### TÜRKİYE CUMHURİYETİ YILDIZ TEKNİK ÜNİVERSİTESİ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ



# ALGORİTMA ANALİZİ DÖNEM PROJESİ

Öğrenci No: 19011010 Öğrenci Adı Soyadı: Beyda Güler

Öğrenci e-posta: beyda.guler@std.yildiz.edu.tr

Ders / Grup: BLM3021-Algoritma Analizi / Grup:1

Ders Yürütücüsü: Mine Elif KARSLIGİL Ocak , 2023

## İçindekiler

YÖNTEM	3
Problem	3
Çözüm	3
UYGULAMA	3
MENU	3
NORMAL MOD	4
DETAYLI MOD	5
SONUÇ	7
RotateTheRow() Fonksiyonunun Zaman ve Yer Karmaşıklığı	7
IsSafe() Fonksiyonunun Zaman ve Yer Karmaşıklığı	7
Solve() Fonksiyonunun Zaman ve Yer Karmaşıklığı	8
VIDEO LINKI:	

#### YÖNTEM

#### Problem

Kullanıcıdan alınan N sayısı kadar renkle oluşturulan NxN'lik matrisin her kolonunda renkler bir defa tekrar etmelidir. Bunu sağlamak için backtracking mantığı kullanılması istenmektedir.

#### Çözüm

```
typedef struct color{
   char colorName[20];
   int colorNo;
}COLOR;
```

#### COLOR\*\* CreateMatrix(COLOR\*\* matrix, int N)

Kullanıcının girdileriyle NxN'lik bir matris oluşturulur. Matrisin her gözünde bir renk ve o renk için oluşturulan renk kodu bulunmaktadır.

• int IsSafe(COLOR\*\* matrix, int row, int N)

Fonksiyona girdi olarak verilen 'row' parametresi o anda bulunduğumuz satırı göstermektedir. IsSafe fonksiyonu, bu satırı üstündeki diğer bütün satırlarla karşılaştırarak çakışma olup olmadığını (safe / not safe) tesit etmektedir. Safe ise 1, not safe ise 0 döndürmektedir.

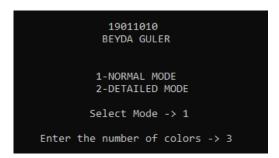
#### COLOR\* RotateTheRow(COLOR\* row,int N)

IsSafe fonksiyonundan not safe olarak dönen satırı safe hale getirmek için rotate yapılmasını sağlayan fonksiyondur. Kaydırma işlemi dairesel bir şekilde gerçekleşmektedir.

#### • int Solve(COLOR\*\* matrix, int N, int row,int mode)

IsSafe ve RotateTheRow fonksiyonlarını kullanarak recursive bir şekilde her satırı safe hale getirmeye çalışmaktadır. Backtracking mantığını kullanır. Yani eğer 1. Ve 2. Satırlar safe durumda fakat 3. Satır not safe durumdaysa bir üstteki satıraçıkılır, yeniden rotate edilir. Rotate'ten sonra 2 hala safe ise 3. Satıra geri dönülür ve bu durum için ihtimaller denenir.

### UYGULAMA MENU



- Normal Mod'da sadece girilen matris için eğer çözüm varsa rotate edilmiş matris, eğer çözüm yoksa NO SOLUTION çıktısı verilir.
- Detaylı Mod'da bütün rotation adımları çözüm olsa da olmasa da gösterilir.

#### **NORMAL MOD**

\*\*\* Her harf bir renk olarak düşünülmüştür.

```
----- NORMAL MODE -----

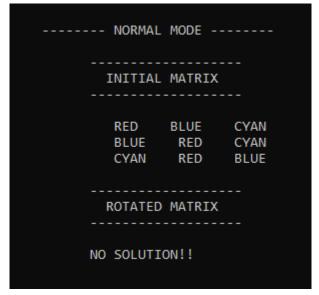
INITIAL MATRIX

a s d f
s d f a
s a f d
f d s a

ROTATED MATRIX

a s d f
d f a s
s a f d
f d s a
```

Şekil 1 - Sonuç elde edildiği durum



Şekil 2 - Sonuç elde edilemeyen durum

```
Select Mode -> 2

Enter the number of colors -> 3

***1st row***

1st column ---> BLUE

2st column ---> RED

***2st row***

1st column ---> BLUE

2st column ---> BLUE

2st column ---> BLUE

2st column ---> BLUE

2st column ---> BLACK

3st column ---> RED

***3st row***

1st column ---> BLUE

2st column ---> BLUE
```

