2020-2021 Güz Yarıyılı
Algoritma Analizi
Ödev – 4

Ders Yürütücüleri
Doç. Dr. M. Elif KARSLIGİL
Dr. Öğr. Üyesi M. Amaç GÜVENSAN

Burak Boz

18011706

Konu: Backtracking

Problem: NxN'lik bir matris görünümündeki oyun tahtasında her satırda aynı N renk farklı sıra ile yer almaktadır. Bir satırdaki renklerin sıralanışı, renkler sağa doğru kaydırılarak değiştirilebilmektedir. Örneğin satırdaki renkler sırası ile kırmızı, mavi, yeşil, mor ise satır 1 defa sağa kaydırıldığında yeni sıralama mor, kırmızı, mavi, yeşil olur. Bir defa daha sağa kaydırılırsa yeşil, mor, kırmızı, mavi elde edilir. Sonuç matrisinde her sütunda her renkten sadece 1 tane olacak şekilde satırları **geri-izleme(backtracking)** yöntemi ile **rekürsif olarak** düzenleyen algoritmayı tasarlayınız.

Çözüm: Öncelikle kullanıcıdan renk matrisini tanımlayabilmek için matrisi boyutlandırmayı sağlayan N değeri alınmıştır. N değerinin 3 ile 8 aralığında olması gerekmektedir. Eğer sınırlar dışında bir değer girilirse N değeri tekrar tekrar istenmektedir. Sonra NxN boyutunda bir matris tutabilmek için 2. Derecen pointer saklayabilen **matris değişkeni hazırlanmış ve bellekte yer açılması işlemleri yapılmıştır.

Matris hazırlandıktan sonra kullanıcıdan renk matrisi istenmiştir. Burada renkler kullanıcıdan belli rakam karşılıkları ile istenmiştir. Renklerin karşılıkları şu şekildedir;

