Yıldız Teknik Üniversitesi

Elektrik-Elektronik Fakültesi

Bilgisayar Mühendisliği Bölümü

BLM1012

YAPISAL PROGRAMLAMAYA GİRİŞ

GR: 2

Öğr. Gör. Dr. Ahmet ELBİR

DÖNEM PROJESİ

İsim: Sude

No: ELİTOK

E-posta: sude.elitok@std.yildiz.edu.tr

```c
#include<stdio.h>
#include <time.h>
#include <stdlib.h>
#define MAX 15   // kullanılacak max sayi

void readFromFile(int matris[][MAX], char *fileName);
void reset(int matris[][MAX], int N);
void fillMtr(int matris[][MAX], int *N);
void drawBoard(int matris[][MAX], int N);
int name(char user[][20], int *k);
int mainMenu();
int gameMenu();
int option();
int getPoint(int matris[][MAX], int N, int *k, int *x1, int *y1, int *x2, int *y2, int top, int move[][7]);
void push(int x1, int y1, int x2, int y2, int *top, int move[][7], int dogrultu, int k);
void undo(int *top, int matris[][MAX], int move[][7]);
int control(int matris[][MAX], int N);
void manuel(int matris[][MAX], int move[][7], int N, int *numUndo, int label[][MAX], int c[][MAX], int index[][4]);
void findIndex(int matris[][MAX], int index[][4], int N);
void copyForLabeling(int matris[][MAX], int label[][MAX], int N, int a);
void dfs(int a, int x, int y, int current_label,int c[][MAX], int N,int label[][MAX]);
int chokepoint(int N, int label[][MAX], int matris[][MAX], int c[][MAX], int index[][4]);
void automatic(int a, int i, int j, int N, int matris[][MAX], int index[][4], int move[][7], int *top, int label[][MAX], int c[][MAX]);
void scoreFunc(int choice, int choice2, int N, int k, int score[], int numUndo);
void printScore(int score[], char user[][20], int k);
```

İlk kısımda kullandığım kütüphaneler ve fonksiyonlar bulunuyor.

1. MAIN FONKSİYONU

```c
int main(){
    srand(time(NULL));
    int i,j,matris[MAX][MAX]={{0}}, label[MAX][MAX], c[MAX][MAX], m, choice, choice2, k=-1, move[300][7], top=-1, dogrultu, index[MAX][4], a=1, man, oto, score[100]={0}, numUndo;
    int N;
    char fileName[20], user[100][20];

    printf("\nSAYI ESLESTIRME\n");
    printf("\nManuel Oyun Kurallari:");
    printf("\n1)Baslangic noktanızda ilerletmek istediginiz sayı bulunmalidir!!");
    printf("\n2)Noktayi her adimda saga, sola, asagiya ya da yukariya bir dogru boyunca ilerletmelisiniz!!");
    printf("\n3)Ilerleme boyunca baska bir sayi bulunmamali!!");
```

Kullandığım değişkenler ve oyunun manuel kısmıyla ilgili bilgilendirme bulunuyor. Değişkenlerin nerelerde kullandığımı ilerleyen kısımlarda açıklayacağım.

```c
do{
    choice = mainMenu();

    if(choice != 4){
        k = name(user, &k); //kullanıcı adı
    }
```

Daha sonrasında do while ile kullanıcıya yapılmak istenen işlemi soruyorum(ana menü):

```c
int mainMenu(){
    int choice;
    do{
        printf("\n\nYapmak istediginiz islemi seciniz:");
        printf("\n1)Rastgele Matris Olustur");
        printf("\n2)Dosyadan Matris Olustur");
        printf("\n3)Kullanicilarin Skorlarini Goster");
        printf("\n4)Cikis");
        printf("\nSecim: ");
        scanf("%d", &choice);
    }while(choice != 1 && choice != 2 && choice != 3 && choice != 4);

    return choice;
}
```

Kullanıcı 1 veya 2'yi tercih ederse ismini alıyorum.

```
int name(char user[][20], int *k){
    int i, c;
    if(*k == -1){
        printf("\n1. oyuncu");
        printf("\nAdinizi giriniz: ");
        scanf("%s", user[0]);
        *k += 1;
    }
    else{
        for(i=0;i<*k;i++){
            printf("%d) %s", i+1, user[i]);
        }
        printf("\nEger daha once oynadiysaniz numaranizi girin. Oynamadıysanız 0'a basin: ");
        scanf("%d", &c);
        if(c == 0){
            *k += 1;
            printf("\n%d. oyuncu", *k+1);
            printf("\nAdinizi giriniz: ");
            scanf("%s", user[*k]);
        }
        else{
            return c; //skor için oyuncunun numarası
        }
    }
    return *k;
}
```

Eğer oyuncu en son oynayan kişinin öncesinde oynadıysa ismini girmiştir, skorunu daha iyi hesaplayabilmek adına hangi oyuncu olduğunu soruyorum. Eğer yeni bir oyuncuysa ismini alıyorum. Return c ve k oyuncunun sırasını belirtiyor, skor hesaplaması için tuttum.

