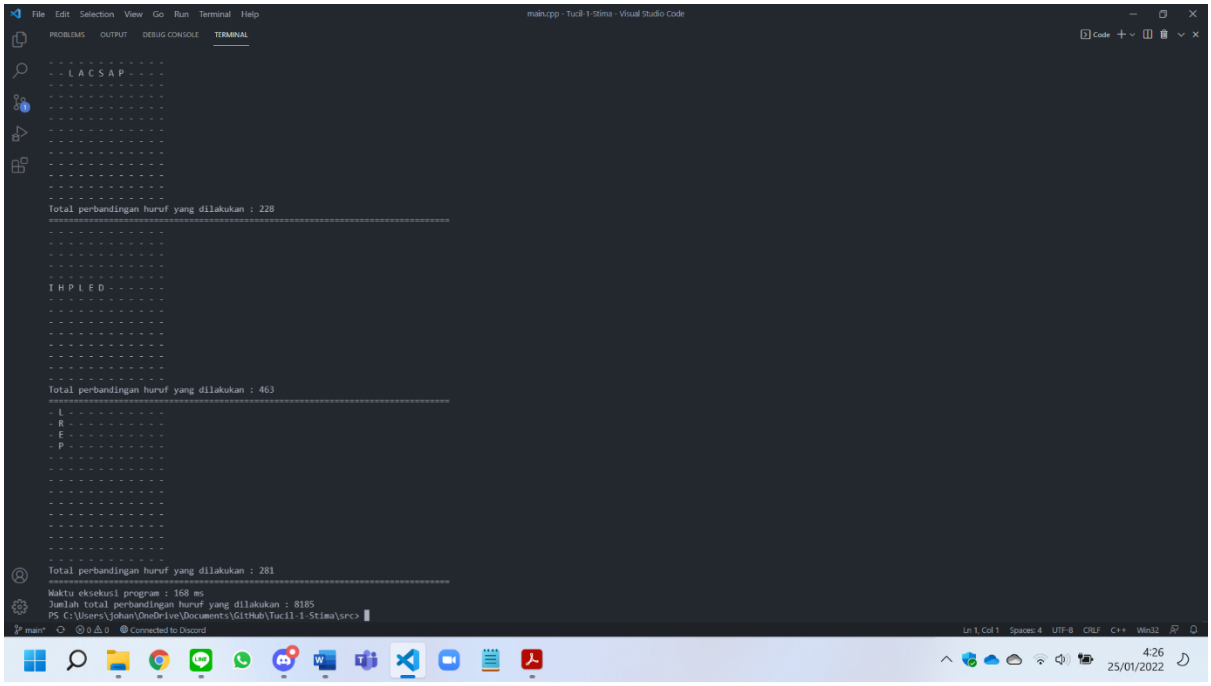
```
                                    if ((l == a) && (m == b) && (c <
word[i].length())) {
                                        cout << character[l][m] << ' ';
                                        a++;
                                        b++;
                                        c++;
                                    } else {
                                        cout << '-' << ' ';
                                    }
                                }
                                cout << endl;
                            }
                            break;
                        }
                    }
                    if (found) {
                        break;
                    }
                    check += 1;
                    checkall += 1;
                    a = j;
                    b = k;
                    c = 0;
                }
                if ((a + word[i].length() - 1 < baris) && (b -
word[i].length() + 1 >= 0)) { // diagonal ke kiri bawah
                    while (character[a][b] == word[i][c]) {
                        a++;
                        b--;
                        c++;
                        check++;
                        checkall++;
                        if (word[i].length() == c) {
                            found = true;
                            a = j;
                            b = k;
                            c = 0;
                            for (l = 0; l < baris; l++) {
                                for (m = 0; m < kolom; m++) {
                                    if ((l == a) && (m == b) && (c <
word[i].length())) {
                                        cout << character[l][m] << ' ';
                                        a++;
                                        b--;
                                        c++;
                                    } else {
                                        cout << '-' << ' ';
                                    }
```

```
                                }
                                cout << endl;
                            }
                            break;
                        }
                    }
                    if (found) {
                        break;
                    }
                    check += 1;
                    checkall += 1;
                    a = j;
                    b = k;
                    c = 0;
                }
            }
            if (found) {
                cout << "Total perbandingan huruf yang dilakukan : " << check
<< endl;
                cout <<
"================================================================================
===" << endl;
                break;
            }
        }
    }
    auto done = std::chrono::high_resolution_clock::now();
    std::cout << "Waktu eksekusi program : " <<
std::chrono::duration_cast<std::chrono::milliseconds>(done-started).count() <<
" ms" << endl;
    cout << "Jumlah total perbandingan huruf yang dilakukan : " << checkall <<
endl;
    return 0;
}
```

