VERİ YAPILARI VE ALGORİTMALAR
PROJE ÖDEVİ

AD: KUTAY

SOYAD: ALPTEKİN

NUMARA: 20011615

DERSİ GRUBU: GRUP 3 - GÖKSEL BİRİCİK

VİDEO LİNKİ: https://youtu.be/c-LsK-Qf61w

Ders proje ödevi olarak bizden labirent içerisinde başlangıç hücresinden labirentin bitiş hücresine olan yolu bulmamız, bu yol esnasında belirli puan değişiklikleri yapmamız ve bu yolu animasyon ile göstermemiz istendi. Öncelikle problemi daha iyi anlamak ve çözüm yolu üzerinde daha iyi düşünebilmek için internette biraz araştırma yaptım. Problemin labirent çözme algoritmaları içerisinde yer aldığını ve farklı algoritmik yaklaşımlar olduğunu gördüm. Dokümandaki nottan da yararlanarak problemin dfs ile çözülen bir graph arama algoritması yaklaşımı ile çözülebileceği kanısına vardım. Çözüm yolu olarak, başlangıçta dosyadan karakter karakter okuduğum labirenti engelleri 0, gidilebilen yolları 1 olacak şekilde daha önceden oluşturmuş olduğum matrisime kayıt ettim, ayrıca labirentin başlangıç ve bitiş koordinatlarını da bu sırada ayrı değişkenlere atadım. Son olarak rastgele bir şekilde bu labirentin içine elmalar ekledim ve labirentimi başarılı bir şekilde okuyup kayıt etmiş oldum.Başlangıç koordinatlarını dosyadan okuyup belirli değişkenlere kayıt ettiğim labirentin başlangıç noktasından dfs ile sürekli olarak gidebileceği yerlere kadar götürüp,çıkmaz bir yola girdiğinde geriye giderek gidebileceği bir önceki yolu denemesi ve bu hareketi sürekli olarak rekürsif bir şekilde tekrar etmesi yaklaşımını izledim.

Şimdi algoritmamı ve kodlarımı tanıtacağım.

```
printf("labirentin satir ve sutun sayisini giriniz:\n");
scanf("%d %d",&n,&m);
matris = (int**)malloc(sizeof(int*)*n);
for(i=0;i<m;i++){
    matris[i] = (int*)calloc(m,sizeof(int));
}
visited = (int**)malloc(sizeof(int*)*n);
for(i=0;i<m;i++){
    visited[i] = (int*)calloc(m,sizeof(int));
}
```

Burada öncelikle labirentin satır ve sütun sayısını alarak 0 ve 1 lerden oluşacak matrisimi ve daha önce gidilen yollara gidilmemesi için visited matrislerimi oluşturdum.

```
while ((c = getc(fptr))!= EOF)
{

    if((char)c=='-' || (char)c=='|'|| (char)c=='+')
    {
        matris[i][j]=0;
        j++;
    }
    else if(c=='\n')
    {
        i++;

        if(hlpr){
            m=j;
            hlpr=0;
        }
        j=0;
    }

    else if(c==' ')
    {
        matris[i][j]=1;
        j++;
    }
    else if(c=='b')
    {
        startx=j;
        starty=i;
        matris[i][j]=1;
        j++;
    }
    else if(c=='c')
    {
        endx=j;
        endy=i;
        matris[i][j]=1;
        j++;
    }
}
n=i;
```

Burada ise dosyadan karakter karakter okuduğum labirenti eğer duvarsa 0,boşluksa 1 olarak atadım. eğer b(başlangıç) ve c(bitiş) karakterlerini okuduysam o koordinatları kaybetmemek için belirli değişkenlerde kayıt ettim. Eğer satır sonuna geldiysem bir sonraki satıra geçtim ve 1 kereliğine mahsus olarak sütun sayısını m'ye ve satır sayısını n'ye kaydettim.

```c
void createRandomApples(int **matris,int **visited,int n,int m)
{
    int i,j,k=0,apple;
    printf("eklenecek elma sayisini giriniz:\n");
    scanf("%d",&apple);

    while(k<apple){
        i=rand()%n;
        j=rand()%m;
        if(matris[i][j]==1)
        {
            visited[i][j]=2;
            matris[i][j]=248;
            k++;
        }
    }
}
```