Sonrasında oyuncunun ana menüdeki seçimine göre oyun ilerliyor.

1.1 RASTGELE MATRİS OLUŞTUR

```
if(choice == 1){

    do{
        choice2 = gameMenu();

        if(choice2 != 3){
            fillMtr(matris, &N);
            drawBoard(matris, N);
        }

        if(choice2 == 1){ //manuel mod
            manuel(matris, move, N, &numUndo, label, c, index);
        }

        if(choice2 == 2){//otomatik mod
            findIndex(matris, index, N);
            i = index[0][0];
            j = index[0][1];
            top = -1;
            a = 1;
            if(control(matris, N) && !chokepoint(N, label, matris, c, index)){
                drawBoard(matris, N);
            }
            else{
                printf("\n!!Bu matrisin cozumu yok!!\n");
            }
          }
        scoreFunc(choice, choice2, N, k, score, numUndo);
    }while(choice2 != 3);
}
```

Oyuncu ana menüde 1 numarayı seçtiğinde karşısına ikinci bir seçenek sunuluyor:

```
int gameMenu(){
    int choice;

    do{
        printf("\n\nYapilacak islemi seciniz:");
        printf("\n1)Manuel Modda Oyna");
        printf("\n2)Otomatik Modda Oyna");
        printf("\n3)Ana Menuye Don");
        printf("\nSecim: ");
        scanf("%d", &choice);
    }while(choice != 1 && choice != 2 && choice != 3);

    return choice;
}
```

Bu sorunun ardından seçimine göre 1 ya da 2'yi tercih ettiyse istedikleri boyuttaki matris rastgele doldurulup ekrana yazılıyor.

```c
void reset(int matris[][MAX], int N){
    int i, j;

    for(i=0 ; i<N ; i++){
        for(j=0 ; j<N ; j++){
            matris[i][j] = 0;
        }
    }
}

void fillMtr(int matris[][MAX], int *N){
    int i, j, k, m, tempx, tempy, flag=0;

    printf("\nMatris boyutu: ");
    scanf(" %d", N);

    k = *N;
    reset(matris, k); //matrisi sıfırlama

    for(k=1 ; k<=*N ; k++){//matrisi rastgele sayilarla doldurma
        tempx = -2; //-2 vermemin nedeni randomda 0 çıkınca -1 ile karşılaştıracak olması
        tempy = -2;
        for(m=0 ; m<2 ; m++){ //aynı sayıdan 2 kere eklmememiz lazım
            do{
                i = rand()% *N;
                j = rand()% *N;
                if(i-1 == tempx || i+1 == tempx || j-1 == tempy || j+1 == tempy){ //diğer sayının çevresinde olup olmadığının kontrolü
                    flag = 1;
                }
                else{
                    flag = 0;
                }
            }while(matris[i][j] != 0 || flag == 1);
            matris[i][j] = k;
            tempx = i;
            tempy = j;
        }
    }
}
```

Önce istenilen boyut alınıyor daha sonrasında o boyut kadarki matris sıfırlanıp sayılar rastgele yerleştiriliyor. Aynı sayıların yan yana ya da çapraz gelmemesi adına kontrol yaptım.

```c
void drawBoard(int matris[][MAX], int N){
    int i,j,k;
    for(i=0;i<N;i++){
        printf("\n");
        for (k=0;k<N;k++)
            printf("-------");
        printf("\n");
        for(j=0;j<N;j++)
            if (matris[i][j]!=0)
                printf("  %d   |",matris[i][j]);
            else
                printf("       |",matris[i][j]);
    }
}
```

Matrisi ekrana yazdırma.