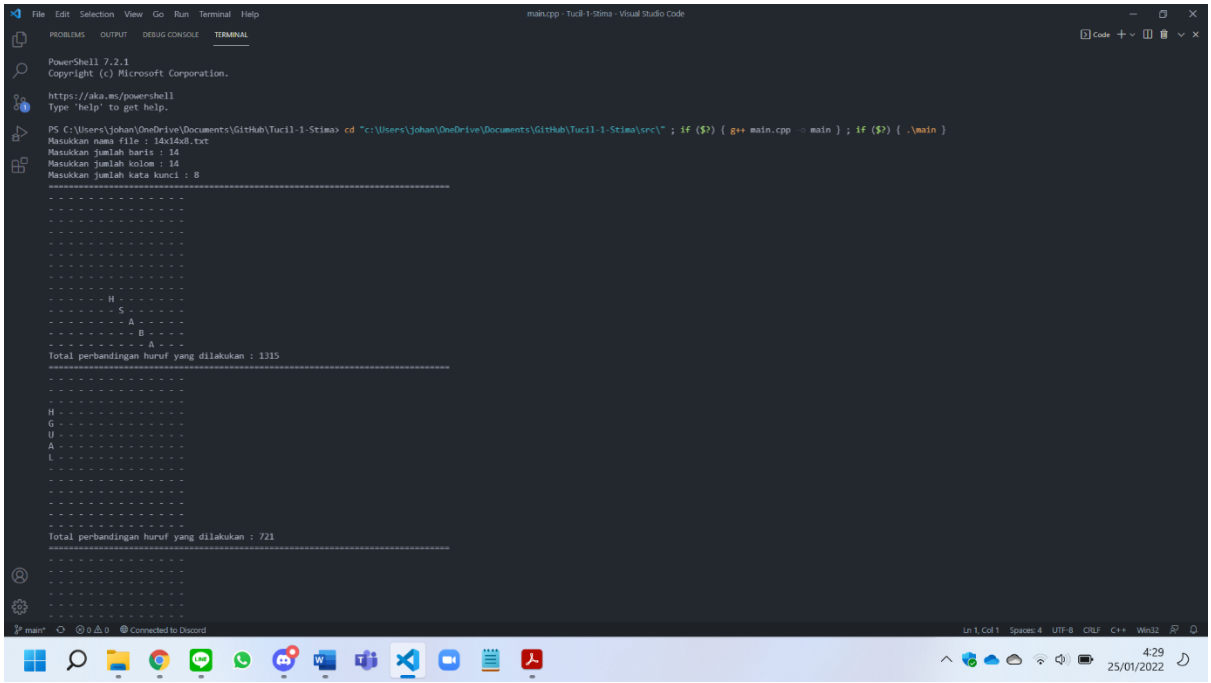
Screenshots

1. 14x12x16.txt

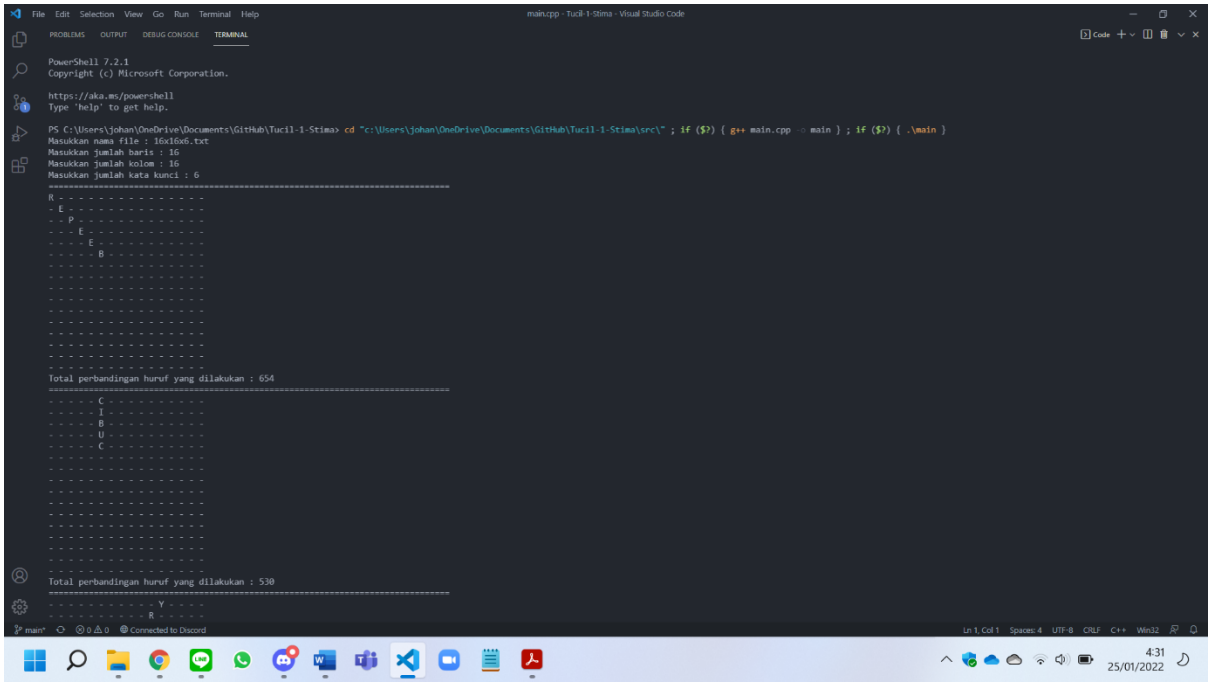


