IF2211 Strategi Algoritma

Tugas Kecil 1

Penyelesaian Permainan Queens Linkedin

Dipersiapkan oleh:

13524046 - Farrell Limjaya

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
JL. GANESHA 10, BANDUNG 40132
2026

I. Algoritma Brute Force

Diterapkan dalam bentuk 2 function yaitu solve dan validasi, dimana function solve(r) akan menjadi iterasinya(rekursif) untuk membuat semua kemungkinan yang ada bisa diperiksa dan function validasi sebagai fungsi pengecekan akhir dari setiap kemungkinan.

```
FUNCTION solve(r) //r adalah row
  IF r = N THEN
    kasus ← kasus + 1
    pushQueue()
    RETURN validasi()
  ENDIF

  FOR setiap kolom c dari 0 sampai N-1 DO
    pos[r] ← c

    IF solve(r + 1) = true THEN
      RETURN true
    ENDIF
  ENDFOR

  RETURN false
END FUNCTION
```

Pada function solve(rekursif), yang dilakukan adalah mencoba semua posisi queen, dilakukan melalui (FOR setiap kolom c dari 0 sampai N-1 DO pos[r] ← c) untuk percobaan penempatan queen di setiap kolom, dan (IF solve(r + 1) = true) untuk berpindah baris. Ketika mencapai bagian (IF r = N) maka sudah mencapai baris terakhir dan saatnya melakukan validasi() untuk pengecekan apakah memenuhi untuk menjadi jawaban. Pada bagian ini juga banyak kasus yang ditinjau bertambah. pushQueue digunakan untuk menyimpan 20 konfigurasi percobaan pengecekan terakhir.

```
FUNCTION validasi
  Reset queen, colUsed, regionUsed menjadi 0

  FOR setiap baris r dari 0 sampai N-1 DO
    c ← pos[r]

    IF kolom c sudah terpakai THEN
      RETURN false
    ENDIF
    tandai kolom c sebagai terpakai

    IF region papan[r][c] sudah terpakai THEN
      RETURN false
    ENDIF
```

```
      tandai region papan[r][c] sebagai terpakai

      letakkan queen di posisi (r, c)
   ENDFOR

   FOR setiap baris r dari 0 sampai N-1 DO
      c ← pos[r]

      FOR setiap arah i dari 0 sampai 7 DO
         nr ← r + dr[i]
         nc ← c + dc[i]

         IF (nr, nc) berada di dalam papan
           DAN terdapat queen di (nr, nc) THEN
            RETURN false
         ENDIF
      ENDFOR
   ENDFOR

   RETURN true
END FUNCTION
```

Pada function validasi() yang dilakukan adalah memeriksa apakah sudah memenuhi aturan permainan Queen, yaitu apakah semua queen berada pada posisi yang aman, yaitu tidak ada yang sama baris, sama kolom, sama region, dan saling berdekatan pada salah satu dari 8 arah. Pengecekan untuk sama baris tidak dilakukan karena sudah pasti tidak ada queen pada baris yang sama melalui penempatan pada function solve.

```
IF kolom c sudah terpakai THEN
        RETURN false
      ENDIF
      tandai kolom c sebagai terpakai
```

Ini dilakukan untuk memastikan tidak melanggar aturan kolom yang sama. Jika terdeteksi belum ada queen pada kolom tersebut maka bisa, tetapi kolom tersebut akan langsung ditandai sebagai kolom yang sudah dipakai. Jika queen selanjutnya ditempatkan pada kolom yang sama, maka langsung return false(tidak valid).

```
IF region papan[r][c] sudah terpakai THEN
        RETURN false
      ENDIF
      tandai region papan[r][c] sebagai terpakai
```

Sama dengan kolom yang sama, tetapi ini dilakukan pada region.

```
 letakkan queen di posisi (r, c)
```

Ini dilakukan sebagai asumsi sementara queen sudah benar dan digunakan untuk membantu pengecekan aturan terakhir.

```
 FOR setiap arah i dari 0 sampai 7 DO
        nr ← r + dr[i]
        nc ← c + dc[i]

        IF (nr, nc) berada di dalam papan
```

DAN terdapat queen di (nr, nc) THEN
RETURN false
ENDIF
ENDFOR

Ini sebagai pengecekan yang 8 arah di sekitar queen. Melalui nr dan nc, dimana dr[i] dan dc[i] adalah angka yang akan membantu memeriksa 8 koordinat di sekitar queen(kiri, kiri atas, atas, kanan atas, kanan, kanan bawah, bawah, dan kiri bawah.). Jika ada yang terdeteksi sebagai queen, maka akan return false dan tidak valid. Jika tidak valid, maka akan kembali ke function solve yang akan memberikan kemungkinan posisi selanjutnya. Jika ternyata valid, maka solusi permainan ditemukan.