1. Sarı
2. Lacivert
3. Kırmızı
4. Mor
5. Yeşil
6. Pembe
7. Gri
8. Mavi

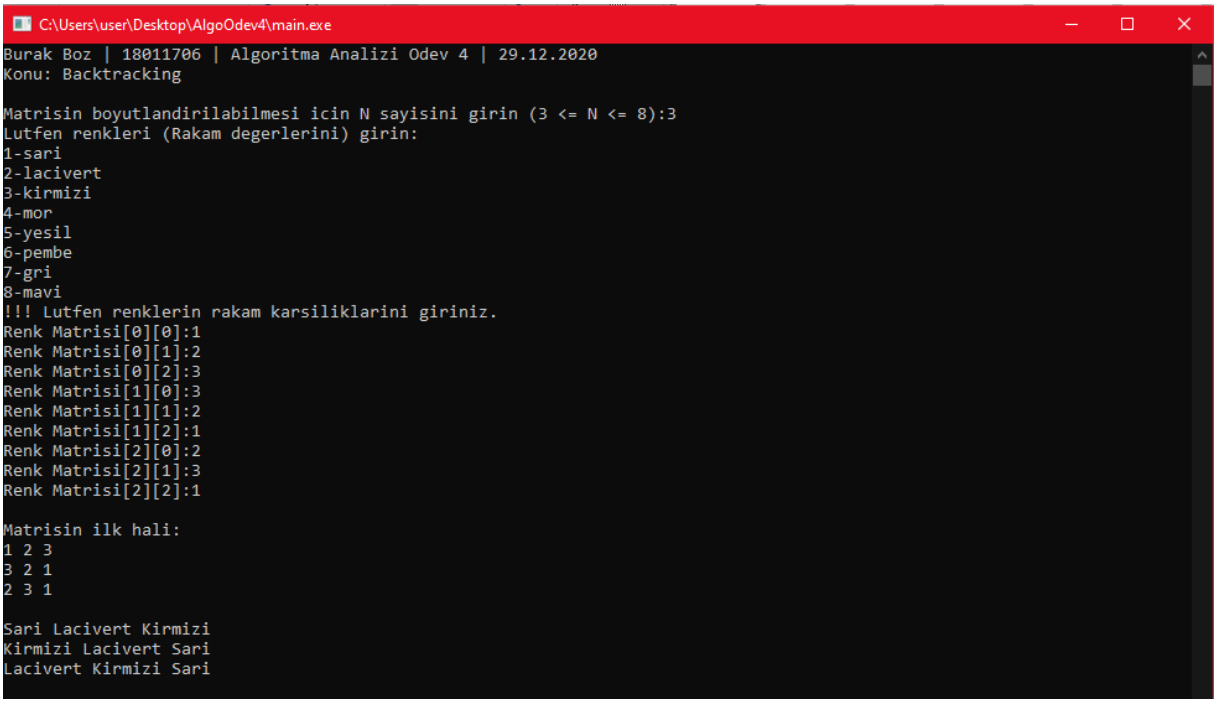
Örneğin

Sarı	Lacivert	Kırmızı
Kırmızı	Lacivert	Sarı
Lacivert	Kırmızı	Sarı

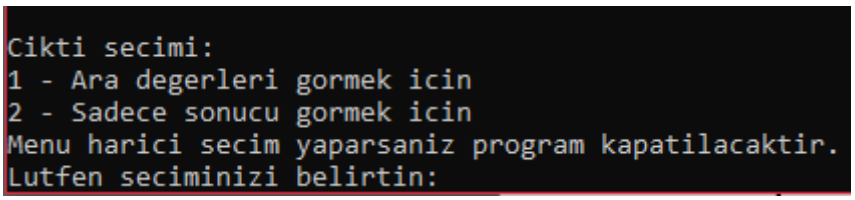
Şeklindeki bir matris kullanıcıdan

1	2	3
3	2	1
2	3	1

Şeklinde alınmıştır.

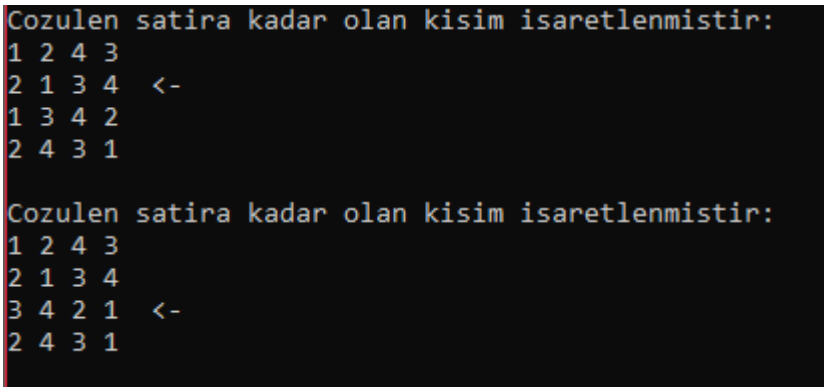


Çözüm aşamasından önce kullanıcıya çıktıları nasıl görmek istediği sorulmaktadır. Kullanıcı problemde belirtildiği üzere 2 seçim hakkına sahiptir. Eğer 1. Seçimi yaparsa matris satırları uygun hale geldikçe ekrana yazdırılacak, eğer 2. Seçimi yaparsa sadece sonuç ekrana yazdırılacaktır.

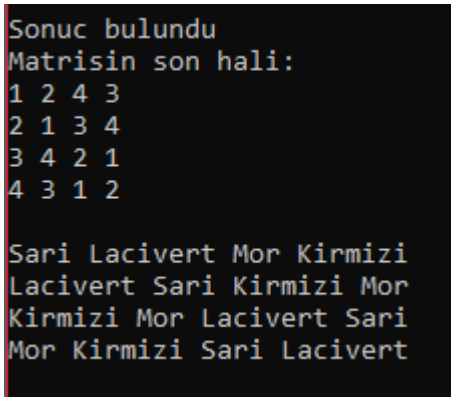


Eğer kullanıcı 1 veya 2 harici seçim yaparsa bellek temizlenerek program kapatılacaktır.

Çözüm aşamasına geçildiğinde kullanıcıdan alınan matris, N değeri ile birlikte recursive fonksiyonlara gönderilir. Seçilen çıktı seçimine göre 2 farklı recursive fonksiyonu bulunmaktadır. Eğer 1 seçimi yapıldıysa recursiveA fonksiyonu çalışmakta ve ara adımlar bir işaretçi ile birlikte ekrana yazdırılmaktadır.



Eğer çözüm bulunabilirse sonuç ekrana gösterilir. Eğer 2. Seçim yapılırsa recursiveB fonksiyonu çalışır ve ara değerler gösterilmeksizin sadece sonuç ekrana yazdırılır.