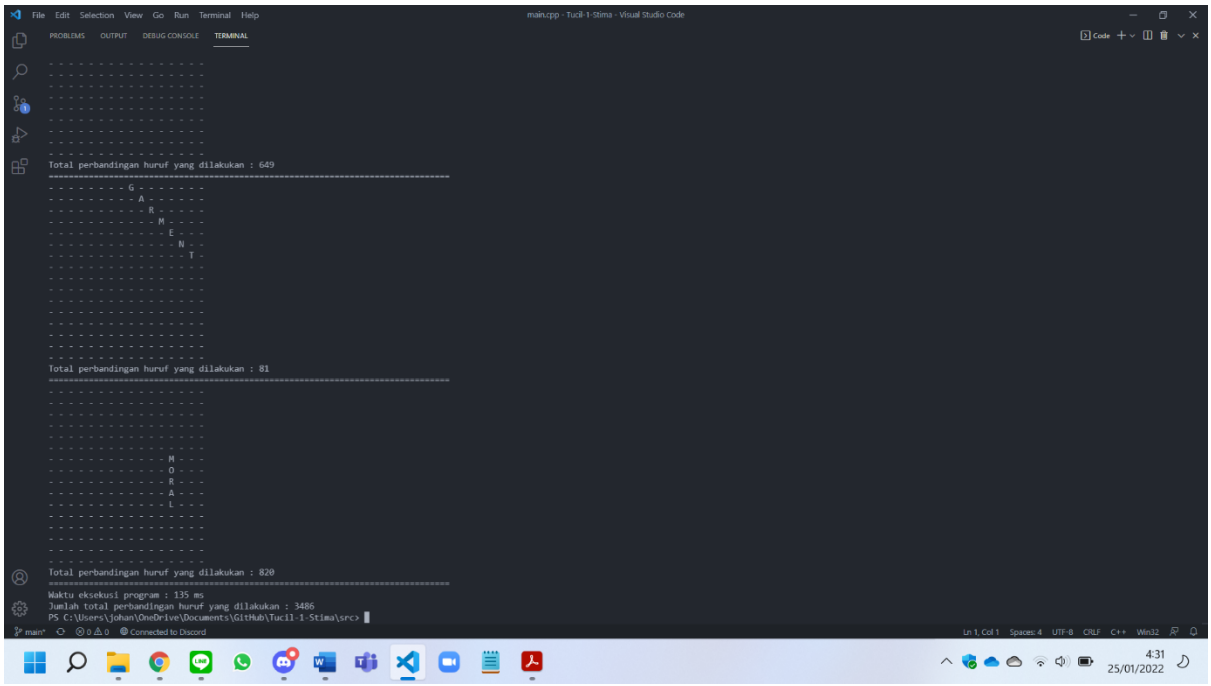


2. 14x14x8.txt

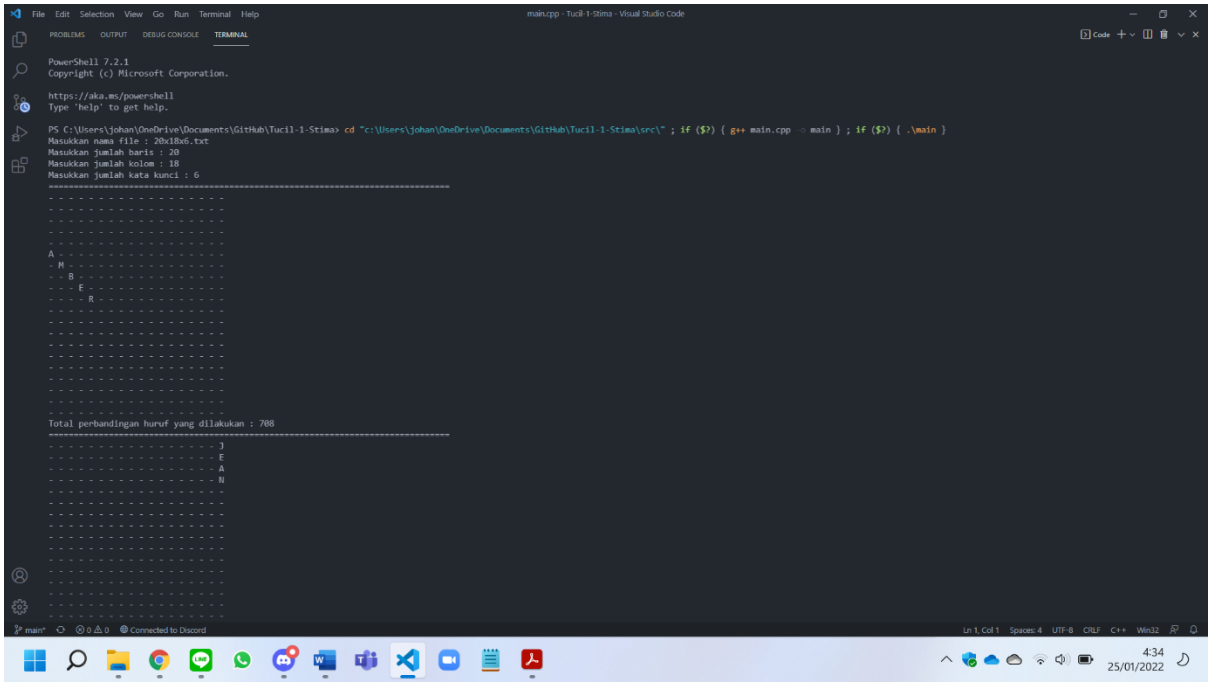


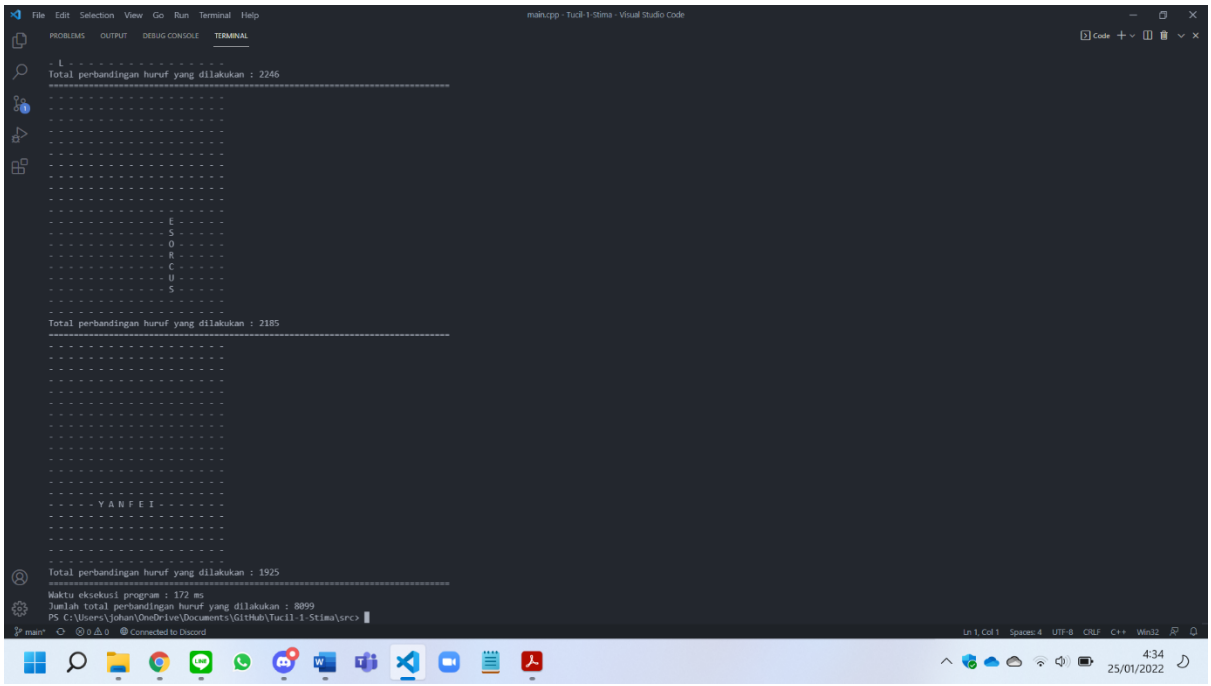
