Laporan Tucil 1

Nama: Devinzen

NIM: 13522064

Repository github

https://github.com/Devinzenzhang/Tucil1_13522064

Algoritma Brute Force

1. Jika ukuran buffer lebih kecil daripada 2, reward maksimalnya 0 karena sekuens panjangnya minimal 2.
2. Hitung reward maksmial yang bisa didapatkan dari sekuens, kalau lebih besar dari 0 lanjut.
3. Cari semua kemungkinan bergerak (kalau buffer belum penuh), lalu hitung reward yang paling besar dan gerakannya secara rekursif. Jika reward yang didapatkan sama dengan reward maksimal langsung berhenti.

Source Code bagian algoritma (C)

```c
void We_Blowing_Up_Our_Computers_With_This_One(struct moves *X){
    if(X->size != buffer_size){
        struct moves Y; Y.coordinates = (int *) malloc (buffer_size * 2 * sizeof(int)); Copy_Moves(*X, &Y);
        Y.coordinates[Y.size * 2] = Y.coordinates[(Y.size - 1) * 2];
        Y.coordinates[Y.size * 2 + 1] = Y.coordinates[(Y.size - 1) * 2 + 1];
        Y.size += 1;
        struct moves temp; temp.coordinates = (int *) malloc (buffer_size * 2 * sizeof(int)); Copy_Moves(Y, &temp);
        if (X->size % 2){ // vertikal
            for (int i = 1; i <= matrix_height; i++){
                Y.coordinates[(Y.size - 1) * 2 + 1] = i;
                if (!check_duplicate(Y)){
                    Y.reward = Check_Sekuens(Y);
                    We_Blowing_Up_Our_Computers_With_This_One(&Y);
                    if (Y.reward > X->reward){Copy_Moves(Y, X); if (X->reward == max_reward){break;}}
                    Copy_Moves(temp, &Y);
                }
            }
        } else { // horizontal
            for (int i = 1; i <= matrix_width; i++){
                Y.coordinates[(Y.size - 1) * 2] = i;
                if (!check_duplicate(Y)){
                    Y.reward = Check_Sekuens(Y);
                    We_Blowing_Up_Our_Computers_With_This_One(&Y);
                    if (Y.reward > X->reward){Copy_Moves(Y, X); if (X->reward == max_reward){break;}}
                    Copy_Moves(temp, &Y);
                }
            }
        }
        free(Y.coordinates); free(temp.coordinates);
    }
}
```

fungsi rekursif

```c
int Check_Sekuens(struct moves check){
    int reward = 0;
    int isThere;
    for (int i = 0; i < number_of_sequences; i++){
        isThere = 0;
        if (sequence_list[i].size <= check.size){
            for (int j = 0; j <= (check.size - sequence_list[i].size); j++){
                isThere = 1;
                for (int k = 0; k < sequence_list[i].size; k++){
                    if (
                        (sequence_list[i].tokens[k * 2] !=
                        matrix[((check.coordinates[(j + k) * 2] - 1) + (check.coordinates[(j + k) * 2 + 1] - 1) * matrix_width) * 2])
                        ||
                        (sequence_list[i].tokens[k * 2 + 1] !=
                        matrix[((check.coordinates[(j + k) * 2] - 1) + (check.coordinates[(j + k) * 2 + 1] - 1) * matrix_width) * 2 + 1])
                    ){
                        isThere = 0; break;
                    }
                }
                if (isThere){break;}
            }
        }
        if(isThere){reward += sequence_list[i].reward;}
    }
    return reward;
}
```

cek reward yang diberikan

```c
struct moves AbsoluteSolver64(){
    struct moves result;
    result.size = 0; result.reward = 0;
    if (buffer_size < 2){return result;} // sekuens minimal dua tokens
    for (int i = 0; i < number_of_sequences; i++){ // hitung max reward
        if (sequence_list[i].reward > 0){max_reward += sequence_list[i].reward;}
    }
    if (max_reward <= 0){return result;}
    result.coordinates = (int *) malloc (buffer_size * 2 * sizeof(int));
    result.coordinates[0] = 1; result.coordinates[1] = 1; result.size = 1;
    We_Blowing_Up_Our_Computers_With_This_One(&result);
    return result;
}

void show_moves(struct moves P){
    printf("%d\n", P.reward);
    for (int i = 0; i < P.size; i++){
        printf("%c", matrix[((P.coordinates[i * 2] - 1) + (P.coordinates[i * 2 + 1] - 1) * matrix_width) * 2]);
        printf("%c", matrix[((P.coordinates[i * 2] - 1) + (P.coordinates[i * 2 + 1] - 1) * matrix_width) * 2 + 1]);
        if (i < (P.size - 1)){printf(" ");} else {printf("\n");}
    }
    for (int i = 0; i < P.size; i++){
        printf("%d, %d\n", P.coordinates[i * 2], P.coordinates[i * 2 + 1]);
    }
}
```

```c
void Copy_Moves(struct moves In, struct moves *Out){ // Out sudah dialokasi
    Out->size = In.size; Out->reward = In.reward;
    for (int i = 0; i < In.size; i++){
        Out->coordinates[i * 2] = In.coordinates[i * 2];
        Out->coordinates[i * 2 + 1] = In.coordinates[i * 2 + 1];
    }
}

int check_duplicate(struct moves X){ // hanya yg terakhir
    int isDupe = 0; int L = X.size - 1;
    for(int i = 0; i < L; i++){
        if ((X.coordinates[i * 2] == X.coordinates[L * 2]) && (X.coordinates[i * 2 + 1] == X.coordinates[L * 2 + 1])){isDupe = 1; break;}
    }
    return isDupe;
}
```

