Generate Doppler Effect Pattern Matrix with Recursive

Fakultas Ilmu Komputer

Universitas Sriwijaya

Nama : Muhammad Irfan Triananto Putra

NIM : 09021281621046

1. About Program

Cara kerja :

Program akan meminta inputan ukuran matriks

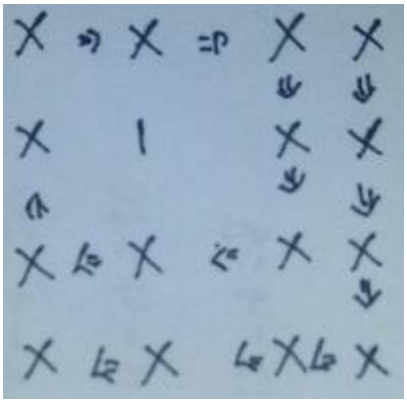
Program akan meminta inputan lokasi baris dan lokasi kolom untuk titik awal

Program akan mengeluarkan output berupa matriks dengan pola doppler effect

Program ini dibuat dengan metode **recursive**, yang setiap perulangannya dilakukan **traversal** secara N^2

Traversal berlangsung dari bagian atas -> bagian kanan -> bagian bawah -> bagian kiri.

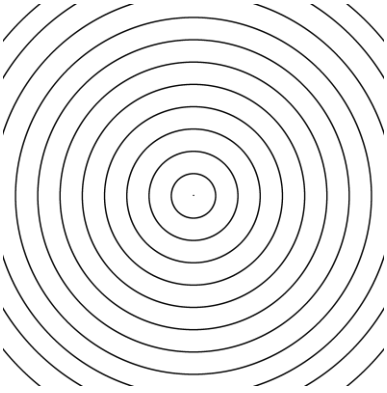
Ilustrasi traversal pada matrix 4x4 dengan lokasi di titik [1,1]



Diluar titik awal, ada matrix 3x3, traversal dilakukan dari titik 0,0 kemudian traverse ke tepi kanan, tepi bawah, dan terakhir sampai di tepi kiri akhir yaitu 0,1

Pada traverse kedua, dimulai dari matrix 5x5 karena tidak ada bagian atas, traverse di skip dan dilanjutkan ke traverse tepi kanan, dan berakhir di tepi bawah karena tepi kiri tidak ada

Ilustrasi doppler effect



2. Screenshot & Penjelasan

2.1 Procedure prototype

```
void input(int *baris, int *kolom, int *a, int *b);
void bikin_m1(int M[30][30], int baris, int kolom, int a, int b);
void bikin_m2(int M[30][30], int baris, int kolom, int a, int b, int nilai);
void cetak_m(int M[30][30],int baris, int kolom);
```

Pada kodingan ini, saya mendeklarasikan procedure prototype agar semua prosedur bisa dilihat lebih mudah oleh orang lain.

2.2 main()

```
int main()
{
    int M[30][30], baris, kolom, a, b;

    input(&baris,&kolom, &a, &b);
    bikin_m1(M, baris,kolom, a-1, b-1);
    bikin_m2(M, baris,kolom, a-1, b-1, 2);
    cetak_m(M, baris, kolom);
}
```

Pada main(), hanya ada deklarasi dan prosedur, tidak ada proses seperti perulangan dan perkondisian yang terjadi. Detail tiap prosedur pada main() akan dijelaskan pada poinnya masing-masing.

nb: pada parameter, a dan b dikurang 1, ini dilakukan agar nilai yang masuk berupa **index lokasi, bukan lokasi titik.**

2.3 input

```
void input(int *baris, int *kolom, int *a, int *b)
{
    printf("baris: ");
    scanf("%d",&*baris);
    printf("kolom: ");
    scanf("%d",&*kolom);
    printf("lokasi baris: ");
    scanf("%d",&*a);
    printf("lokasi kolom: ");
    scanf("%d",&*b);
}
```

Pada prosedur ini, program diperintahkan untuk menanyakan inputan dari user, berupa ukuran matriks, dan a (baris) & b (kolom) untuk lokasi titik awal.