3. 16x16x6.txt



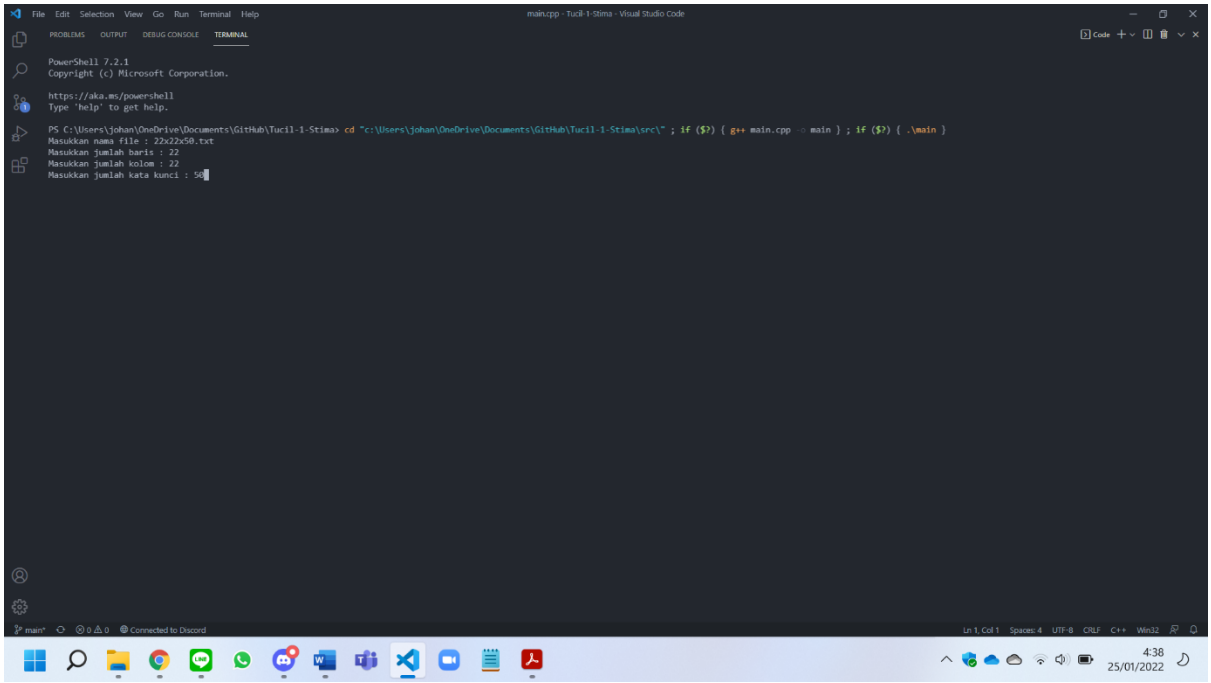


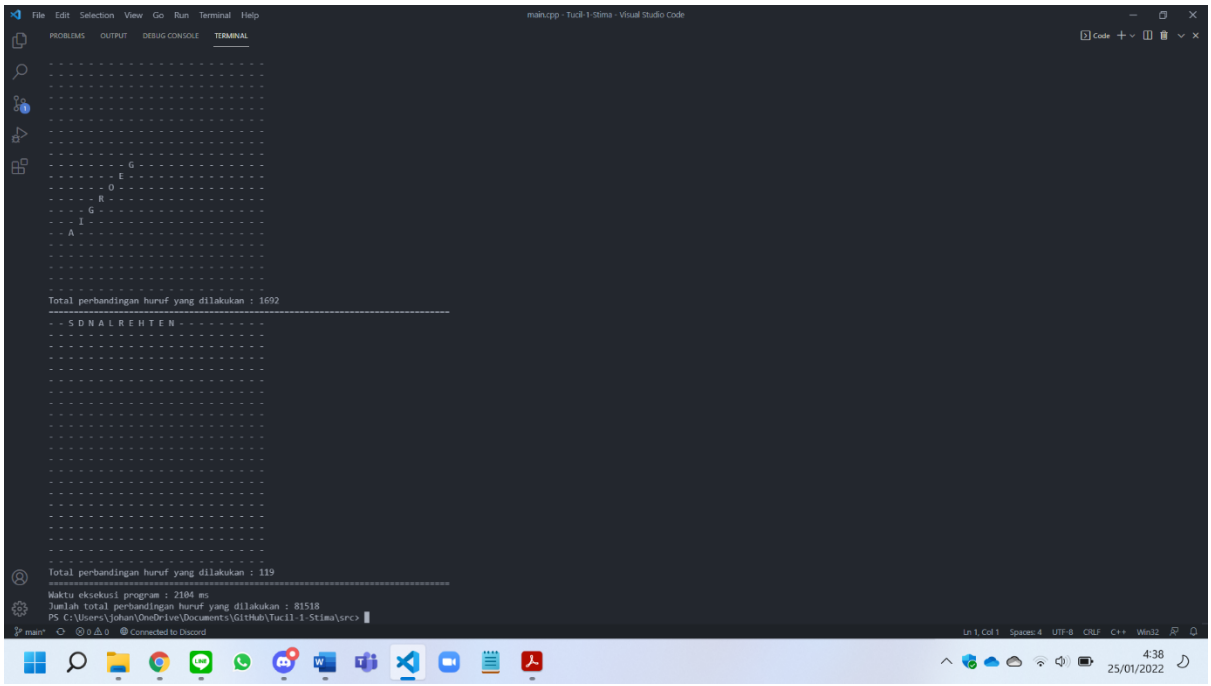
4. 20x18x6.txt