Screenshot Input/Output

```
PS C:\Users\devin\Tucil1_13522064\bin> ./BruteForcer64_Windows.exe
pilih cara masukan:
1. file .txt
2. input teks (matriks, sekuens dan rewardnya random)
1
Masukkan nama file (di folder txt, jangan ada spasi): test_empty.txt
0

0 ms

Apakah ingin menyimpan solusi? (y/n) y
Nama file yang mau disimpan (jangan ada spasi): sol_empty.txt
```

Output kalau reward maksimalnya 0

```
PS C:\Users\devin\Tucil1_13522064\bin> ./BruteForcer64_Windows.exe
pilih cara masukan:
1. file .txt
2. input teks (matriks, sekuens dan rewardnya random)
1
Masukkan nama file (di folder txt, jangan ada spasi): test_shortpath.txt
69
AA 11 AA 11 AA 11 BB 11 BB 22
1, 1
1, 2
2, 2
2, 1
3, 1
3, 2
4, 2
4, 1
5, 1
5, 2

0 ms

Apakah ingin menyimpan solusi? (y/n) y
Nama file yang mau disimpan (jangan ada spasi): sol_shortpath.txt
```

Best case scenario (langsung dapat reward maksmial)

```
PS C:\Users\devin\Tucil1_13522064\bin> ./BruteForcer64_Windows.exe
pilih cara masukan:
1. file .txt
2. input teks (matriks, sekuens dan rewardnya random)
1
Masukkan nama file (di folder txt, jangan ada spasi): test_longpath.txt
420
AA 11 AA 11 AA 11 BB 11 BB 22
1, 1
1, 9
7, 9
7, 8
6, 8
6, 9
5, 9
5, 8
4, 8
4, 9

7430 ms

Apakah ingin menyimpan solusi? (y/n) y
Nama file yang mau disimpan (jangan ada spasi): sol_longpath.txt
```

Menelusuri semua jalur untuk matriks 7x9 dengan ukuran buffer 10

```
PS C:\Users\devin\Tucil1_13522064\bin> ./BruteForcer64_Windows.exe
pilih cara masukan:
1. file .txt
2. input teks (matriks, sekuens dan rewardnya random)
2
Masukkan jumlah token unik: 7
Masukkan token-tokennya: DE AD BE EF C0 FF EE
Masukkan ukuran buffer: 9
Masukkan ukuran matriks: 5 6
Masukkan jumlah sekuens: 7
Masukkan ukuran maksimal sekuens: 8
Generating Matrix...
BE AD AD C0 C0
AD FF C0 EE EE
EE EE DE AD EF
FF EE DE EF DE
AD AD AD BE EE
FF EE EE C0 EE
Generating Sequence List...
C0 DE C0 FF C0
reward: 50
AD FF
reward: 90
C0 C0 AD C0 EF AD DE C0
reward: -65
FF BE DE DE EE DE
reward: 45
DE FF DE EF FF EE
reward: -60
AD AD BE DE EF AD EF EE
reward: 175
AD EF EF C0 AD FF C0
reward: 160
90
BE AD FF
1, 1
1, 2
2, 2

80 ms

Apakah ingin menyimpan solusi? (y/n) y
Nama file yang mau disimpan (jangan ada spasi): sol_random.txt
```

```
PS C:\Users\devin\Tucil1_13522064\bin> ./BruteForcer64_Windows.exe
pilih cara masukan:
1. file .txt
2. input teks (matriks, sekuens dan rewardnya random)
2
Masukkan jumlah token unik: 3
Masukkan token-tokennya: 69 AD BC
Masukkan ukuran buffer: 9
Masukkan ukuran matriks: 7 8
Masukkan jumlah sekuens: 7
Masukkan ukuran maksimal sekuens: 8
Generating Matrix...
69 69 AD AD 69 BC AD
BC 69 BC AD AD BC 69
AD 69 AD BC AD BC BC
BC BC AD AD 69 BC BC
AD 69 BC BC 69 BC AD
69 AD AD BC 69 AD AD
AD BC 69 69 BC 69 AD
BC BC BC BC BC BC BC
Generating Sequence List...
69 BC 69 AD 69 BC
reward: 70
AD AD 69
reward: 105
69 69
reward: 90
69 AD BC BC
reward: -70
69 BC AD
reward: 40
BC 69 BC BC AD
reward: 120
BC AD
reward: 85
400
69 BC 69 BC BC AD AD 69 69
1, 1
1, 2
2, 2
2, 4
1, 4
1, 3
3, 3
3, 7
4, 7

2447 ms

Apakah ingin menyimpan solusi? (y/n) y
Nama file yang mau disimpan (jangan ada spasi): sol_random2.txt
```

```
PS C:\Users\devin\Tucil1_13522064\bin> ./BruteForcer64_Windows.exe
pilih cara masukan:
1. file .txt
2. input teks (matriks, sekuens dan rewardnya random)
1
Masukkan nama file (di folder txt, jangan ada spasi): test1.txt
50
7A BD 7A BD 1C BD 55
1, 1
1, 4
3, 4
3, 5
6, 5
6, 3
1, 3

4 ms

Apakah ingin menyimpan solusi? (y/n) y
Nama file yang mau disimpan (jangan ada spasi): sol1.txt
```

Test case yang diberikan

Checklist

Poin	Ya	Tidak
1. Program berhasil dikompilasi tanpa kesalahan	✓	
2. Program berhasil dijalankan	✓	
3. Program dapat membaca masukan berkas .txt	✓	
4. Program dapat menghasilkan masukan secara acak	✓	
5. Solusi yang diberikan program optimal	✓	
6. Program dapat menyimpan solusi dalam berkas .txt	✓	
7. Program memiliki GUI		✓