1.1.1 MANUEL MODDA OYNA

```
void manuel(int matris[][MAX], int move[][7], int n, int *numUndo, int label[][MAX], int c[][MAX], int index[][4]){
    int top=-1, i, j, x1, y1, x2, y2, k, dogrultu, selection;

    *numUndo = 0; //skor için gerekli
    do{
        selection = option(); //hamle yap, geri al, pes et
        if(selection == 1){ //hamle yap
            dogrultu = getPoint(matris, n, &k, &x1, &y1, &x2, &y2, top, move);
            push(x1, y1, x2, y2, &top, move, dogrultu, k);

            i = x1;
            j = y1;

            if(matris[i][j] == matris[x2][y2]){
                move[top][6] = 1;
                printf("\nSayilar eslestirildi!");
            }

            if(dogrultu){ //y üzerinde hareket
                do{
                    matris[x1][j] = matris[x1][y1];
                    j += move[top][5];
                }while(j != y2);
                matris[x1][j] = matris[x1][y1];
            }

            else{
                do{
                    matris[i][y1] = matris[x1][y1];
                    i += move[top][5];
                }while(i != x2);
                matris[i][y1] = matris[x1][y1];
            }
            drawBoard(matris, n);

        }

        else if(selection == 2){ //geri al
            *numUndo += 1;
            undo(&top, matris, move);
            drawBoard(matris, n);
        }

        if(control(matris, n) && chokepoint(n, label, matris, c, index)){
            printf("\n\nTEBRIKLER KAZANDINIZ.");
            selection = 3;
        }

    }while(selection != 3);
}
```

Selection ile pes edildiğinde(çözüm olmama ihtimaline karşı) ya da matris tamalandığında oyun tamamlanıyor.

```
int option(){ //oynarken yapacağı işlem
    int choice;
    do{
        printf("\n\nYapilacak islemi seciniz:");
        printf("\n1)Hamle yap");
        printf("\n2)Hamleyi geri al");
        printf("\n3)Pes et ve oyun menusune don"); //oyun çözümsüz çıkarsa ya da devam etmek istemezse
        printf("\nSecim: ");
        scanf("%d", &choice);
    }while(choice != 1 && choice != 2 && choice != 3);
    return choice;
}
```

Oyuncuya hamle, geri alma pes etme seçenekleri sunuluyor. Oyuncunun seçimine göre oyun ilerliyor.

1.1.1.1 HAMLE YAP

```
int getPoint(int matris[][MAX], int N, int *k, int *x1, int *y1, int *x2, int *y2, int top, int move[][7]){
    int i, j, flag=0, flag2=0, dogrultu;

    do{

        printf("\n\nBaslangic noktasi (x,y): ");
        scanf("%d %d", x1, y1);
        printf("\nBitis noktasi (x,y): ");
        scanf("%d %d", x2, y2);

        if(*x1 != *x2 && *y1 != *y2){ // doğru boyunca hareket ettirebilmek adına x veya y'lerden biri aynı olmalı
            printf("\n!!Noktayi sadece bir dogru boyunca hareket ettirebilirsiniz!!\n");
            flag = 1;
        }
        else if(*x1 == *x2 && *y1 == *y2){
            printf("\nNoktalar birbirinden farkli olmali");
            flag = 1;
        }
        else if(*x1>=N || *x2>=N || *y1>=N || *y2>=N || *x1<0 || *x2<0 || *y1<0 || *y2<0){ //matrisin dışına çıkma
            printf("\n!!Matrisin sinirlari disina cikmamalisiniz!!\n");
            flag = 1;
        }

        else{ // ilerleme yolunda başka bir sayının olup olmadığının kontrolü
            flag = 0;
            if(*x1 == *x2){ //x'in mi yoksa y'nin mi doğrultusunda hareket var? i'ler aynıysa x ekseninde hareket
                dogrultu = 1;
                if(*y1 < *y2){
                    *k = 1; //sağa ilerleme
                }

                else{
                    *k = -1; //sola ilerleme
                }

                i = *x1;
                j = *y1 + *k;
            }
```

```
else{ //i'ler aynı değilse y ekseninde
    dogrultu = 0;
    if(*x1 < *x2){
        *k = 1; //aşağı ilerleme
    }

    else{
        *k = -1; //yukarı ilerleme
    }

    i = *x1 + *k;
    j = *y1;
}

while(matris[i][j] == 0 && flag2 == 0){
    if(dogrultu){ //dogrultu=1 ise x üzerinde hareket
        if(j == *y2){
            flag2 = 1; //y2'yi geçmesini engellemek için
        }

        else{
            j += *k;
        }
    }

    else{ // dogrultu=0'dır. x üzerinde hareket
        if(i == *x2){
            flag2 = 1; //x2'yi geçmesini engellemek için
        }

        else{
            i += *k;
        }
    }
}
```

```
            if(flag2 == 1 && (matris[*x2][*y2] == matris[*x1][*y1] || matris[*x2][*y2] == 0)){ //bu adımda sayılar eşleşmiş ya da son adımdaki sayı 0 olmuş olacak ve hamle yapılabilecek
                flag2 = 0;
            }

            if(flag2 == 1){ //doğrultu üzerinde başka bir sayı var
                printf("\n!!Gidecegi yol uzerinde baska bir sayi bulunmamali!!\n");
            }
        }
    }while(flag == 1 || flag2 == 1);

    return dogrultu; //manuel fonksiyonunda kullanabilmek için
}
```