Eğer matrisin sonucu varsa sonuç önce rakamsal olarak, ardından matrisSonucYazdir() isimli fonksiyonda sayıları renge dönüştürerek tekrar yazdırılır. Sonuç bulunamadıysa “Sonuç bulunamadı” şeklinde belirtilir.

Kullanılan Fonksiyonlar:

- **void matrisYazdir(int **,int);**
  İnputlar: 2 boyutlu int matris pointeri , N değeri
  Açıklama: Bu fonksiyon matrisi sayısal olarak yazdırır.

- **void matrisAraDegerYazdir(int **,int,int);**
  İnputlar: 2 boyutlu int matris pointeri, N değeri, Satır indeksi
  Açıklama: Çıktı için 1. Seçim yapılırsa ara değerleri ekrana yazdırmak için kullanılır. Yukarıdan aşağıya olacak şekilde en son düzenlenen satırı işaretleyerek yazdırır.

- **void shiftRight(int **,int, int);**
  İnputlar: 2 boyutlu int matris pointeri, N değeri, Kaydırılacak indeks
  Açıklama: Bu fonksiyon gelen indeks değerine göre matrisin satırını 1 adım sağa shift etmek için kullanılır.

- **int recursiveA(int **,int,int);**
  İnputlar: 2 boyutlu int matris pointeri, N değeri, Satır indeksi
  Output: Çözüm bulunduysa 1, bulunamadıysa 0 döndürür.
  Açıklama: Eğer çıktı için 1. Seçim seçildiyse bu fonksiyon matrisi çözmeye çalışacak ve ara değerlerin ekrana yazdırılmasını sağlayacak olan recursive fonksiyondur. Yukarıdan aşağıya olacak şekilde ilk satırı temel satır olarak alır ve 2. Satırdan itibaren işlemleri yapmaya başlar. Eğer işlediği satırdaki sayılar üst satırla uyum içerisindeyse bir alt satıra geçer. Alt satırda çözüm bulamaz ise bir üst satıra geri döner ve satırı sağa kaydırarak tekrar çözüm dener. Uyumlu satır buldukça ekrana yazdırılır. İçerisindeki step değişkeni N değeriyle eşit ve result değişkeni 0 a eşitlenmişse sonuç bulunamamış demektir ve geriye 0 döndürür. Diğer durumlarda ise sonuç bulunmuştur ve 1 döndürür.

- **int recursiveB(int **,int,int)**
  İnputlar: 2 boyutlu int matris pointeri, N değeri, Satır indeksi
  Output: Çözüm bulunduysa 1, bulunamadıysa 0 döndürür. Çalışma mantığı bir önceki fonksiyon olan recursiveA fonksiyonuyla aynıdır. Tek farkı bu fonksiyon çıktı seçimi olarak 2. Seçim seçildiyse çalışmaktadır. Sadece sonucu ekrana yazdırmaktadır. Ara değerler ekrana yazdırılmaz.

- **void matrisSonucYazdir(int **,int);**
  İnputlar: 2 boyutlu int matris pointeri, N değeri
  Açıklama: Sonuç olarak bulunan ve rakamsal değerler ile dolu olan matrisin, rakam karşılıklarının tekrar renge çevirilmesi ve ekrana yazdırılması için kullanılan fonksiyondur.

Kullanılan Değişkenler:

int n;//Matrisin boyutu için kullanılacaktır.

int i,j;//Döngüler için kullanılacaklardır.

int sonuc;//Sonucun bulunup bulunamadığı bu değişkende tutulacaktır.

int secim;//Çıktının nasıl istendiği bu değişkende saklanacaktır

int **matris;//Renkler matrisi için kullanılacaktır. 2 boyutlu integer tutacak şekilde tasarlanmıştır.

Ekran Çıktıları:

Örnek 1:

4x4 çözümü olmayan bir renk matrisi girilmiş ve çıktı seçimi olarak 1 seçilmiş ve ara değerlerin ekrana yazdırılması istenmiştir.