II. Source Code(Program)

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>

char board[50][50];
int queen[50][50];
int colUsed[50];
int regionUsed[256];

int N;
long long kasus = 0;
int pos[50];
clock_t last_display = 0;

int dr[8] = {-1,-1,-1,0,0,1,1,1};
int dc[8] = {-1,0,1,-1,1,-1,0,1};

void displayLive(int currentRow)
{
    system("cls");
    for(int i=0; i<N; i++)
    {
        for(int j=0; j<N; j++)
        {
            if(i < currentRow && queen[i][j])
            {
                printf("#");
            }
            else if(i == currentRow)
            {
```

```
                if(j == pos[currentRow])
                    printf("#");
                else
                    printf("%c", board[i][j]);
            }
            else
            {
                printf("%c", board[i][j]);
            }
        }
        printf("\n");
    }
    fflush(stdout);
}

int validasi()
{
    memset(queen, 0, sizeof(queen));
    memset(colUsed, 0, sizeof(colUsed));
    memset(regionUsed, 0, sizeof(regionUsed));
    for(int r=0;r<N;r++)
    {
        int c=pos[r];
        if(colUsed[c])
        {
            return 0;
        }
        colUsed[c]=1;
        if(regionUsed[(int)board[r][c]])
        {
            return 0;
        }
        regionUsed[(int)board[r][c]]=1;
        queen[r][c]=1;
    }
    for(int r=0;r<N;r++)
    {
        int c=pos[r];
        for(int i=0;i<=7;i++)
        {
            int nr=r+dr[i];
            int nc=c+dc[i];
            if(nr>=0 && nc>=0 && nr<N && nc<N && queen[nr][nc])
```

```
            {
                return 0;
            }
        }
    }
    return 1;
}

int solve(int r)
{
    if(r==N)
    {
        kasus++;
        return validasi();
    }
    else
    {
        for(int c=0;c<N;c++)
        {
            pos[r]=c;
            clock_t now = clock();
            double elapsed = (double)(now - last_display) * 1000
/ CLOCKS_PER_SEC;
            if(elapsed >= 300)
            {
                displayLive(r);
                last_display = now;
            }

            if(solve(r+1))
            {
                return 1;
            }
        }
        return 0;
    }
}

int validateInput()
{
    int region[256] = {0};
    int unique = 0;
    for (int i = 0; i < N; i++)
```

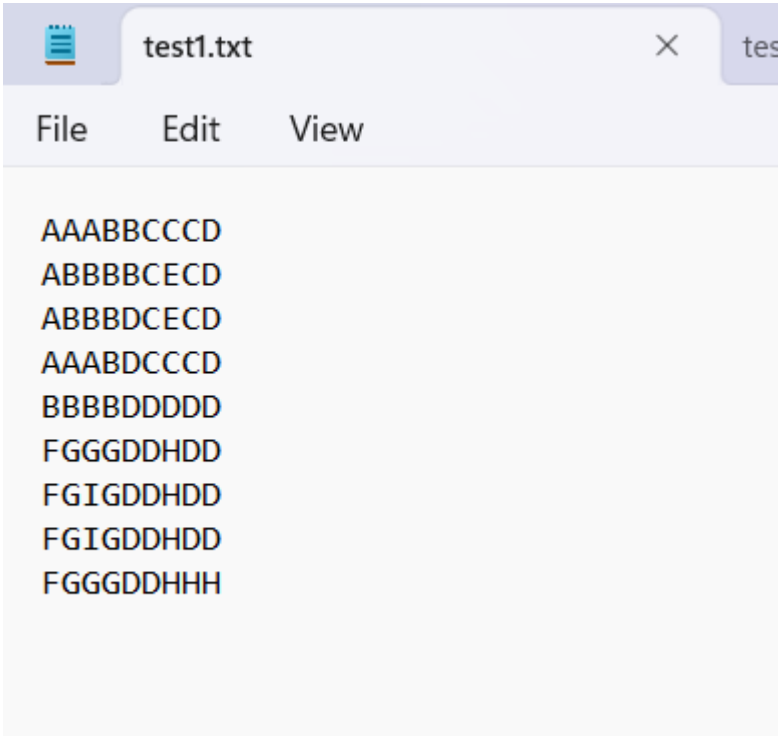
```
    {
        if ((int)strlen(board[i]) != N)
        {
            return 0;
        }
        for (int j = 0; j < N; j++)
        {
            if (!region[(int)board[i][j]])
            {
                region[(int)board[i][j]] = 1;
                unique++;
            }
        }
    }
    return unique == N;
}

void saveSolution(char board[50][50], int N, double waktu, long
long kasus, int found)
{
    char name[100];
    char filename[120];
    printf("Masukkan nama file output: ");
    scanf("%s", name);

    sprintf(filename, "../test/%s.txt", name);