Yapılabilecek hamleyi oyuncudan aldıktan sonra alınan hamleyi başlangıç ve bitiş noktalarını, doğrultusunu, yönünü ve eşleşip eşleşmeme durumunu bir matrise aktarıyorum. Geri alma işlemi için kullanacağım.

```
void push(int x1, int y1, int x2, int y2, int *top, int move[][7], int dogrultu, int k){
    int i;

    *top = *top + 1;
    move[*top][0] = x1;
    move[*top][1] = y1;
    move[*top][2] = x2;
    move[*top][3] = y2;
    move[*top][4] = dogrultu;
    move[*top][5] = k;
    move[*top][6] = 0; //eşleştirilirse sonrasında değiştireceğim

}
```

İlerlediğim yerdeki sayı ve bulunduğum sayı birbirine eşitse sayılar eşleştirilmiş olur.

```
if(matris[i][j] == matris[x2][y2]){
    move[top][6] = 1;
    printf("\nSayilar eslestirildi!");
}
```

Daha sonrasında ilerlemeyi matrise yansıtıyorum.

```
if(dogrultu){ //y üzerinde hareket
    do{
        matris[x1][j] = matris[x1][y1];
        j += move[top][5];
    }while(j != y2);
    matris[x1][j] = matris[x1][y1];
}

else{
    do{
        matris[i][y1] = matris[x1][y1];
        i += move[top][5];
    }while(i != x2);
    matris[i][y1] = matris[x1][y1];
}
drawBoard(matris, N);
```

1.1.1.2 HAMLEYİ GERİ AL

```
else if(selection == 2){ //geri al
    *numUndo += 1;
    undo(&top, matris, move);
    drawBoard(matris, N);
}
```

Yapılan geri alma işlemi sayısını tuttuğum değişkeni 1 arttırıyorum(skor için).

```c
void undo(int *top, int matris[][MAX], int move[][7]){

    if(*top != -1){
        if(move[*top][4]){ //doğrultu x
            if(move[*top][6] == 1){ //o hamlede sayı eşleştirilmiştir yani o yerde bulunan sayı başta rastgele dağıtılan sayıdır. silinemez
                move[*top][3] -= move[*top][5]; //bir önce bulunduğu yer
            }
            while(move[*top][3] != move[*top][1]){
                matris[move[*top][2]][move[*top][3]] = 0;
                move[*top][3] -= move[*top][5];
            }
        }

        else{ //doğrultu y
            if(move[*top][6] == 1){ //o hamlede sayı eşleştirilmiştir yani o yerde bulunan sayı başta rastgele dağıtılan sayıdır. silinemez
                move[*top][2] -= move[*top][5]; //bir önce bulunduğu yer
            }
            while(move[*top][2] != move[*top][0]){
                matris[move[*top][2]][move[*top][3]] = 0;
                move[*top][2] -= move[*top][5];
            }
        }

        *top = *top - 1; //move'u değiştirmeye gerek duymadım zaten bir sonraki hamlede üstüne yazılacak
    }
    else{
        printf("\n!!Geri alinacak hamle yok!!\n");
    }
}
```

Yukarıdaki işlemler gerçekleştiriliyor. *top en sonki hamlenin numarasını tutuyor. Hamle yapıldığında +1, geri alma yapıldığında -1 değiştiriyorum. Geri alma tamamlandıktan sonra matrisin son halini yazdırıyorum.

```c
if(control(matris, N) && chokepoint(N, label, matris, c, index)){
    printf("\n\nTEBRIKLER KAZANDINIZ.");
    selection = 3;
}
```

Matrisin tamamlanıp tamamlanmadığını görmek adına çeşitli kontrolleri yapıyorum. Eğer tamamlandıysa selection'ın değerini 3 yaparak oyundan çıkarıyorum.

```c
int control(int matris[][MAX], int N){ //matrisin dolup dolmadığının kontrolü. 1 ise dolmuştur
    int i=0, j=0;

    while (i < N) {
        j = 0;
        while (j < N) {
            if (matris[i][j] == 0){
                return 0;
            }
            j++;
        }
        i++;
    }
    return 1;
}
```