İnput:

```
Sarı     Lacivert Mor      Kırmızı       1 2 4 3
Mor      Kırmızı  Sarı     Lacivert      4 3 1 2
Sarı     Kırmızı  Mor      Lacivert      4 3 4 2
Kırmızı  Sarı     Lacivert Mor           3 1 2 4
```

Ara Değerler:

```
Cozulen satira kadar olan kisim isaretlenmistir:
1 2 4 3
4 3 1 2  <-
2 1 3 4
3 1 2 4

Cozulen satira kadar olan kisim isaretlenmistir:
1 2 4 3
4 3 1 2
2 1 3 4  <-
3 1 2 4

Cozulen satira kadar olan kisim isaretlenmistir:
1 2 4 3
4 3 1 2
2 1 3 4  <-
3 1 2 4

Cozulen satira kadar olan kisim isaretlenmistir:
1 2 4 3
4 3 1 2
2 1 3 4  <-
3 1 2 4

Cozulen satira kadar olan kisim isaretlenmistir:
1 2 4 3
4 3 1 2
2 1 3 4  <-
3 1 2 4

Cozulen satira kadar olan kisim isaretlenmistir:
1 2 4 3
4 3 1 2  <-
2 1 3 4
3 1 2 4
```

Bu görselde, matris çözülmeye çalışılırken satırlar çözülemedikçe Backtracking algoritmasına uygun olacak şekilde üst satırlara geri dönüldüğü gösterilmiştir.

Sonuç:

```
Cozulen satira kadar olan kisim isaretlenmistir:
1 2 4 3
4 3 1 2
2 1 3 4  <-
3 1 2 4

Sonuc bulunamadi...
--------------------------------
Process exited after 26.62 seconds with return value 0
Press any key to continue . . .
```

Örnek 2:

4x4 boyutunda ve sonucu bulunan bir matris girilmiş, çıktı seçimi olarak 1 seçilmiş ve ara değerlerinde ekrana yazdırılması sağlanmıştır.

İnput:

Sarı	Lacivert	Mor	Kırmızı	1 2 4 3
Lacivert	Sarı	Kırmızı	Mor	2 1 3 4
Sarı	Kırmızı	Mor	Lacivert	1 3 4 2
Lacivert	Mor	Kırmızı	Sarı	2 4 3 1

Ara Değerler:

```
Cozulen satira kadar olan kisim isaretlenmistir:
1 2 4 3
2 1 3 4  <-
1 3 4 2
2 4 3 1

Cozulen satira kadar olan kisim isaretlenmistir:
1 2 4 3
2 1 3 4
3 4 2 1  <-
2 4 3 1

Cozulen satira kadar olan kisim isaretlenmistir:
1 2 4 3
2 1 3 4
3 4 2 1
4 3 1 2  <-
```

Sonuç:

```
Sonuc bulundu
Matrisin son hali:
1 2 4 3
2 1 3 4
3 4 2 1
4 3 1 2

Sari Lacivert Mor Kirmizi
Lacivert Sari Kirmizi Mor
Kirmizi Mor Lacivert Sari
Mor Kirmizi Sari Lacivert
```

Örnek 3:

5x5 boyutunda, çözümü bulunan bir matris çıktı seçimi olarak 2. Seçimi ile çözülmüştür.

İnput:

```
Sarı     Lacivert Kırmızı  Mor      Yeşil        1 2 3 4 5
Yeşil    Sarı     Lacivert Kırmızı  Mor          5 1 2 3 4
Sarı     Lacivert Kırmızı  Mor      Yeşil        1 2 3 4 5
Mor      Yeşil    Sarı     Lacivert Kırmızı      4 5 1 2 3
Kırmızı  Mor      Yeşil    Sarı     Lacivert     3 4 5 1 2
```

Sonuç:

```
Cikti secimi:
1 - Ara degerleri gormek icin
2 - Sadece sonucu gormek icin
Menu harici secim yaparsaniz program kapatilacaktir.
Lutfen seciminizi belirtin:2
Sonuc bulundu
Matrisin son hali:
1 2 3 4 5
5 1 2 3 4
4 5 1 2 3
3 4 5 1 2
2 3 4 5 1

Sari Lacivert Kirmizi Mor Yesil
Yesil Sari Lacivert Kirmizi Mor
Mor Yesil Sari Lacivert Kirmizi
Kirmizi Mor Yesil Sari Lacivert
Lacivert Kirmizi Mor Yesil Sari

--------------------------------
Process exited after 29.15 seconds with return value 0
Press any key to continue . . .
```

Örnek 4:

6x6 boyutunda sonucu olmayan bir matrisi ara değerleri gösterilecek şekilde çözülmeye çalışılması:

İnputlar:

```
Sarı     Kırmızı  Yeşil    Lacivert Pembe    Mor         1 3 5 2 6 4
Lacivert Yeşil    Pembe    Kırmızı  Mor      Sarı        2 5 6 3 4 1
Kırmızı  Pembe    Mor      Sarı     Yeşil    Lacivert    3 6 4 1 5 2
Mor      Sarı     Kırmızı  Yeşil    Lacivert Pembe       4 1 3 5 2 6
Yeşil    Lacivert Sarı     Pembe    Kırmızı  Mor         5 2 1 6 3 4
Pembe    Mor      Lacivert Kırmızı  Sarı     Yeşil       6 4 2 3 1 5
```

Ara Değerler:

```
Cozulen satira kadar olan kisim isaretlenmistir:
1 3 5 2 6 4
2 5 6 3 4 1
3 6 4 1 5 2
4 1 3 5 2 6  <-
5 2 1 6 3 4
6 4 2 3 1 5

Cozulen satira kadar olan kisim isaretlenmistir:
1 3 5 2 6 4
2 5 6 3 4 1
3 6 4 1 5 2
4 1 3 5 2 6  <-
5 2 1 6 3 4
6 4 2 3 1 5

Cozulen satira kadar olan kisim isaretlenmistir:
1 3 5 2 6 4
2 5 6 3 4 1
3 6 4 1 5 2
4 1 3 5 2 6  <-
5 2 1 6 3 4
6 4 2 3 1 5

Cozulen satira kadar olan kisim isaretlenmistir:
1 3 5 2 6 4
2 5 6 3 4 1
3 6 4 1 5 2  <-
4 1 3 5 2 6
5 2 1 6 3 4
6 4 2 3 1 5

Cozulen satira kadar olan kisim isaretlenmistir:
1 3 5 2 6 4
2 5 6 3 4 1
3 6 4 1 5 2
4 1 3 5 2 6  <-
5 2 1 6 3 4
6 4 2 3 1 5
```

Sonuç:

```
Cozulen satira kadar olan kisim isaretlenmistir:
1 3 5 2 6 4
2 5 6 3 4 1
3 6 4 1 5 2
4 1 3 5 2 6  <-
5 2 1 6 3 4
6 4 2 3 1 5

Cozulen satira kadar olan kisim isaretlenmistir:
1 3 5 2 6 4
2 5 6 3 4 1
3 6 4 1 5 2
4 1 3 5 2 6  <-
5 2 1 6 3 4
6 4 2 3 1 5

Sonuc bulunamadi...
--------------------------------
Process exited after 100.6 seconds with return value 0
Press any key to continue . . .
```