2.4 Membuat Matriks 0

```
void bikin_m1(int M[30][30], int baris, int kolom, int a, int b)
{
    for(int i=0;i<baris;i++)
    {
        for(int j=0;j<kolom;j++)
        {
            M[i][j] = 0;

            if(i == a && j == b)
            {
                M[i][j] = 1;
            }
        }
    }
}
```

Pada prosedur ini, matriks akan dibuat berdasarkan data yang diberikan, diberi nilai 0 untuk semua elemen, dan nilai 1 untuk lokasi awal.

2.5 Membuat Matriks Pola Doppler Effect

2.5.1 Deklarasi

```
void bikin_m2(int M[30][30], int baris, int kolom, int a, int b, int nilai)
{
    int i, T, B, L, R;
```

Parameter yang diberikan adalah **matriks**, **ukuran matriks**, **baris titik**, **kolom titik**, **nilai**.

Dan variabel yang di deklarasikan adalah i, T, B, L, R.

Nilai adalah nilai selanjutnya dari titik awal, karena titik awal bernilai 1, maka **angka selanjutnya** adalah 2, oleh karena itu pada main() diberikan nilai 2 di parameter-nya.

2.5.1.1 Ilustrasi “Angka Selanjutnya”

matriks 3x3, lokasi titik [0,0] (baris ke-1 dan kolom ke-1)

1 2 X

2 2 X

X X X

Variabel nilai dibuat dengan dengan **tujuan untuk menyimpan nilai sebelumnya**, karena prosedur ini dilakukan secara **rekursif**, **jika nilai tidak disimpan**, maka nilai berikutnya akan tetap **konstan** yaitu 2.

i = variabel perulangan

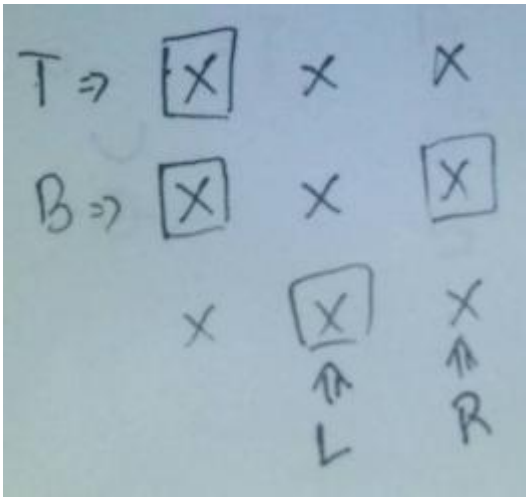
T = singkatan dari Top, variabel yang digunakan untuk menandakan posisi atas

B = singkatan dari Bottom, variabel yang digunakan untuk menandakan posisi bawah

L = singkatan dari Left, variabel yang digunakan untuk menandakan posisi kiri

R = singkatan dari Right, variabel yang digunakan untuk menandakan posisi kanan

2.5.1.2 Contoh lokasi T,B,L,R pada matriks 3x3



Pada ilustrasi 2.5.1.1, menandakan posisi T = 1. B = 2. L = 2 . R = 3.

Detail penggunaan TBLR ini akan dijelaskan di subbab 2.5.2 sampai 2.5.4

2.5.2 Top-Bottom-Left-Right

```
T = a - (nilai-1);
B = a + (nilai-1);
L = b - (nilai-1);
R = b + (nilai-1);
```

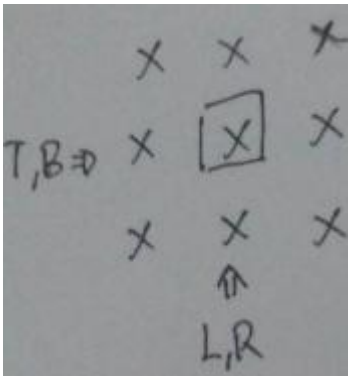
Pada ilustrasi 2.5.1.2 :