Boş alan kontrolü.

```
int chokepoint(int N, int label[][MAX], int matris[][MAX], int c[][MAX], int index[][4]){ //o anki matrisin durumunda diğer sayıların birbirine ulaşıp ulaşamayacağını kontrol eder
    int i, j, k=0, flag=0, a=1, component=1;

    while(flag == 0 && a <= N){
        component = 1;
        copyForLabeling(matris, c, N, a);
        reset(label, N);
        for (i = 0; i < N; i++){
            for (j = 0; j < N; j++){
                // if not labelled and c(i,j)=a
                if (!label[i][j] && c[i][j] == a)
                    dfs(a, i, j, component++, c, N, label);
            }
        }

        if(label[index[a-1][0]][index[a-1][1]] != label[index[a-1][2]][index[a-1][3]]){ // a sayıları işaretlenmiştir ve a sayıları matrisin o anki haliyle birleştirilemez
            flag = 1;

        }

        else{
            a++;
        }
    }

    if(flag == 0)
        return 0;
    else
        return 1;
}
```

Burada ise bütün sayıların eşleşip eşleşmediğinin kontrolünü yapıyorum. Connected component labelling yöntemini kullandım.(başta otomatik mod için kodlamıştım sonrasında manuelde de kullanmaya karar verdim, ikisine de uyumlu). Aşağıdaki iki fonksiyondan yararlandım.

```
void copyForLabeling(int matris[][MAX], int c[][MAX], int N, int a){
    int i, j;
    reset(c, N);
    for(i=0 ; i<N ; i++){
        for(j=0 ; j<N ; j++){
            if(matris[i][j] == 0 || matris[i][j] == a){ //0 olan kısımları bağlı olup olmadığına bakmak istediğim sayıya eşitliyorum
                c[i][j] = a;
            }
        }
    }
}


void dfs(int a, int x, int y, int current_label,int c[][MAX], int N,int label[][MAX]){
    int direction;
    // direction vectors
    int dx[] = {+1, 0, -1, 0};
    int dy[] = {0, +1,  0, -1};
    //      down, right, up, left

    if (x < 0 || x == N)
        return; // out of bounds
    if (y < 0 || y == N)
        return; // out of bounds
    if (label[x][y] || c[x][y] != a)
        return; // already labeled or not marked with a in m

    // mark the current cell
    label[x][y] = current_label;
    // recursively mark the neighbors
    dfs(a, x+1, y, current_label,c,N,label);
    dfs(a, x, y+1, current_label,c,N,label);
    dfs(a, x-1, y, current_label,c,N,label);
    dfs(a, x, y-1, current_label,c,N,label);
}
```

Manuel mod oyun bitene ya da kullanıcı pes edene kadar bu şekilde devam ediyor. Sonlandığında oyun menüsüne dönülüyor.

1.1.2 OTOMATİK MODDA OYNA

Otomatik olarak oyunu aynar

```
if(choice2 == 2){//otomatik mod
    findIndex(matris, index, N);
    i = index[0][0];
    j = index[0][1];
    top = -1;
    a = 1;
    automatic(a, i, j, N, matris, index, move, &top, label, c);
    if(control(matris, N) && !chokepoint(N, label, matris, c, index)){
        drawBoard(matris, N);
    }
    else{
        printf("\n!!Bu matrisin cozumu yok!!\n");
    }
  }
```

Öncelikle rastgele atanan matristeki sayıların yerini bulup bir index matrisine kaydediyorum.

```
void findIndex(int matris[][MAX], int index[][4], int N){
    int i=0, j=0, k=0;

    for(i=0 ; i<N ; i++){ //index'in elemanlarını -1' eşitliyorum böylelikle elemanların indislerini kopyalayıp kopyalamadığım belli olacak
        for(j=0 ; j<4 ; j++){
            index[i][j] = -1;
        }
    }

    i = 0;
    while (i < N && k != 2*N) { //k == 2*N olursa bütün sayılar bulunmuş demektir
        j = 0;
        while (j < N && k != 2*N) {
            if (matris[i][j] != 0){
                if(index[matris[i][j]-1][0] == -1){ //o sayının ilki bulundu //-1
                    index[matris[i][j]-1][0] = i; //x değeri
                    index[matris[i][j]-1][1] = j; //y değeri
                }

                else{ //o sayının ikincisi
                    index[matris[i][j]-1][2] = i; //x değeri
                    index[matris[i][j]-1][3] = j; //y değeri
                }
                k++;
            }
            j++;
        }
        i++;
    }

}
```