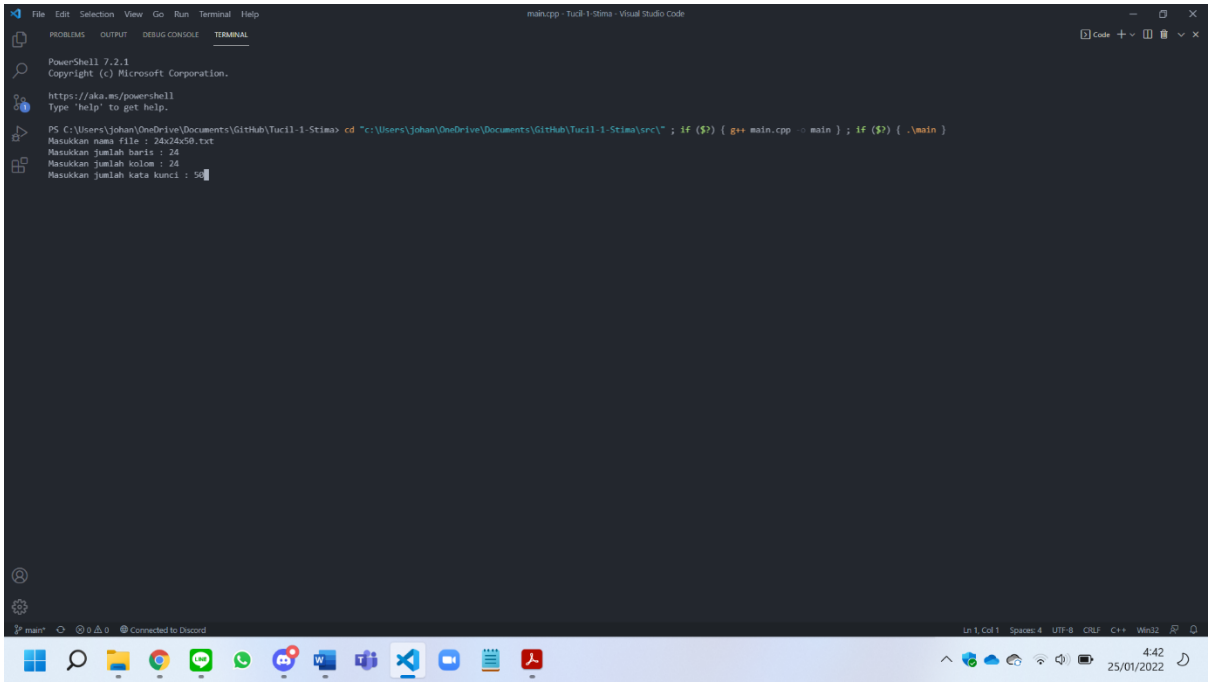


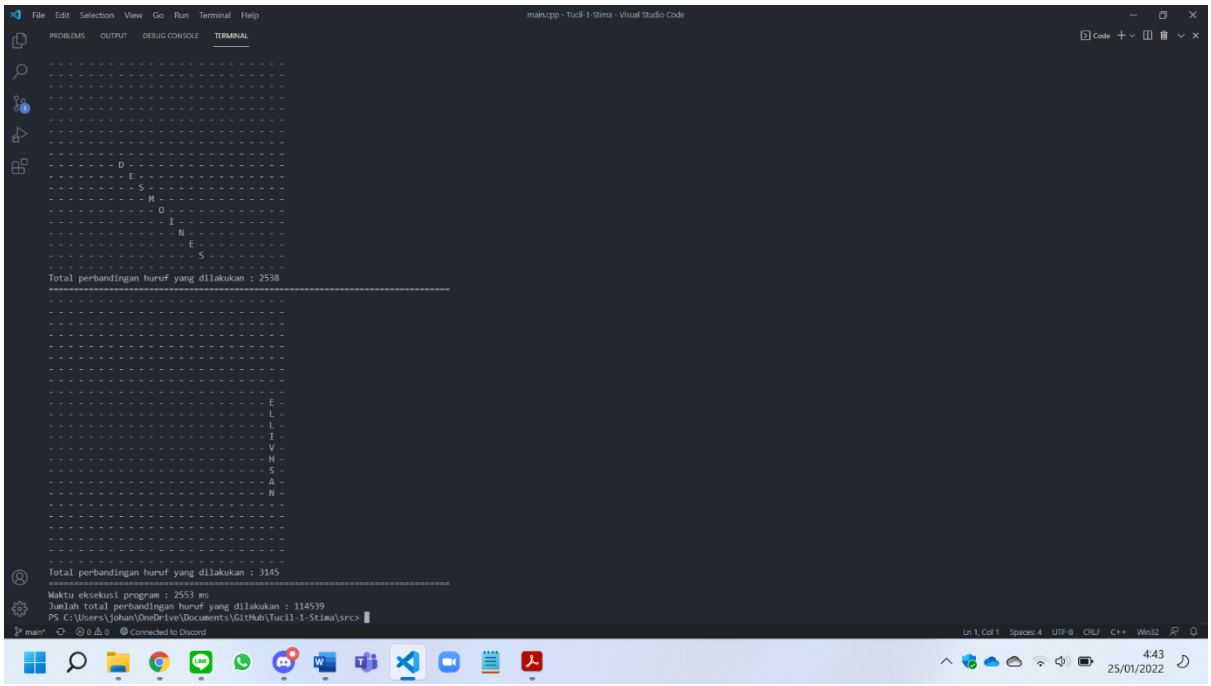
5. 22x22x50.txt



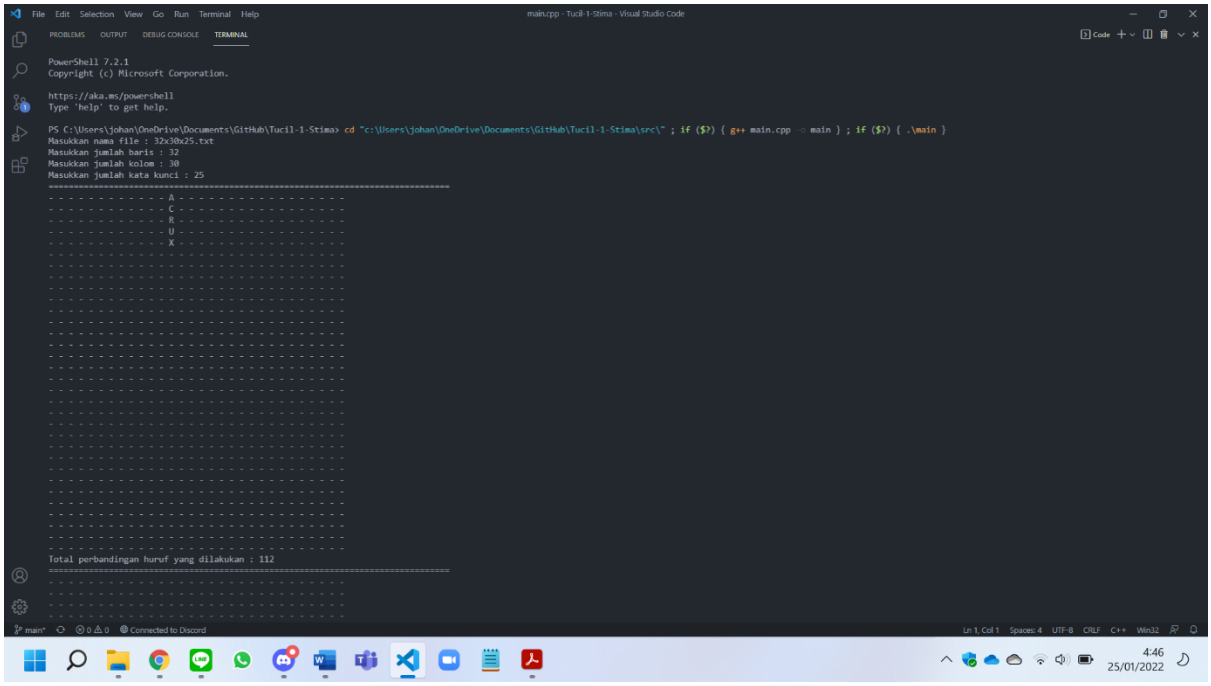