    FILE *f = fopen(filename, "w");
    if (f == NULL)
    {
        printf("Gagal membuat file\n");
        return;
    }

    if (!found)
    {
        fprintf(f, "\nTidak ada solusi\n\n");
    }
    else
    {
        for (int i = 0; i < N; i++)
        {
            for (int j = 0; j < N; j++)
```

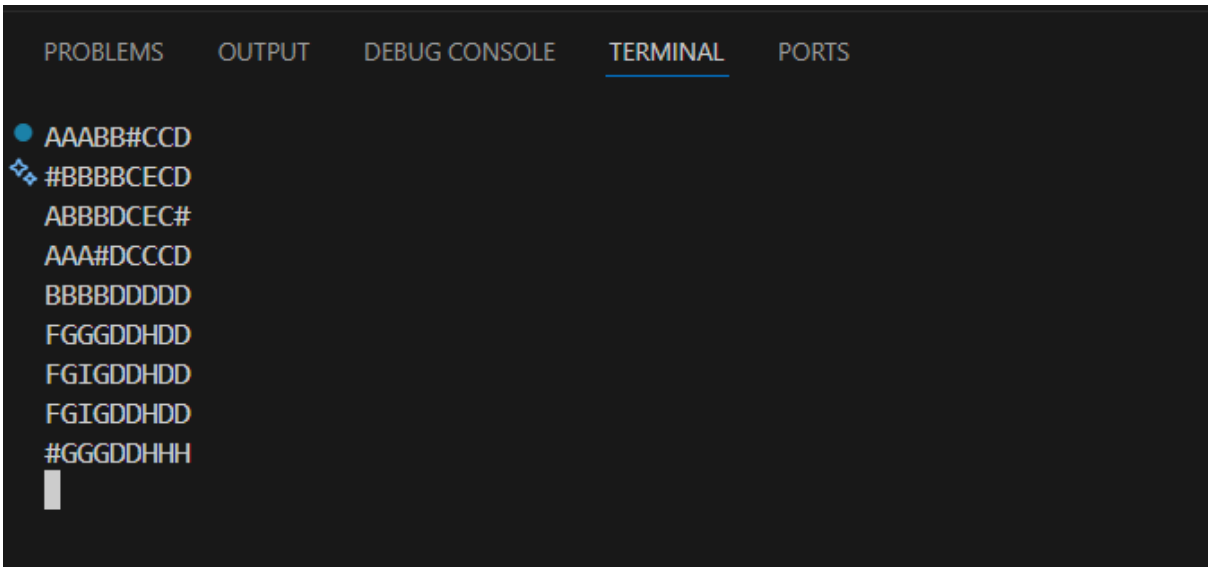
```
            {
                if (queen[i][j])
                    fputc('#', f);
                else
                    fputc(board[i][j], f);
            }
            fputc('\n', f);
        }
    }
    fprintf(f, "\nWaktu pencarian: %f ms\n", waktu);
    fprintf(f, "Banyak kasus yang ditinjau: %lld\n", kasus);
    fclose(f);
    printf("Solusi berhasil disimpan\n");
}


int main()
{
    char filename[100];
    printf("Masukkan nama file input: ");
    scanf("%s", filename);

    FILE *f = fopen(filename, "r");
    if (!f)
    {
        printf("File tidak ditemukan!\n");
        return 1;
    }

    char line[100];
    N = 0;
    while (fgets(line, sizeof(line), f))
    {
        line[strcspn(line, "\n")] = 0;
        strcpy(board[N], line);
        N++;
    }
    fclose(f);
    if (!validateInput())
    {
        printf("Input tidak valid!\n");
        return 1;
    }
```

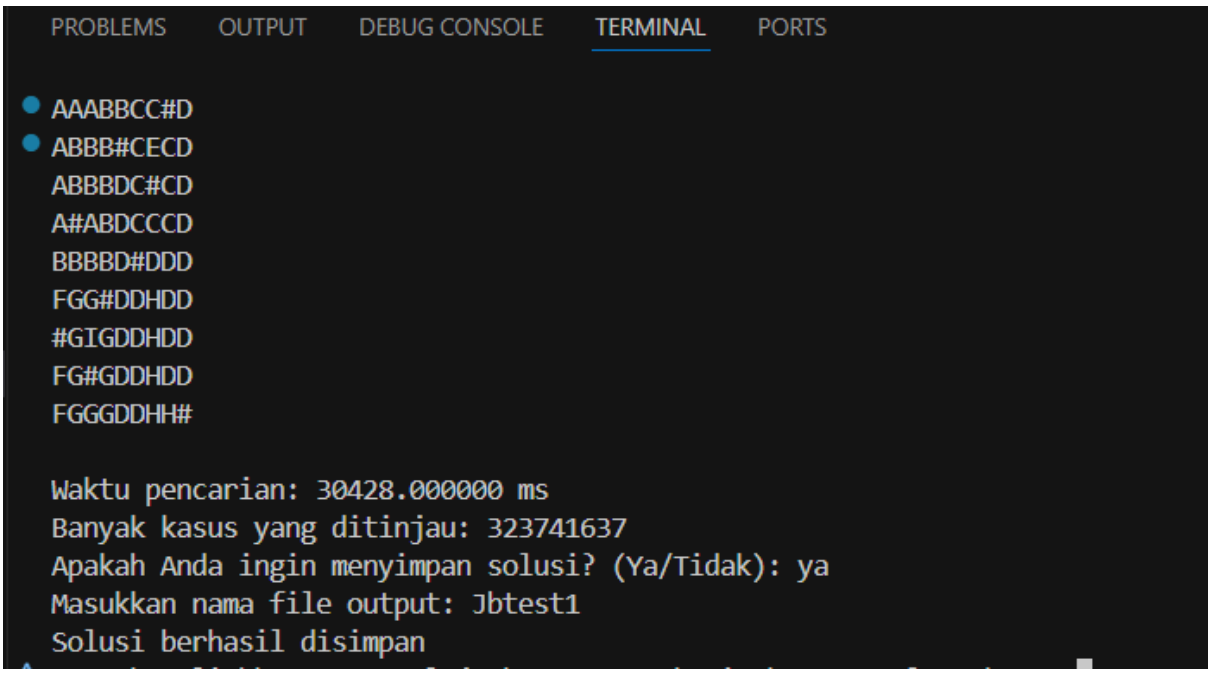
```
    clock_t mulai = clock();
    int found = solve(0);
    clock_t selesai = clock();