Daha sonra kullanıcıdan aldığım elma sayısını rastgele bir şekilde labirente ekledim. Buradaki 248 sayısı elma karakterinin sayısal karşılığını ifade ediyor. Elmanın visitedlerini ise 2 olarak kayıt ettim. Elmayı topladığı zaman visitedleri 1 olacak ve elma kayıp olacak. 0 yapmama sebebim 0 sayısını gezinirken gidilen yola dönmeme kontrolünü yaparken kullanıyorum.

```c
int recursiveSolve(int x, int y,int endX,int endY,int **matris,int **visited,int *puan,int n,int m) {
    if (x == endX && y == endY){
        printf("\n\ncikis bulundu!!");
        return 1;
    }

   usleep(699999);

    visited[y][x]=1;
     system("cls");
    printMatris(visited,matris,n,m,puan);
     if(x+1<m && visited[y][x+1]!=1 && matris[y][x+1])
     {
        if(visited[y][x+1]==2){
            (*puan)+=10;
        }
        if(recursiveSolve(x+1,y,endX,endY,matris,visited,puan,n,m)){
            return 1;
        }
     }
     else if(!matris[y][x+1] && !matris[y+1][x] && !matris[y-1][x]) {
        (*puan)-=5;
     }


      if(x-1>0 && visited[y][x-1]!=1 && matris[y][x-1])
     {
        if(visited[y][x-1]==2){
            (*puan)+=10;
        }
        if(recursiveSolve(x-1,y,endX,endY,matris,visited,puan,n,m)){
            return 1;
        }

     }
     else if(!matris[y][x-1] && !matris[y+1][x] && !matris[y-1][x])
     {
        (*puan)-=5;
      }
```

```
    if(y+1<n && visited[y+1][x]!=1 && matris[y+1][x])
     {
        if(visited[y+1][x]==2) {
            (*puan)+=10;
         }
        if(recursiveSolve(x,y+1,endX,endY,matris,visited,puan,n,m)) {
            return 1;
         }
     }
     else if(!matris[y+1][x] && !matris[y][x+1] && !matris[y][x-1])
     {
        (*puan)-=5;
      }

     if(y-1>0 && visited[y-1][x]!=1 && matris[y-1][x])
     {
        if(visited[y-1][x]==2) {
            (*puan)+=10;
         }
        if(recursiveSolve(x,y-1,endX,endY,matris,visited,puan,n,m)) {
            return 1;
         }
     }
     else if(!matris[y-1][x]&& !matris[y][x-1]&& !matris[y][x+1])
     {
        (*puan)-=5;
      }
     visited[y][x]=0;

    return 0;
}
```

Algoritmam ise burada yer alıyor. Eğer x ve y bitiş koordinatlarına ulaştıysa bulundu mesajını veriyorum. Öncelikle if kontrollerimin ilk kısmında yer alan kontrolleri yapıp hala labirentin içinde olduğumu ve sınırların dışına çıkmadığımı teyit ediyorum. Daha sonra eğer gideceğim yer ziyaret edilmediyse ve herhangi bir engel yoksa kontrolün içine giriyorum eğer bir elmaya ulaştıysam puanımı 10 artırıyor,rekürsif bir şekilde kendimi çağırmaya devam ediyorum ve gidebileceğim yere kadar gidiyorum.

Eğer engelle karşılaşırsam gidebileceğim yer var mı kontrolünü yapıyorum çıkmaz bir yola girdiysem puanımı 5 azaltıyorum ve geriye giderek bir önceki gidilebilecek yerden yoluma devam ediyorum.

Her rekürsif çağrıda konsol satırını silip labirentimi tekrardan yazdırıyorum. Bu şekilde labirentteki ilerleyiş görünür hale geliyor.