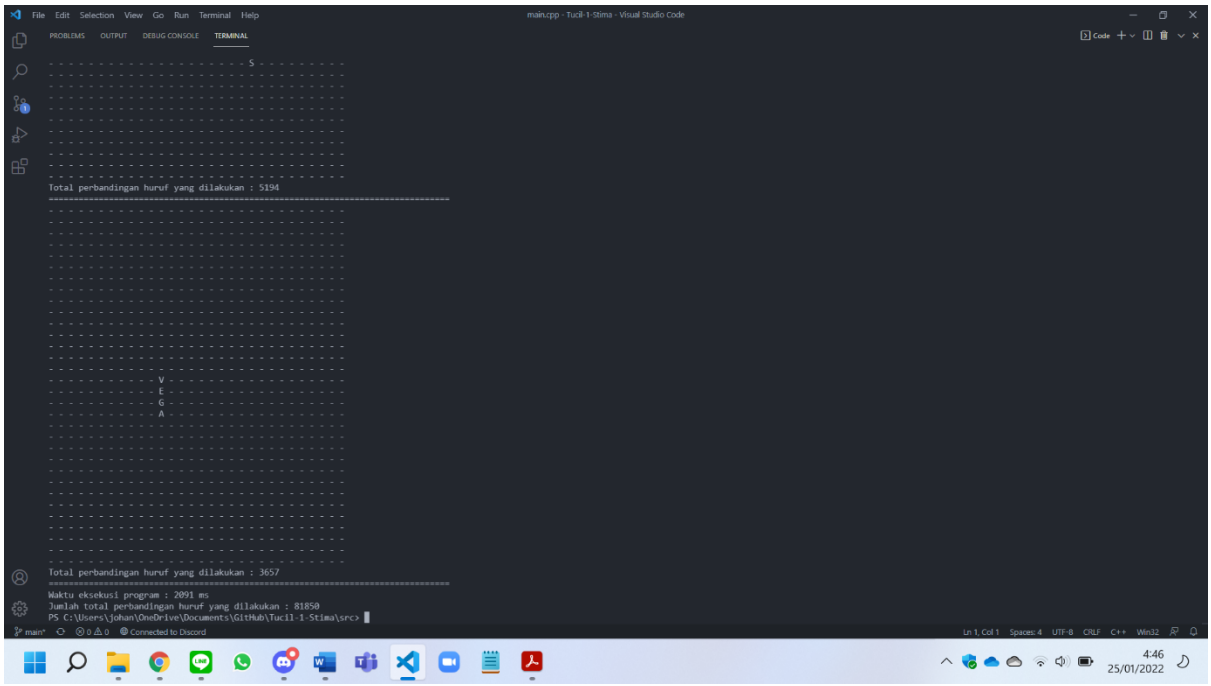
6. 24x24x50.txt



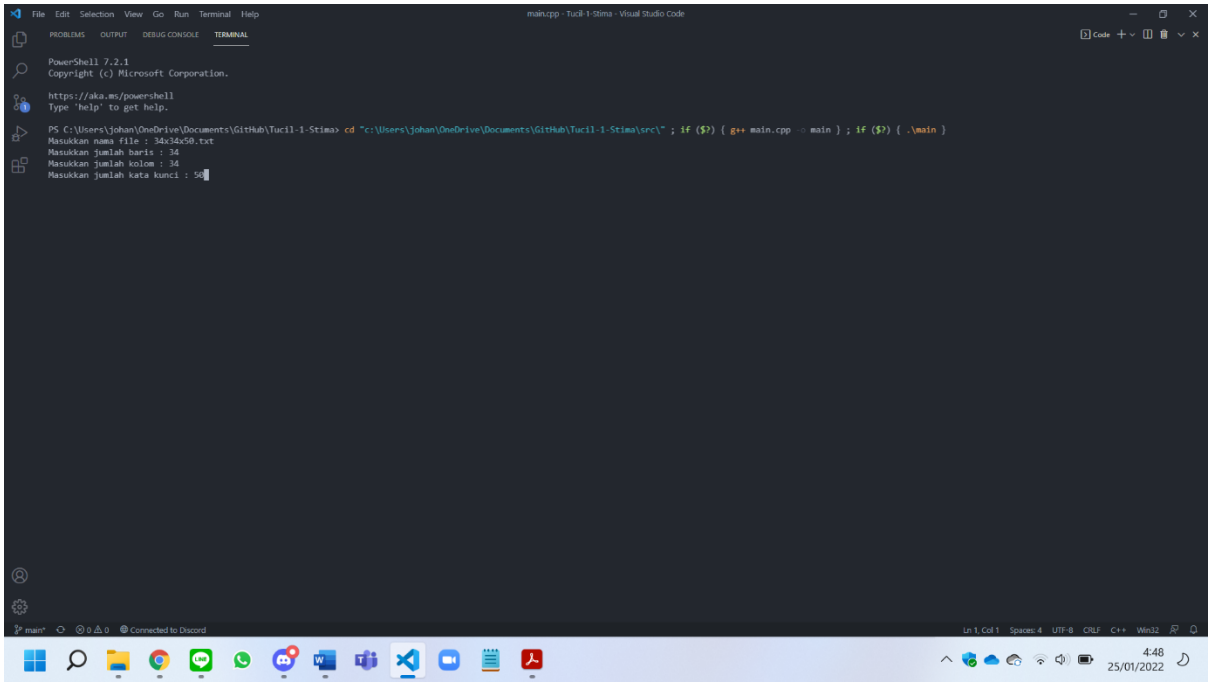


