Algoritma Analizi Proje Ödevi

1-Proje Kapsamı

Projede amacımız bize girdi olarak verilen NxN lik bir matriste bulunan N adet renk için her sütunda aynı renkler bulunanbilecek şekilde bir çıktı olabilme olasılığına bakan ve bunu backtracking yöntemi ile yapan bir algoritma geliştirmek.Backtracking algoritmasını kullanmamızın sebebi doğru dizilimi bulurken olabilicek en az denemeyle bulmak.

2-)Girdi Almak

Matrisdeki renk verileri ve matrisin boyutunu kullanıcıdan almamız istenmiş bu yüzden öncellikle matris verisini kullanıcıdan nasıl alacağımızı düşünelim.Mantıken önce matrisin kaça kaçlık olduğunu kullanıcıdan almamız gerekiyor.Aşağıda da görüldüğü üzere kullanıcıyıda bilgilendiricek şekilde n sayısı 3 ve 8 arasında olucak şekilde N sayısı kullanıcıdan alınıyor.Burada önemli olan eğer N sayısı 3 ve 8 arasında değilse diye kontrol yapıp kullanıcıyı uyarıp tekrar girdi alıyoruz.

```
Satir ve sutun sayisi icin N sayisini giriniz
N sayisi 3 <= N <= 8 olmak zorundadir!!!!!!!!!
N:
```

Aşağıda görüldüğü gibi N sayısı 9 girildiği için kullanıcıyı uyarıyoruz ve tekar girdi istiyoruz.

```
Satir ve sutun sayisi icin N sayisini giriniz
N sayisi 3 <= N <= 8 olmak zorundadir!!!!!!!!!
N:9

9 sayisi 3 ile 8 arasinda degil lutfen 3 <= N <= 8 sartini saglayin...
Satir ve sutun sayisi icin N sayisini giriniz
N sayisi 3 <= N <= 8 olmak zorundadir!!!!!!!!!
N:
```

Daha sonra kullanıcıdan string olarak değer almak hem bizim için hem kullanıcı için zor olduğu için bir renk ,index listesi paylaşıp kullanıcıdan renk isimleri yerine renklerin numarasını alıyoruz.

```
Satir Sutun verileri icin girmek istediginiz renkleri giriniz!0:Sari
1:Kirmizi
2:Yesil
3:Turuncu
4:Mavi
5:Mor
6:Lacivert
7:Lila
satir:0  sutun:0 renk numarasini girin:
```

Daha sonra alınan N sayısına göre allocate edilmiş int dizimize satır ve sütün numaraları verilmiş şekilde tek tek verileri alıyoruz.Böylece kullanıcıdan veri alma kısmımız bitmiş oluyor

```
satir:0  sutun:0 renk numarasini girin:1

satir:0  sutun:1 renk numarasini girin:2

satir:0  sutun:2 renk numarasini girin:3

satir:1  sutun:0 renk numarasini girin:1

satir:1  sutun:1 renk numarasini girin:2

satir:1  sutun:2 renk numarasini girin:3

satir:2  sutun:0 renk numarasini girin:1

satir:2  sutun:1 renk numarasini girin:2

satir:2  sutun:2 renk numarasini girin:3
```

3-Kod Açıklamaları

Kullanıcıdan verileri aldıktan sonra veriler acaba doğru alındımı alınmadımı diye kontrol etmek için print_matris adlı fonksiyonu çağırıyoruz.Bu fonksiyon aldığımız index verilerine göre bize renk tablosunu yazdırıyor.

```
Kirmizi-Yesil-Turuncu-
Kirmizi-Yesil-Turuncu-
Kirmizi-Yesil-Turuncu-

Kirmizi-Yesil-Turuncu-
Turuncu-Kirmizi-Yesil-
Kirmizi-Yesil-Turuncu-

Kirmizi-Yesil-Turuncu-
Turuncu-Kirmizi-Yesil-
Turuncu-Kirmizi-Yesil-
```