Variabel T dan B menunjuk bagian baris sedangkan,

Variabel L dan R menunjuk bagian kolom

(nilai-1) digunakan sebagai **counter**, menghitung jumlah traversal, karena nilai dimulai dari 2, maka (nilai-1) adalah 1. Jadi, **hasil (nilai-1) adalah iterasi traversal**. Pada subbab 2.5.5 terdapat **nilai**++, nilai ini akan digunakan lagi pada rekursif berikutnya, yang kemudian akan terjadi traversal ke-2 dan seterusnya. Detail di ilustrasi 2.5.2.1 dan 2.5.2.2

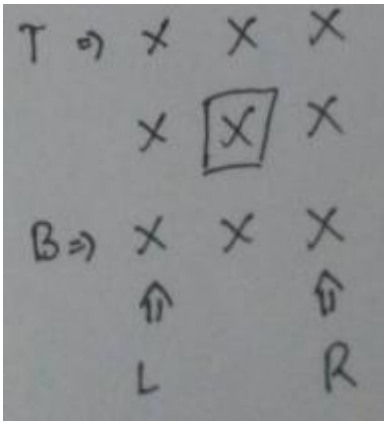
2.5.2.1 Inisialisasi



X yang diberi kotak, merupakan titik awal dengan index [1,1]

Ini merupakan posisi awal T,B menunjuk baris 2, dan L,R menunjuk kolom 2

2.5.2.2 Traversal ke-1



Sesuai dengan rumusnya, yaitu T = a – (nilai-1)

a = **index baris lokasi**, yaitu 1.

nilai = 2 (nilai yang diberikan di parameter di main())

jadi T = 1-(2-1) = 0, begitupula yang lainnya

B = 1+(2-1) = 2 (hasil L dengan R kebetulan sama dengan hasil T dengan B, yaitu 0 dan 2)

L dan R menggunakan rumus berdasarkan **index kolom lokasi.**

Detail perjalanan traversalnya dijelaskan di subbab 2.5.4

2.5.3 Melewati Batas

```
if(T<0)T=0;
if(B>baris-1)B=baris-1;
if(L<0)L=0;
if(R>kolom-1)R=kolom-1;
```

Pada subbab 2.5.2 dijelaskan bahwa nilai T dan L akan berkurang setiap rekursif terjadi, sebaliknya nilai B dan R akan bertambah.

Oleh karena itu diberikan **kondisi batas** agak tidak keluar dari matriks yang ditentukan

Karena T dan L selalu **berkurang** setiap rekursif terjadi, maka jika nilai mereka **kurang dari 0** (index terkecil matrks adalah 0), maka nilai mereka akan tetap menjadi 0.

Karena B dan R selalu **bertambah** setiap rekursif terjadi, maka jika nilai mereka **lebih dari batas baris atau kolom matriks**, maka nilai mereka akan tetap sesuai ukuran yang ditentukan

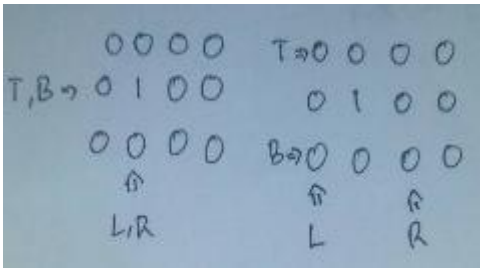
B akan sesuai baris-1 (karena B dipakai sebagai index, jadi dikurang 1)

R akan sesuai kolom-1 (karena R dipakai sebagai index, jadi dikurang 1)

2.5.4 Traversal

```
for(i = L;i<=R;i++)
    if(M[T][i] == 0)M[T][i] = nilai;
for(i = T;i<=B;i++)
    if(M[i][R] == 0)M[i][R] = nilai;
for(i = R;i>=L;i--)
    if(M[B][i] == 0)M[B][i] = nilai;
for(i = B;i>=T;i--)
    if(M[i][L] == 0)M[i][L] = nilai;
```

2.5.4.1 Inisialisasi Matriks 3x4, Lokasi [1,1] dan Traverse ke-1



Dilakukan traverse pertama, (T dan L berkurang, B dan R bertambah).