    double waktu = (double)(selesai - mulai) * 1000 /
CLOCKS_PER_SEC;

    if (!found)
    {
        system("cls");
        printf("Tidak ada solusi.\n");
        return 0;
    }

    system("cls");
    for (int i = 0; i < N; i++)
    {
        for (int j = 0; j < N; j++)
        {
            if (queen[i][j])
            {
                printf("#");
            }
            else
            {
                printf("%c", board[i][j]);
            }
        }
        printf("\n");
    }
    printf("\nWaktu pencarian: %f ms\n", waktu);
    printf("Banyak kasus yang ditinjau: %lld\n", kasus);
    char pilihan[10];
    printf("Apakah Anda ingin menyimpan solusi? (Ya/Tidak): ");
    scanf("%s", pilihan);

    if (strcmp(pilihan, "Ya") == 0 || strcmp(pilihan, "ya") == 0)
    {
        saveSolution(board, N, waktu, kasus, found);
    }
    return 0;
}
```

III. Tangkapan Layar

1.

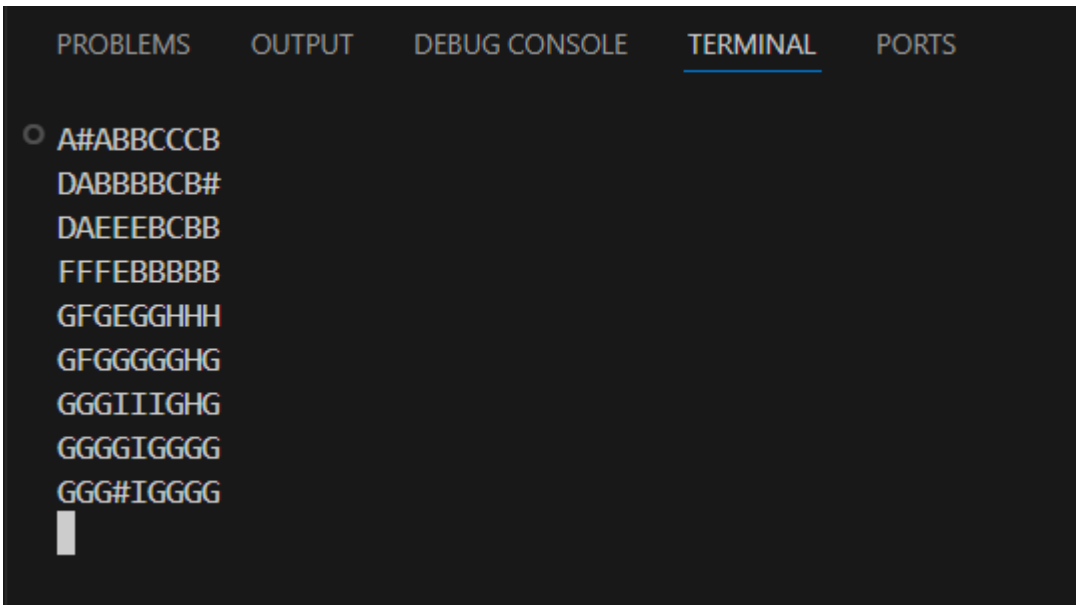


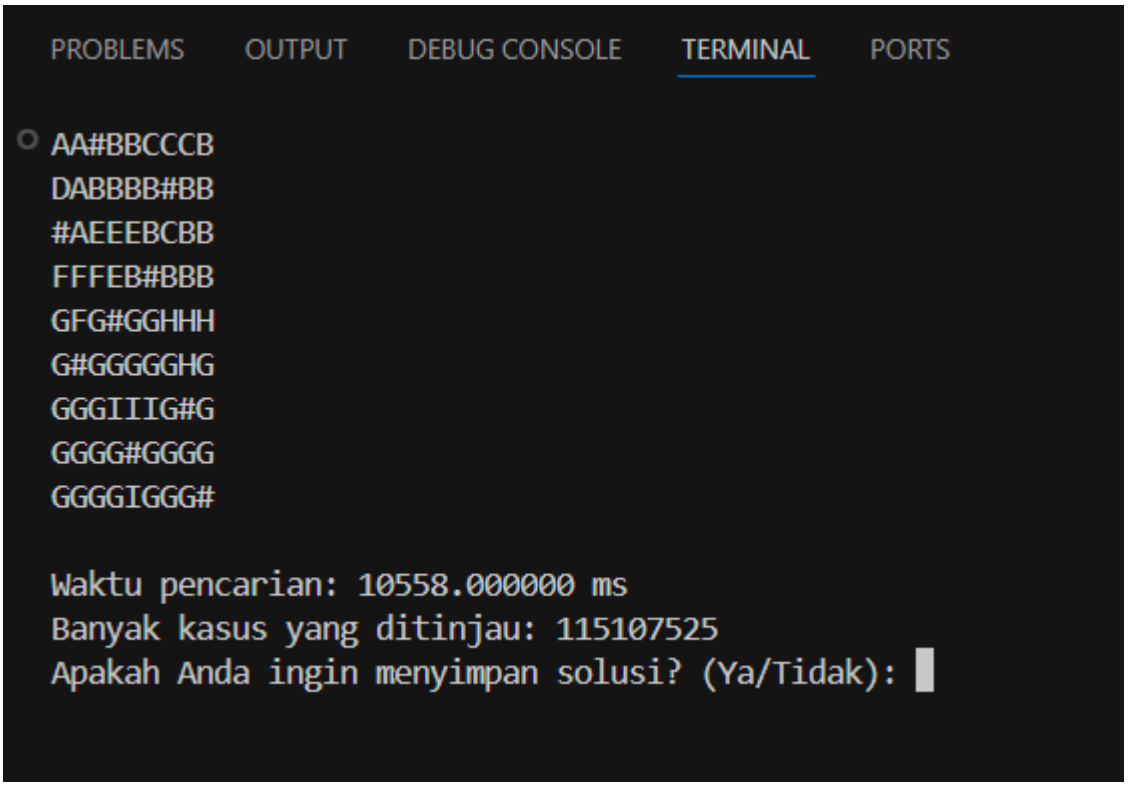



PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
AAABBCC#D
ABBB#CECD
ABBBDC#CD
A#ABDCCCD
BBBBD#DDD
FGG#DDHDD
#GIGDDHDD
FG#GDDHDD
FGGGDDHH#
Waktu pencarian: 30428.000000 ms
Banyak kasus yang ditinjau: 323741637
Apakah Anda ingin menyimpan solusi? (Ya/Tidak): ya
Masukkan nama file output: Jbtest1
Solusi berhasil disimpan

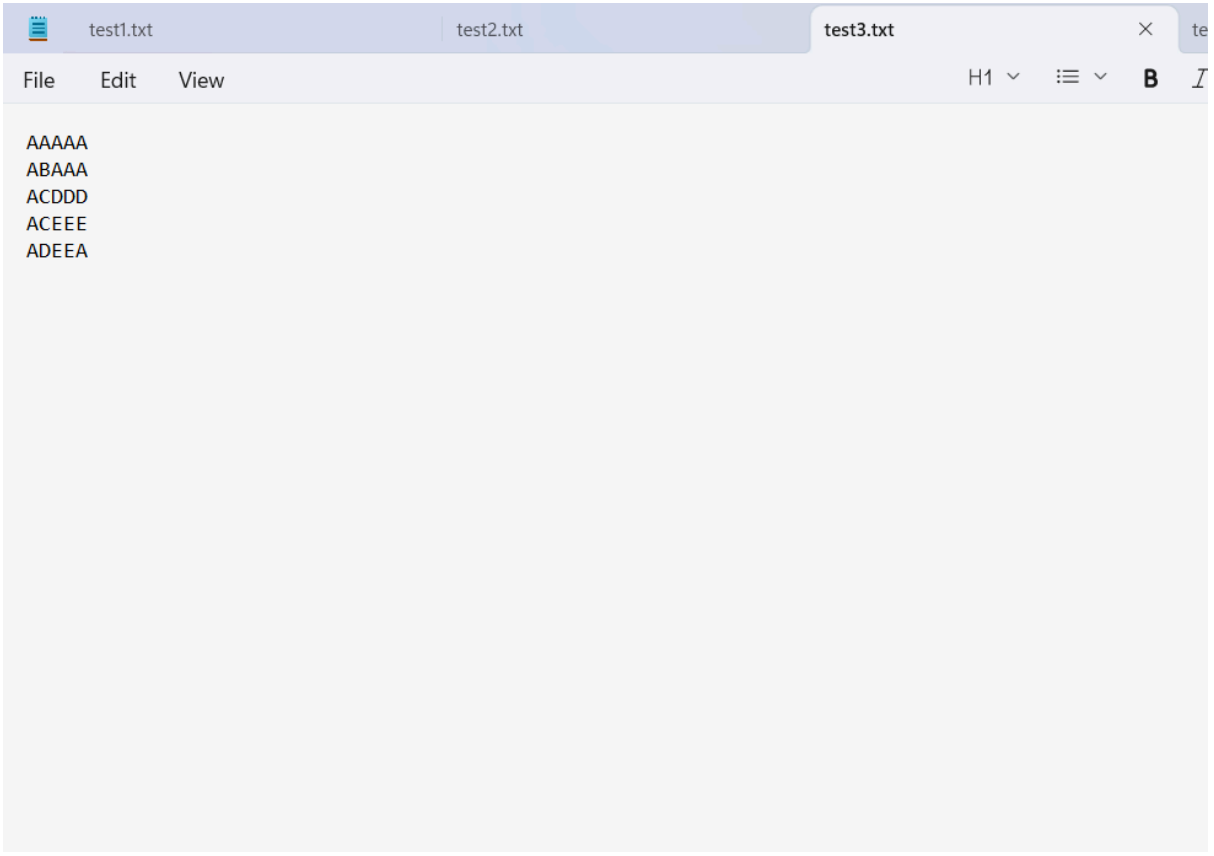
2.

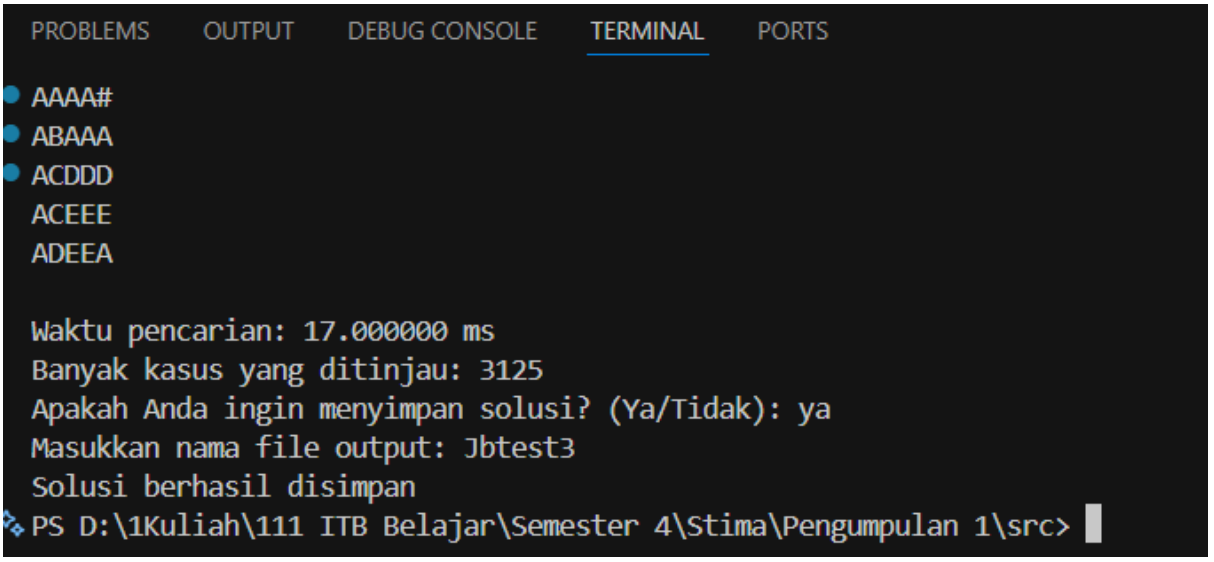

test1.txt
test2.txt
File
Edit
View
AAABBCCCB
DABBBBCBB
DAEEEBCBB
FFFEBBBBB
GFGEGGHHH
GFGGGGGHG