Sütunlarda veriler aynı olduğunda satırları döndürmemiz gerekicek bu yüzden satır döndüren bir fonksiyon yazmamız gerekiyor.Bu yüzden turn_line adlı fonksiyonu yazdık.Bu fonksiyon verilen n değeri ve sütun değeri için o sütündaki verileri kaydırma işlemi yapıyor başta bakıldığında kolay gibi gözüksede bu fonksiyonu yazarken biraz zorlandım çünkü değer yazarken üzerine yazdığımız indexteki verileride kayıp etmememiz gerekiyor.Bu yüzden tersten bir for işlemi döndürdüm ve sonuca vardım.

```
void turn_line(int **dizi,int satir,int n){
    int k,sayi=n-1;
    k=dizi[satir][sayi];
    sayi--;
    for (int i=sayi;i>=0;i--){

        dizi[satir][i+1]=dizi[satir][i];
        sayi--;
    }

    dizi[satir][0]=k;

}
```

Daha sonra sutundaki verilerin aynı olmamasını kontrol etmek için satir_compare adlı bir fonksiyon yazdım. Bu fonksiyon verilen n değeri için döngüye girip 2 sutun için karşılşatırma işlemi yapıyor.Bulunucak ilk eşitlikte direk fonksiyonu bitirip false dönüyoruz en az karşılaştırma yapmak için eğer tüm değerler farklıysa for bitiminde true değeri dönülüyor.

```
bool satir_compare(int **dizi,int sutun1,int sutun2,int n){
    for(int i=0;i<n;i++){
        if(dizi[sutun1][i]==dizi[sutun2][i]){
            return false;
        }
    }

    return true;
}
```

Daha sonra ise 2 3 satır için karşılaştırma işlemleri kolay gözüksede iş derinleştikçe ve 6,7 gibi rakamlar gidildikçe kendinden önceki tüm satırlar için kontrol işlemi yapılması gerekiyor bu yüzden de all_sutun compare diye bir fonksiyon yazıyoruz.Bu fonksiyon temelde satir_Compare fonksiyonunu kullanıyor ama bunu kendinden önceki her satır için yapıyor.

```
bool all_satir_compare(int **dizi,int sutun,int n){
    for ( int i = 0;i < sutun;i++){
        if(!satir_compare(dizi,i,sutun,n)){
            return false;
        }

    }
    return true;
}
```

Daha sonra aşağıdaki yapıyı kurarak her veri için karşılaştırma yaparak olası çözüm var mı yok mu kontrol ediyoruz.

```
int k=0,a,c=0;
for(int i=1;i<n;i++){
    a=0;
    while(!all_satir_compare(dizi,i,n) && a<n+1){

        turn_line(dizi,i,n);
        printf("\n");
        print_matris(dizi,n);
        a++;
    }
    if(a==n+1){
        printf("Cozum yok malesef!!!!!!!!!!!!!!!");
        c=1;
        break;
    }


    }
    if(c==0){
        printf("Cozum var!!!!!!!!!!!!!!!");
}
```

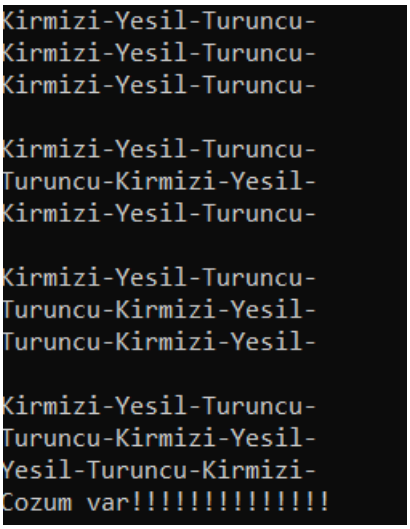
N=4 İÇİN ALGORİTMA ÇIKTISI.