Pseudocode-nya:

for i <- L to R (karena perulangan sepanjang L sampai R)

    M[T][i] <- nilai (karena T konstan sepanjang perulangan)

for i <- T to B

    M[i][R] <- nilai

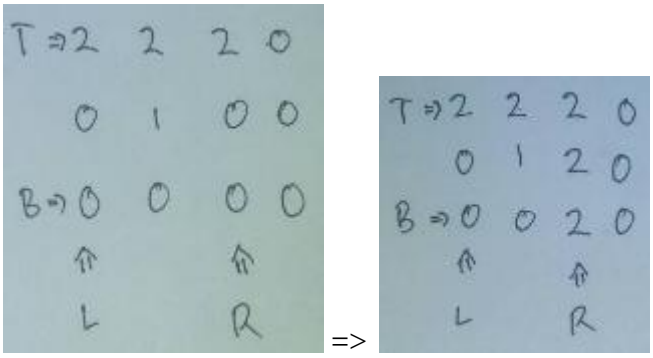
for i <- R down to L

    M[B][i] <- nilai

for i <- B down to T

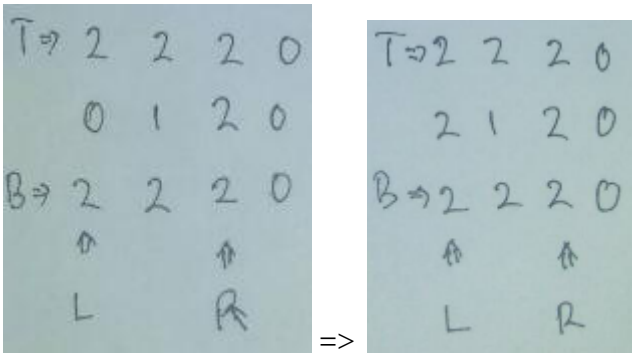
    M[i][L] <- nilai

2.5.4.2 Traverse 1, Top - Right



Ilustrasi untuk 2 perulangan pertama

2.5.4.3 Traverse 1, Bottom - Left



Ilustrasi untuk 2 perulangan terakhir, untuk kolom ke-4 tetap 0, karena **belum ada perulangan secara rekursif**.

2.5.5 Recursive and Return

```
nilai++;
if(T==0 && B==baris-1 && L==0 && R==kolom-1)return;
bikin_m2(M, baris, kolom, a, b, nilai);
```

Return digunakan ketika kondisi sudah terpenuhi, yaitu traversal ke seluruh elemen matriks telah selesai.

Karena T dan L selalu berkurang, maka kondisi akhirnya adalah 0 (titik awal matriks)

Karena B dan R selalu bertambah, maka kondisi akhirnya adalah ukuran matriks-1 (index), dimana B berakhir sesuai index baris dan R sesuai index kolom.

bikin_m2 merupakan prosedur yang rekursif (memanggil dirinya sendiri), digunakan agar traversal berikutnya terjadi sampai semua elemen tidak ada yang 0.

nb: Pada subbab 2.5.2 dijelaskan kegunaan nilai++.

2.6 Cetak

```
void cetak_m(int M[30][30],int baris, int kolom)
{
    for(int i=0;i<baris;i++)
    {
        for(int j=0; j<kolom; j++)
        {
            printf("%3d ",M[i][j]);
        }
        printf("\n");
    }
}
```

Prosedur ini digunakan untuk mencetak isi dari matriks yang berisi pola Doppler Effect.