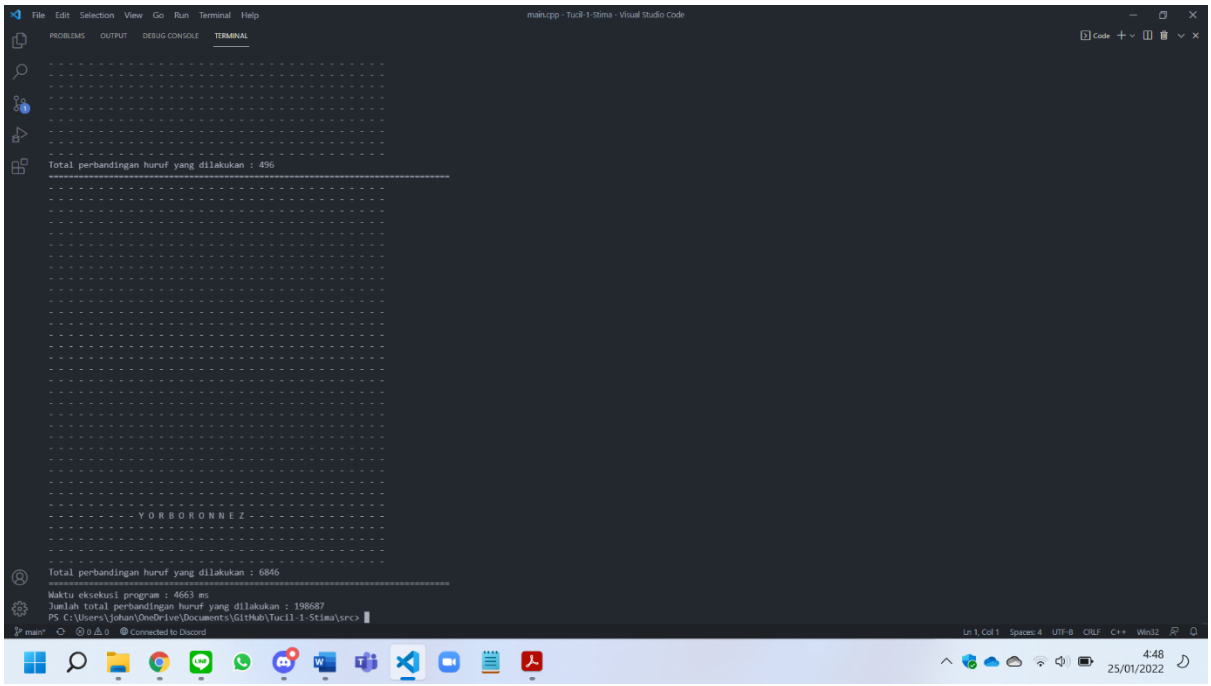
7. 32x30x25.txt



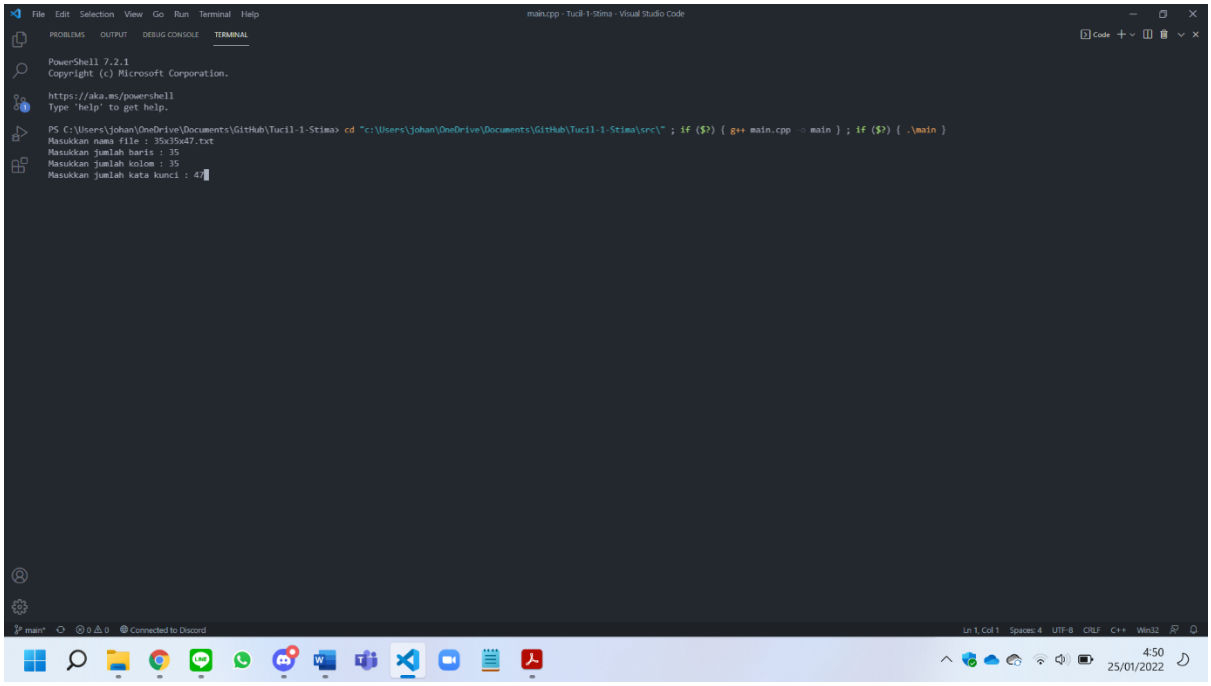


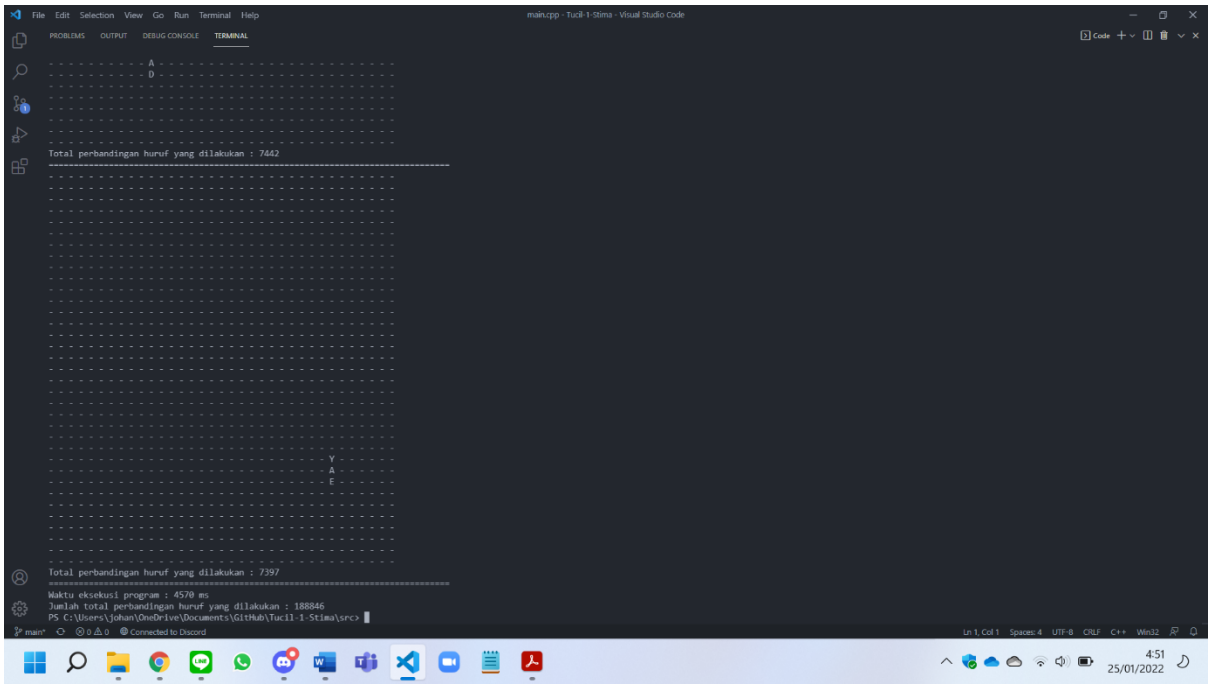
8. 34x34x50.txt





9. 35x35x47.txt





Alamat repository GitHub : https://github.com/johannes-ws/Tucil-1-Stima