While döngüsüyle elemanları arıyorum hepsi bulununca döngüden çıkıyor. Sonrasında otomatik moda başlamak için i ve j'ye 1. Sayının ilk indislerinin veriyorum ve diğer 1'e ulaşana kadar hamleler yaptırıyorum.

```
void automatic(int a, int i, int j, int N, int matris[][MAX], int index[][4], int move[][7], int *top, int label[][MAX], int c[][MAX]){

    if(i < 0 || j < 0){
        *top -= 1;
        return;
    }

    if(i >= N || j >= N){
        *top -= 1;
        return;
    }

    if(control(matris, N)){
        return;
    }

    if(matris[i][j] != 0 && !((i == index[a-1][0] && j == index[a-1][1] && (*top == -1 || move[*top-1][6] == 1)) || (i == index[a-1][2] && j == index[a-1][3]))){
        *top -= 1;
        return;
    }
```

i ve j değerleri için gerekli kontrolleri gerçekleştiriyorum. 4. if bir elemanı eşleştirip diğer elemana geçtiğimizde hamla yapmayı sağlamak, sayıları birbiri üstüne yazmamak ve aynı sayıları birbirine eşitlemeyi sağlamak için kontrol sağlıyor.

```
    matris[i][j] = a;

    *top = *top + 1;

    move[*top][6] = 0;

    drawBoard(matris,N);
    printf("\n");

    if(i == index[a-1][2] && j == index[a-1][3] && a <= N){
        move[*top][6] = 1; //eşleştirildi
        a++;
        i = index[a-1][0];
        j = index[a-1][1];
        automatic(a, i, j, N, matris, index, move, top, label, c);
    }

    if(chokepoint(N, label, matris, c, index)){
        *top -= 1;
        undo(top, matris, move);
        return;
    }
```

Sonrasında hamleyi gerçekleştiriyorum ve ekrana yazdırıyorum. Eğer eşleştirilmişse recursive olarak diğer sayıya geçiyorum. Eğer eşleştirilmemişse connected component labelling kullanarak yaptığım hamlenin diğer sayıların birleşmesini engelleyip engellemediğine bakıyorum.

```
int chokepoint(int N, int label[][MAX], int matris[][MAX], int c[][MAX], int index[][4]){ //o anki matrisin durumunda diğer sayıların birbirine ulaşıp ulaşamayacağını kontrol eder
    int i, j, k=0, flag=0, a=1, component=1;

    while(flag == 0 && a <= N){
        component = 1;
        copyForLabeling(matris, c, N, a);
        reset(label, N);
        for (i = 0; i < N; i++){
            for (j = 0; j < N; j++){
                // if not labelled and c(i,j)=a
                if (!label[i][j] && c[i][j] == a)
                    dfs(a, i, j, component++, c, N, label);
            }
        }

        if(label[index[a-1][0]][index[a-1][1]] != label[index[a-1][2]][index[a-1][3]]){ // a sayıları işaretlenmiştir ve a sayıları matrisin o anki haliyle birleştirilemez
            flag = 1;

        }

        else{
            a++;
        }
    }

    if(flag == 0)
        return 0;
    else
        return 1;
}
```

```
void copyForLabeling(int matris[][MAX], int c[][MAX], int N, int a){
    int i, j;
    reset(c, N);
    for(i=0 ; i<N ; i++){
        for(j=0 ; j<N ; j++){
            if(matris[i][j] == 0 || matris[i][j] == a){ //0 olan kısımları bağlı olup olmadığına bakmak istediğim sayıya eşitliyorum
                c[i][j] = a;
            }
        }
    }
}
```

0lar üzerinden sayılar birbirine bağlantılı mı değil mi bakmak için 0'ları sayıya eşitleyerek labelling için bir matrise atıyorum.

```
void dfs(int a, int x, int y, int current_label,int c[][MAX], int N,int label[][MAX]){
    int direction;
    // direction vectors
    int dx[] = {+1, 0, -1, 0};
    int dy[] = {0, +1,  0, -1};
    //      down, right, up, left

    if (x < 0 || x == N)
        return; // out of bounds
    if (y < 0 || y == N)
        return; // out of bounds
    if (label[x][y] || c[x][y] != a)
        return; // already labeled or not marked with a in m

    // mark the current cell
    label[x][y] = current_label;
    // recursively mark the neighbors
    dfs(a, x+1, y, current_label,c,N,label);
    dfs(a, x, y+1, current_label,c,N,label);
    dfs(a, x-1, y, current_label,c,N,label);
    dfs(a, x, y-1, current_label,c,N,label);
}
```

Eğer işaretledikten sonra findIndex ile bulduğum indisleri birbiriyle karşılaştırarak bağlanabilip bağlanamayacağına bakıyorum. Eğer işaretlediğim matriste değerler birbirine eşitse bağlanabilir. Eşit değilse bağlanamaz, hamle yanlıştır, geri almam lazım.

```
push(i, j, i-1, j, top, move, 0, -1);
automatic(a, i-1, j, N, matris, index, move, top, label, c); //yukarı


push(i, j, i+1, j, top, move, 0, 1);
automatic(a, i+1, j, N, matris, index, move, top, label, c); //aşağı


push(i, j, i, j+1, top, move, 1, 1);
automatic(a, i, j+1, N, matris, index, move, top, label, c); //sağ


push(i, j, i, j-1, top, move, 1, -1);
automatic(a, i, j-1, N, matris, index, move, top, label, c); //sol
```