Şekil 3 - Input

```
----- DETAILED MODE -----
          BLACK
   RED
                  BLACK
                  BLACK
          BLACK
   RED
                  BLACK
   BLACK
   BLUE
          BLACK
                   RED
   RED
                  BLACK
   RED
          BLACK
                  BLUE
   BLUE
          BLACK
                   RED
          BLUE
                  BLACK
   RED
   BLUE
          RED
                  BLACK
   BLUE
          BLACK
                   RED
   BLACK
                  BLUE
           RED
          RED
                  BLACK
   BLUE
          BLACK
                   RED
   BLACK
           RED
                   BLUE
   BLACK
  BLUE
BLACK
           BLACK
                   RED
          BLACK
          BLACK
   BLACK
                  BLUE
          RED
   BLUE
                  BLACK
   BLUE
           BLACK
   BLUE
           BLACK
                    RED
           RED
                  BLACK
NO SOLUTION!!
```

Şekil 4 - Rotation adımları ve sonuç

```
Select Mode -> 2
Enter the number of colors -> 4
          ***1st row***
        1st column ---> BLUE
        2st column ---> RED
        3st column ---> WHITE
        4st column ---> CYAN
          ***2st row***
       1st column ---> BLUE
        2st column ---> RED
        3st column ---> WHITE
       4st column ---> CYAN
          ***3st row***
       1st column ---> BLUE
        2st column ---> RED
        3st column ---> WHITE
       4st column ---> CYAN
          ***4st row***
        1st column ---> BLUE
        2st column ---> RED
        3st column ---> WHITE
        4st column ---> CYAN
```

Şekil 5 - Input

```
----- DETAILED MODE -----
                 WHITE
 BLUE
         RED
                         CYAN
         BLUE
                 RED
                         WHITE
 BLUE
         RED
                 WHITE
                         CYAN
                 WHITE
                          CYAN
         RED
                 WHITE
 BLUE
         RED
                         CYAN
 CYAN
                 RED
                         WHITE
 CYAN
         BLUE
                 RED
                         WHITE
                 WHITE
                          CYAN
 BLUE
         RED
                 WHITE
                          CYAN
                         WHITE
RED
 CYAN
         BLUE
          CYAN
 WHITE
         RED
                 WHITE
                          CYAN
 BLUE
                 WHITE
                          CYAN
                         WHITE
 CYAN
                  RED
 WHITE
          CYAN
                          RED
 CYAN
         BLUE
                  RED
                         WHITE
          RED
                 WHITE
                          CYAN
 CYAN
                         WHITE
 WHITE
          CYAN
                           RED
 WHITE
          CYAN
                           RED
          RED
                 WHITE
                          CYAN
 CYAN
                  RED
                         WHITE
 WHITE
          CYAN
                          RED
 RED
         WHITE
                  CYAN
                          BLUE
```

Şekil 6 - Rotation adımları ve sonuç

#### CreateMatrix() Fonksiyonunun Zaman ve Yer Karmaşıklığı

```
COLOR** CreateMatrix(COLOR** matrix , int N){
    int i,j,k;
    for(i = 0 ; i < N ; i++){
        printf("\n\t\t\t ***%dst row***\n",i+1);
        for(j = 0 ; j < N ; j++){
    printf("\n\t\t\t%dst column ---> ",j+1);
            scanf("%s",matrix[i][j].colorName);
            if(i == 0)
                 matrix[i][j].colorNo = j+1;
            else{
                 k=0;
                 while(strcmp(matrix[0][k].colorName,matrix[i][j].colorName) != 0)
                 matrix[i][j].colorNo = matrix[0][k].colorNo;
    return matrix;
}
                                   Zaman : O(N^2)
                                     Yer: O(N^2)
```

#### RotateTheRow() Fonksiyonunun Zaman ve Yer Karmaşıklığı

#### IsSafe() Fonksiyonunun Zaman ve Yer Karmaşıklığı

Yer :  $O(N^2)$ 

#### Solve() Fonksiyonunun Zaman ve Yer Karmaşıklığı

```
int Solve(COLOR** matrix, int N, int row,int mode){
    int i;
    for(i=0;i<N;i++){</pre>
        if(IsSafe(matrix,row,N)){
                                                           // IF IT IS SAFE
            if(row!=N-1){
                if(Solve(matrix,N,row+1,mode))
                    return 1;
                else{
                    matrix[row]=RotateTheRow(matrix[row],N);
                    if(mode==2)
                        PrintMatrix(matrix,N);
            }else
                return 1;
                                                          // IF IT IS NOT SAFE
        }else{
            matrix[row]=RotateTheRow(matrix[row],N);
            if(mode==2)
                PrintMatrix(matrix,N);
    return 0;
}
```

Zaman : O(N!)Yer :  $O(N^2)$ 

VIDEO LINKI:

https://youtu.be/LyNcWriZu9Y