Kodlar:

```
/*
Burak Boz | 18011706
Algoritma Analizi Ödev 4
29.12.2020
*/

#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>


void matrisYazdir(int **,int); //inputlar : (Matris başlangıç pointeri , N değeri) -> Matrisin mevcut halini ekrana yazdırır.
void matrisAraDegerYazdir(int **,int,int); //inputlar: (Matris başlangıç pointeri , N değeri , işlem yapılan satır) -> Ara değerleri yazdırmak için kullanılır.
void shiftRight(int **,int, int);		//inputlar (Matris başlangıç değeri, N değeri , Sağa Kaydırılacak satırın indexi) -> İndeks değeri gönderilen satırı sağa 1 satır shift eder.
int recursiveA(int **,int,int); //(Matris başlangıç pointeri, N değeri, İşlem yapılan satır indeksi) -> Çıktı tercihlerinden 1. seçenek seçilirse bu fonksiyon çalışır.
int recursiveB(int **,int,int); //(Matris başlangıç pointeri, N değeri, İşlem yapılan satır indeksi) -> Çıktı tercihlerinden 2. seçenek seçilirse bu fonksiyon çalışır.
void matrisSonucYazdir(int **,int); //(Matris başlangıç pointeri, N değeri) -> Matrisi sayı olarak değilde renk olarak yazdırmak için kullanılır.

int main()
{
	printf("Burak Boz | 18011706 | Algoritma Analizi Odev 4 | 29.12.2020\n");
	printf("Konu: Backtracking\n\n");

	int n=0;   //Matrisin boyutu için kullanılacaktır.
	int i=0,j=0;		//Döngüler için kullanılacaklardır.
	int sonuc=0;		//Sonucun bulunup bulunamadığı bu değişkende tutulacaktır.
	int secim=0;		//Çıktının nasıl istendiği bu değişkende saklanacaktır
	do{
		printf("Matrisin boyutlandirilabilmesi icin N sayisini girin (3 <= N <= 8):");
		scanf("%d",&n);
	}while(!(n>=3 && n<=8));

	int **matris;		//Renkler matrisi için kullanılacaktır. 2 boyutlu integer tutacak şekilde tasarlanmıştır.
	matris=(int **)calloc(n,sizeof(int*));
	if(matris!=NULL)
	{
		for(i=0;i<n;i++)
		{
			matris[i]=(int *)calloc(n,sizeof(int));
			if(matris[i]==NULL){
				printf("Bellekte yeterince yer bulunmuyor.");//Bellekte yer ayrılamadıysa bu hata ekrana yazdırılır ve program kapatılır.
				return -1;
			}
		}
	}


	printf("Lutfen renkleri (Rakam degerlerini) girin:\n");
	printf("1-sari\n2-lacivert\n3-kirmizi\n4-mor\n5-yesil\n6-pembe\n7-gri\n8-mavi\n");
	printf("!!! Lutfen renklerin rakam karsiliklarini giriniz.\n");
```

```
	for(i=0;i<n;i++)//Kullanıcıdan renkler matrisi alınıyor. Renkler yerine temsil edilen sayılar kullanıcıdan isteniyor.
	{
		for(j=0;j<n;j++)
		{
			printf("Renk Matrisi[%d][%d]:",i,j);
			scanf("%d",&matris[i][j]);
		}
	}

	printf("\nMatrisin ilk hali:\n");
	matrisYazdir(matris,n);
	printf("\n");
	matrisSonucYazdir(matris,n);

	printf("\n\nCikti secimi:\n");
	printf("1 - Ara degerleri gormek icin\n");
	printf("2 - Sadece sonucu gormek icin\n");
	printf("Menu harici secim yaparsaniz program kapatilacaktir.\nLutfen seciminizi belirtin:");
	scanf("%d",&secim);
	/*
		Raporda belirtilen çıktılardan A maddesini görmek için 1 seçimini yapınız.
		1 seçimini yaptığınızda matrisin üst satırlara göre düzgün sıra ile yerleştiği yere kadar bir işaretçi ile
ekrana yazdırılacaktır.
		2 seçimini yaparsanız sadece sonucu yazdıracaktır.
	*/
	if(secim==1)
	{
		sonuc = recursiveA(matris,n,1);
		if(sonuc==1)
		{
			printf("Sonuc bulundu\n");
			printf("Matrisin son hali:\n");
			matrisYazdir(matris,n);
			printf("\n");
			matrisSonucYazdir(matris,n);
		}
		else
		{
			printf("Sonuc bulunamadi...");
		}
	}
	else if(secim==2)
	{
		sonuc = recursiveB(matris,n,1);
		if(sonuc==1)
		{
			printf("Sonuc bulundu\n");
			printf("Matrisin son hali:\n");
			matrisYazdir(matris,n);
			printf("\n");
			matrisSonucYazdir(matris,n);
		}
		else
		{
			printf("Sonuc bulunamadi...");
		}
	}
	else
	{
```

```
                printf("Yanlis secim. Program kapatiliyor.");
                for(i=0;i<n;i++)
                {
                        free(matris[i]);
                }
                free(matris);
                return -2;
        }



        //Bellekte ayrılan alanların işletim sistemine bırakılması
        for(i=0;i<n;i++)
        {
                free(matris[i]);
        }
        free(matris);
        return 0;
}

//Matrisin o anki durumunu sayı olarak ekrana yazdırmak için kullanılır.
void matrisYazdir(int **matris,int n)
{
        int i=0,j=0;
        for(i=0;i<n;i++)
        {
                for(j=0;j<n;j++)
                {
                        printf("%d ",matris[i][j]);
                }
                printf("\n");
        }
}

//Matrisin o anki durumundaki sayıları renklere çevirerek yazdırır.
void matrisSonucYazdir(int **matris,int n)
{
        int i=0,j=0;