Sonraki kısımda ise bir sonraki hamleyi belirleyip recursive olarak fonksiyonu tekrar çağırıyorum.(push manuelle aynı)

```
if(control(matris, N) && !chokepoint(N, label, matris, c, index)){
    return;//çözüm
}

else{
    *top -= 1;
    undo(top, matris, move);
    return;
}
```

Son kısımda ise matrisin dolup dolmadığına ve bütün sayıların eşleşip eşleşmediğini kontrol ediyorum. Eğer şartlar sağlanmıyorsa hamleleri geri alıyorum.

```
scoreFunc(choice, choice2, N, k, score, numUndo);
```

Ayrıca oyun bitiminde skor hesaplamasını yapıyorum.

```
void scoreFunc(int choice, int choice2, int N, int k, int score[], int numUndo){
    int s=0;

    if(choice == 1){
        s += 10;
    }

    else if(choice == 2){
        s += 5;
    }

    if(choice2 == 1){
        s += 15 - numUndo;
    }

    else if(choice2 == 2){
        s += 5;
    }

    s += 3 * N;

    return ;
}
```

Rastgele oluşturma +10

Dosyadan oluşturma +5

Manuel mod +15 + (manuel mod için geri alma * -1)

Otomatik mod +5

Boyut*3

1.2 DOSYADAN MATRİS OLUŞTUR

Rastgele matris oluşturmadan tek farkı matris oluşturma kısımları. Manuel ve otomatik mod aynı olduğu için tekrar açıklamayacağım.

```
if(choice == 2){ //dosyadan oku

    do{
        choice2 = gameMenu();

        if(choice2 != 3){
            printf("\nMatrisin boyutu: ");
            scanf("%d", &N);
            reset(matris, N);
            printf("\nDosya Adini Giriniz: ");
            scanf("%s",fileName);
            readFromFile(matris, fileName);
            drawBoard(matris,N);
        }

        if(choice2 == 1){ //manuel mod
            manuel(matris, move, N, &numUndo, label, c, index);
        }

        if(choice2 == 2){//otomatik mod
            findIndex(matris, index, N);
            i = index[0][0];
            j = index[0][1];
            top = -1;
            a = 1;
            automatic(a, i, j, N, matris, index, move, &top, label, c);
            if(control(matris, N) && !chokepoint(N, label, matris, c, index)){
                drawBoard(matris, N);
            }
            else{
                printf("\n!!Bu matrisin cozumu yok!!\n");
            }
        }
        scoreFunc(choice, choice2, N, k, score, numUndo);
    }while(choice2 != 3);
```

```
void readFromFile(int matris[][MAX], char *fileName){
    int i,j, temp;
    FILE *data = fopen(fileName,"r");
    if(!data){
        printf("Dosya Acilamadi!");
        return;
    }
    while(!feof(data)){
        fscanf(data,"%d %d %d\n",&i,&j,&temp);
        matris[i][j]=temp;
    }
    fclose(data);

}
```