GGGIIIGHG
GGGGIGGGG
GGGGIGGGG


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
A#ABBCCCB
DABBBBCB#
DAEEEBCBB
FFFEBBBBB
GFGEGGHHH
GFGGGGGHG
GGGIIIGHG
GGGGIGGGG
GGG#IGGGG

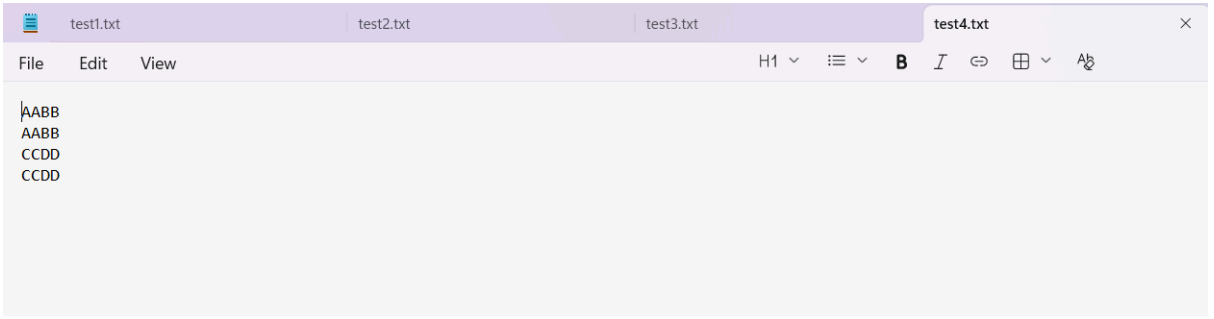

PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
AA#BBCCCB
DABBBB#BB
#AEEEBCBB
FFFEB#BBB
GFG#GGHHH
G#GGGGGHG
GGGIIIG#G
GGGG#GGGG
GGGGIGGG#
Waktu pencarian: 10558.000000 ms
Banyak kasus yang ditinjau: 115107525
Apakah Anda ingin menyimpan solusi? (Ya/Tidak):

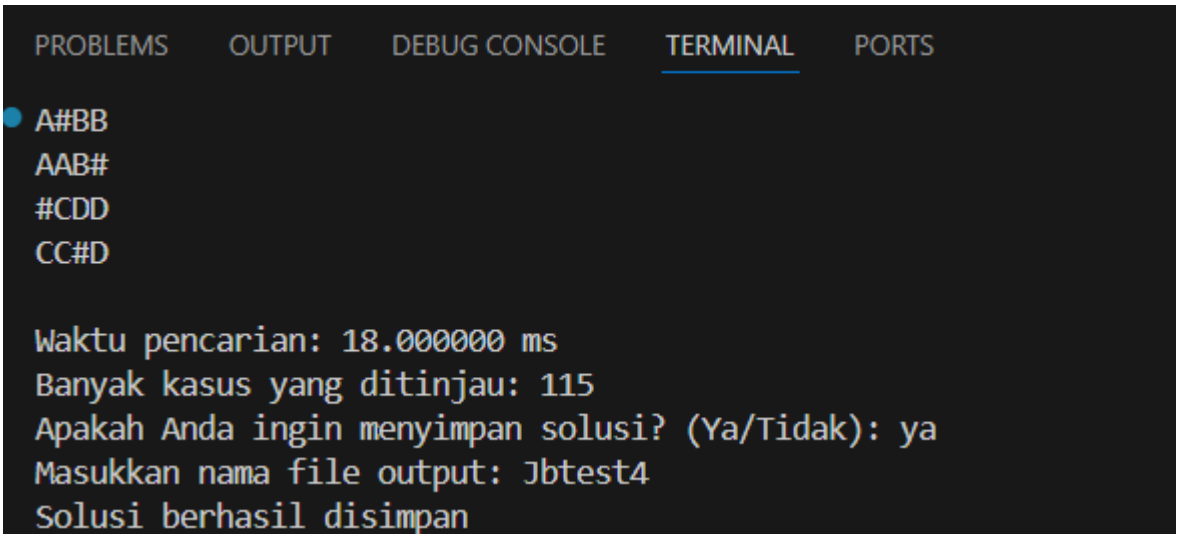
3.


test1.txt
test2.txt
test3.txt
File
Edit
View
H1
AAAAA
ABAAA
ACDDD
ACEEE
ADEEA

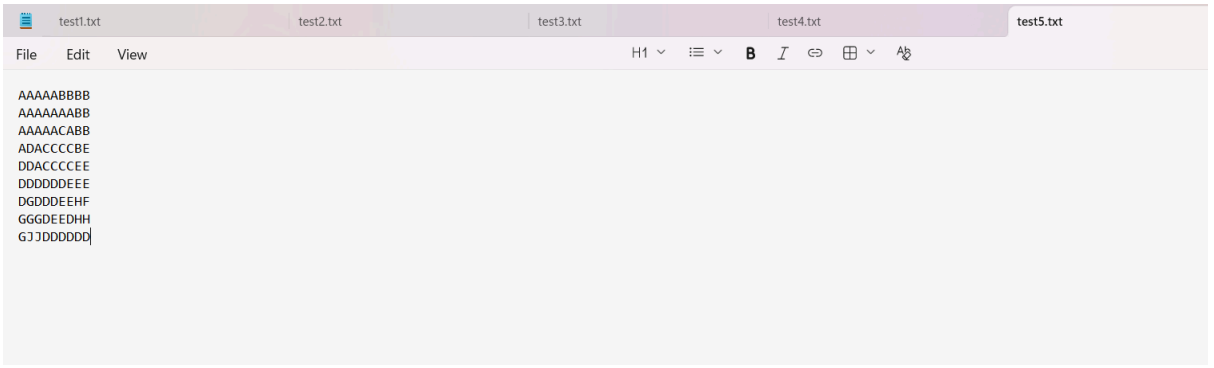

PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
AAAA#
ABAAA
ACDDD
ACEEE
ADEEA