1.MATİS GİRİŞ YAPILAN MATRİS SON MATRİS İSE CÖZÜLMÜŞ MATRİS

```
Kirmizi-Yesil-Turuncu-Mavi-
Kirmizi-Yesil-Turuncu-Mavi-
Kirmizi-Yesil-Turuncu-Mavi-
Kirmizi-Yesil-Turuncu-Mavi-

Kirmizi-Yesil-Turuncu-Mavi-
Mavi-Kirmizi-Yesil-Turuncu-
Kirmizi-Yesil-Turuncu-Mavi-
Kirmizi-Yesil-Turuncu-Mavi-

Kirmizi-Yesil-Turuncu-Mavi-
Mavi-Kirmizi-Yesil-Turuncu-
Mavi-Kirmizi-Yesil-Turuncu-
Kirmizi-Yesil-Turuncu-Mavi-

Kirmizi-Yesil-Turuncu-Mavi-
Mavi-Kirmizi-Yesil-Turuncu-
Turuncu-Mavi-Kirmizi-Yesil-
Kirmizi-Yesil-Turuncu-Mavi-

Kirmizi-Yesil-Turuncu-Mavi-
Mavi-Kirmizi-Yesil-Turuncu-
Turuncu-Mavi-Kirmizi-Yesil-
Mavi-Kirmizi-Yesil-Turuncu-

Kirmizi-Yesil-Turuncu-Mavi-
Mavi-Kirmizi-Yesil-Turuncu-
Turuncu-Mavi-Kirmizi-Yesil-
Turuncu-Mavi-Kirmizi-Yesil-

Kirmizi-Yesil-Turuncu-Mavi-
Mavi-Kirmizi-Yesil-Turuncu-
Turuncu-Mavi-Kirmizi-Yesil-
Yesil-Turuncu-Mavi-Kirmizi-
Cozum var!!!!!!!!!!!!!
```

N=3 İÇİN ÇIKTI

1.MATİS GİRİŞ YAPILAN MATRİS SON MATRİS İSE CÖZÜLMÜŞ MATRİS



N=4 İÇİN ÇIKTI

1.MATİS GİRİŞ YAPILAN MATRİS SON MATRİS İSE CÖZÜLEMEMİŞ MATRİS

```
Kirmizi-Yesil-Turuncu-Mavi-
Kirmizi-Yesil-Mavi-Turuncu-
Kirmizi-Yesil-Mavi-Turuncu-
Kirmizi-Yesil-Turuncu-Mavi-

Kirmizi-Yesil-Turuncu-Mavi-
Turuncu-Kirmizi-Yesil-Mavi-
Kirmizi-Yesil-Mavi-Turuncu-
Kirmizi-Yesil-Turuncu-Mavi-

Kirmizi-Yesil-Turuncu-Mavi-
Mavi-Turuncu-Kirmizi-Yesil-
Kirmizi-Yesil-Mavi-Turuncu-
Kirmizi-Yesil-Turuncu-Mavi-

Kirmizi-Yesil-Turuncu-Mavi-
Mavi-Turuncu-Kirmizi-Yesil-
Turuncu-Kirmizi-Yesil-Mavi-
Kirmizi-Yesil-Turuncu-Mavi-

Kirmizi-Yesil-Turuncu-Mavi-
Mavi-Turuncu-Kirmizi-Yesil-
Mavi-Turuncu-Kirmizi-Yesil-
Kirmizi-Yesil-Turuncu-Mavi-

Kirmizi-Yesil-Turuncu-Mavi-
Mavi-Turuncu-Kirmizi-Yesil-
Yesil-Mavi-Turuncu-Kirmizi-
Kirmizi-Yesil-Turuncu-Mavi-

Kirmizi-Yesil-Turuncu-Mavi-
Mavi-Turuncu-Kirmizi-Yesil-
Kirmizi-Yesil-Mavi-Turuncu-
Kirmizi-Yesil-Turuncu-Mavi-

Kirmizi-Yesil-Turuncu-Mavi-
Mavi-Turuncu-Kirmizi-Yesil-
Turuncu-Kirmizi-Yesil-Mavi-
Kirmizi-Yesil-Turuncu-Mavi-
Cozum yok malesef!!!!!!!!!!!!!
```

FONKSİYONLAR	KARMAŞIKLIK
turn_line(int **dizi,int satir,int n)	O(N)
satir_compare(int **dizi,int sutun1,int sutun2,int n)	O(N)
all_satir_compare(int **dizi,int sutun,int n)	O(N^2)
print_matris(int **dizi,int n)	O(N)