Dosyadan okumak için verilen kod

1.3 KULLANICI SKORLARI

Ana menüde seçildikten sonra oyuncular ve skorlarını yazar.

```
if(choice == 3){ //skorları yazdırma
    printScore(score, user, k);
}
```

```
void printScore(int score[], char user[][20], int k){
    int i;

    if(k == -1){
        printf("\nListelenecek kullanıcı ve skoru yok!");,
    }

    for(i=0 ; i<k ; i++){
        printf("\n%d) %s  --->  %d", i+1, user[i], score[i]);
    }

    return;
}
```

```
SAYI ESLESTIRME

Manuel Oyun Kurallari:
1)Baslangic noktanizda ilerletmek istediginiz say² bulunmalidir!!
2)Noktayi her adimda saga, sola, asagiya ya da yukariya bir dogru boyunca ilerletmelisiniz!!
3)Ilerleme boyunca baska bir sayi bulunmamali!!

Yapmak istediginiz islemi seciniz:
1)Rastgele Matris Olustur
2)Dosyadan Matris Olustur
3)Kullanicilarin Skorlarini Goster
4)Cikis
Secim: 2

1. oyuncu
Adinizi giriniz: sude


Yapilacak islemi seciniz:
1)Manuel Modda Oyna
2)Otomatik Modda Oyna
3)Ana Menuye Don
Secim: 2

Matrisin boyutu: 5

Dosya Adini Giriniz: 5x5_test_case2.txt

-----------------------------------
      |      |      |  1   |  2   |
-----------------------------------
      |  3   |      |      |      |
-----------------------------------
      |      |      |  4   |  5   |
-----------------------------------
  1   |  3   |  2   |      |      |
-----------------------------------
  4   |      |      |      |  5   |
```

```
Matrisin boyutu: 5

Dosya Adini Giriniz: 5x5_test_case2.txt

_____________________________
     |     |     |  1  |  2  |
_____________________________
     |  3  |     |     |     |
_____________________________
     |     |     |  4  |  5  |
_____________________________
  1  |  3  |  2  |     |     |
_____________________________
  4  |     |     |     |  5  |

_____________________________
  1  |  1  |  1  |  1  |  2  |
_____________________________
  1  |  3  |  2  |  2  |  2  |
_____________________________
  1  |  3  |  2  |  4  |  5  |
_____________________________
  1  |  3  |  2  |  4  |  5  |
_____________________________
  4  |  4  |  4  |  4  |  5  |
```

```
Yapilacak islemi seciniz:
1)Manuel Modda Oyna
2)Otomatik Modda Oyna
3)Ana Menuye Don
Secim: 1

Matrisin boyutu: 5

Dosya Adini Giriniz: 5x5_test_case2.txt

-----------------------------------
      |      |      |  1   |  2   |
-----------------------------------
      |  3   |      |      |      |
-----------------------------------
      |      |      |  4   |  5   |
-----------------------------------
  1   |  3   |  2   |      |      |
-----------------------------------
  4   |      |      |      |  5   |

Yapilacak islemi seciniz:
1)Hamle yap
2)Hamleyi geri al
3)Pes et ve oyun menusune don
Secim: 1
```

```
Baslangic noktasi (x,y): 3 0

Bitis noktasi (x,y): 0 0

-----------------------------------
  1   |      |      |  1   |  2   |
-----------------------------------
  1   |  3   |      |      |      |
-----------------------------------
  1   |      |      |  4   |  5   |
-----------------------------------
  1   |  3   |  2   |      |      |
-----------------------------------
  4   |      |      |      |  5   |

Yapilacak islemi seciniz:
1)Hamle yap
2)Hamleyi geri al
3)Pes et ve oyun menusune don
Secim: 1
```

```
Yapilacak islemi seciniz:
1)Hamle yap
2)Hamleyi geri al
3)Pes et ve oyun menusune don
Secim: 1


Baslangic noktasi (x,y): 0 0

Bitis noktasi (x,y): 0 3

Sayilar eslestirildi!
-----------------------------------
  1   |  1   |  1   |  1   |  2   |
-----------------------------------
  1   |  3   |      |      |      |
-----------------------------------
  1   |      |      |  4   |  5   |
-----------------------------------
  1   |  3   |  2   |      |      |
-----------------------------------
  4   |      |      |      |  5   |
```

```
Baslangic noktasi (x,y): 1 1

Bitis noktasi (x,y): 1 2

 ---------------------------------
|  1  |  1  |  1  |  1  |  2  |
 ---------------------------------
|  1  |  3  |  3  |     |     |
 ---------------------------------
|  1  |     |     |  4  |  5  |
 ---------------------------------
|  1  |  3  |  2  |     |     |
 ---------------------------------
|  4  |     |     |     |  5  |

Yapilacak islemi seciniz:
1)Hamle yap
2)Hamleyi geri al
3)Pes et ve oyun menusune don
Secim: 2

 ---------------------------------
|  1  |  1  |  1  |  1  |  2  |
 ---------------------------------
|  1  |  3  |     |     |     |
 ---------------------------------
|  1  |     |     |  4  |  5  |
 ---------------------------------
|  1  |  3  |  2  |     |     |
 ---------------------------------
|  4  |     |     |     |  5  |
```

```
Yapilacak islemi seciniz:
1)Hamle yap
2)Hamleyi geri al
3)Pes et ve oyun menusune don
Secim: 1


Baslangic noktasi (x,y): 3 3

Bitis noktasi (x,y): 1 2

!!Noktayi sadece bir dogru boyunca hareket ettirebilirsiniz!!
```

```
Baslangic noktasi (x,y): 4 3

Bitis noktasi (x,y): 2 3

Sayilar eslestirildi!
---------------------------------
  1   |  1   |  1   |  1   |  2   |
---------------------------------
  1   |  3   |  2   |  2   |  2   |
---------------------------------
  1   |  3   |  2   |  4   |  5   |
---------------------------------
  1   |  3   |  2   |  4   |  5   |
---------------------------------
  4   |  4   |  4   |  4   |  5   |
```