        for(i=0;i<n;i++)
        {
                for(j=0;j<n;j++)
                {
                        switch(matris[i][j])
                        {
                                case 1:
                                        printf("Sari ");
                                        break;
                                case 2:
                                        printf("Lacivert ");
                                        break;
                                case 3:
                                        printf("Kirmizi ");
                                        break;
                                case 4:
                                        printf("Mor ");
                                        break;
                                case 5:
                                        printf("Yesil ");
                                        break;
```

```
                                        case 6:
                                                printf("Pembe ");
                                                break;
                                        case 7:
                                                printf("Gri ");
                                                break;
                                        case 8:
                                                printf("Mavi ");
                                                break;
                                        default:

                                                printf("-1 ");

                                }
                        }
                        printf("\n");
                }
}

/*Matris satırları sağa shift edilir ve eğer üst kısımla uyumlu hale gelirse kullanıcıdan alınan seçime göre
ekrana yazdırılabilir. Bu fonksiyon düzenlenmiş halin hangi satıra kadar uyum içerisinde bulunduğunu, reqursiveA
fonksiyonundan aldığı satırIndex değerine göre bir işaretçiyle işaretler ve ekrana yazdırır.*/
void matrisAraDegerYazdir(int **matris,int n,int satirIndex)
{
        int i=0,j=0;
        for(i=0;i<n;i++)
        {
                for(j=0;j<n;j++)
                {
                        printf("%d ",matris[i][j]);
                }
                if(i==satirIndex)
                {
                        printf(" <- ");
                }
                printf("\n");
        }
}

//index değişkenine gelen sayıya göre matrisin ilgili satırını bir sağa shift eder. En sağdaki değer satırın başına alınır.
void shiftRight(int **matris,int n, int index)
{
        int dizi[n];
        int i=0;
        for(i=1;i<n;i++)
        {
                dizi[i]=matris[index][i-1];
        }
        dizi[0]=matris[index][n-1];

        for(i=0;i<n;i++)
        {
                matris[index][i]=dizi[i];
        }

}

//Matrisi çözmeye çalışan reqursive fonksiyondur. Eğer ara adımlar kullanıcı tarafından görüntülenmek istenirse bu
fonksiyon çalışır.
```

```
//Sonuç bulunduysa 1, bulunamadıysa 0 döndürür.
int recursiveA(int **matris,int n,int satir)
{
	int step=0;
	int result=0;
	while(step<n && result ==0)
	{
		bool cik=false;
		int j=0;
		while(j<n && cik==false)
		{
			int k=satir-1;
			while(k>0 && matris[k][j] != matris[satir][j])
			{
				k--;
			}
			if(matris[k][j] == matris[satir][j])
			{
				cik=true;
			}
			j++;
		}
		if(cik==true)
		{
			shiftRight(matris,n,satir);
		}
		else
		{
			printf("Cozulen satira kadar olan kisim isaretlenmistir:\n");
			matrisAraDegerYazdir(matris,n,satir);//Ara adımın ekrana yazdırılmasını sağlayan fonksiyon
			printf("\n");
			if(satir==n-1)
			{
				result = 1;
			}
			else{
				result = recursiveA(matris,n,satir+1);
			}
		}
		step++;
	}
	if(step == n && result == 0)
	{
		return 0;
	}
	else
	{
		return 1;
	}
}

//Matrisi çözmeye çalışan recursive fonksiyondur. Kullanıcı ara adımları görmeden sadece sonucu görmek isterse bu
fonksiyon çalıştırılır.
//Sonuç bulunduysa 1, bulunamadıysa 0 döndürür.
int recursiveB(int **matris,int n,int satir)
{
	int step=0;
	int result=0;
	while(step<n && result ==0)
```

```
	{
		bool cik=false;
		int j=0;
		while(j<n && cik==false)
		{
			int k=satir-1;
			while(k>0 && matris[k][j] != matris[satir][j])
			{
				k--;
			}
			if(matris[k][j] == matris[satir][j])
			{
				cik=true;
			}
			j++;
		}
		if(cik==true)
		{
			shiftRight(matris,n,satir);
		}
		else
		{
			if(satir==n-1)
			{
				result = 1;
			}
			else{
				result = recursiveB(matris,n,satir+1);
			}
		}
		step++;
	}
	if(step == n && result == 0)
	{
		return 0;
	}
	else
	{
		return 1;
	}
}
```