Waktu pencarian: 17.000000 ms
Banyak kasus yang ditinjau: 3125
Apakah Anda ingin menyimpan solusi? (Ya/Tidak): ya
Masukkan nama file output: Jbtest3
Solusi berhasil disimpan
PS D:\1Kuliah\111 ITB Belajar\Semester 4\Stima\Pengumpulan 1\src>

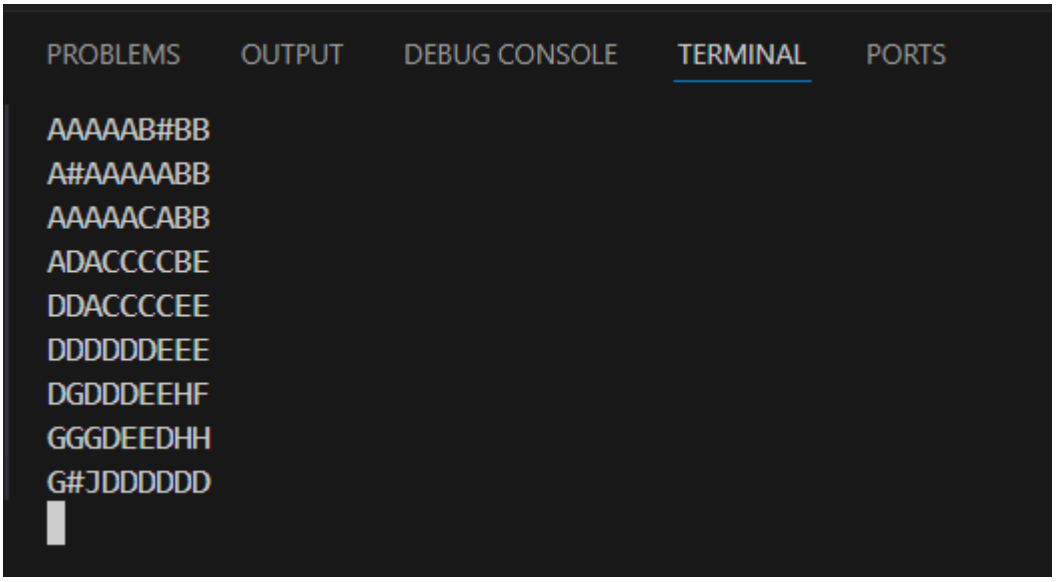
4.

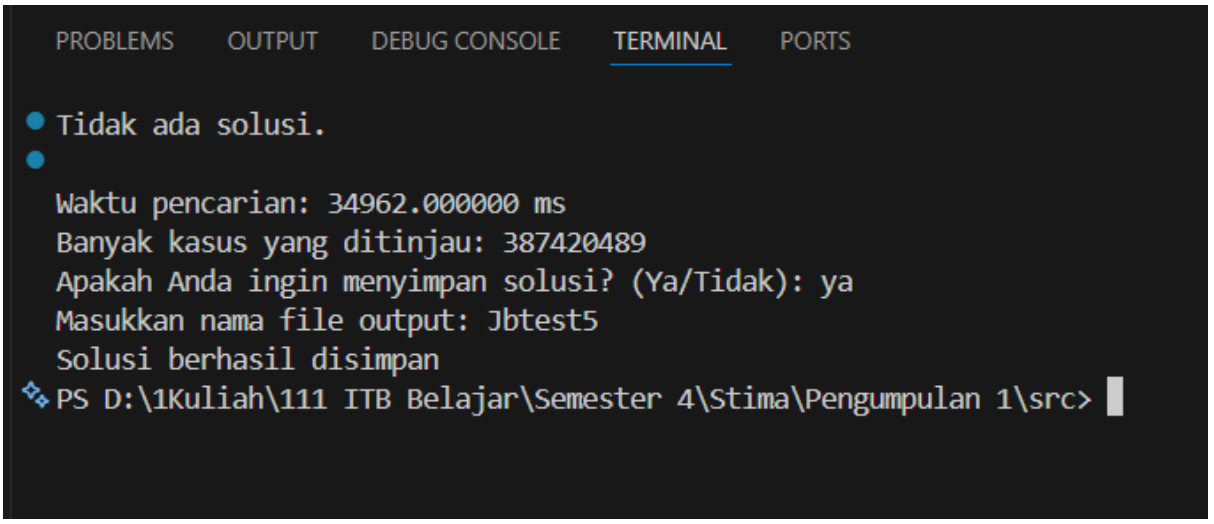

test1.txt
test2.txt
test3.txt
test4.txt
File
Edit
View
H1
AABB
AABB
CCDD
CCDD


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
A#BB
AAB#
#CDD
CC#D
Waktu pencarian: 18.000000 ms
Banyak kasus yang ditinjau: 115
Apakah Anda ingin menyimpan solusi? (Ya/Tidak): ya
Masukkan nama file output: Jbtest4
Solusi berhasil disimpan

5.


test1.txt
test2.txt
test3.txt
test4.txt
test5.txt
File
Edit
View
H1
AAAAABBBB
AAAAAAABB
AAAAACABB
ADACCCCBE
DDACCCCEE
DDDDDDEEE
DGDDDEEHF
GGGDEEDHH
GJJDDDDDD



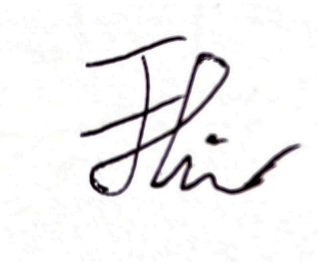


IV. Pranala Repository

https://github.com/Defaro123/Tucil1_13524046.git

V. Pernyataan Tidak Melakukan Kecurangan

Tugas ini disusun sepenuhnya tanpa bantuan kecerdasan buatan (Generative AI), melainkan hasil pemikiran dan analisis mandiri.



Farrell Limjaya

No	Poin	Ya	Tidak
1	Program berhasil dikompilasi tanpa kesalahan	✓	
2	Program berhasil dijalankan	✓	
3	Solusi yang diberikan program benar dan mematuhi aturan permainan	✓	
4	Program dapat membaca masukan berkas .txt serta menyimpan solusi dalam berkas .txt	✓	
5	Program memiliki Graphical User Interface (GUI)		✓
6	Program dapat menyimpan solusi dalam bentuk file gamba		✓