YILDIZ TEKNİK ÜNİVERSİTESİ
ELEKTRİK-ELEKTRONİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ
BÖLÜMÜ

Bilgisayar Bilimlerine Giriş Dönem Projesi:
TRİVERSİ OYUNU

MUHAMMED EMİR GÜL-23011051

Öğretim Görevlisi

Dr. Öğr. Üyesi GÖKSEL BİRİCK

İSTANBUL

2025

ALGORİTMANIN ADIMLARI

1-Kullanıcıdan oyun tahtasının boyutlarını alma ve oyun tahtasını ekrana yazdırma

2-Taşların kullanıcıdan sırayla (K-M-S) alınması

3-İlk kırmızı taşın merkeze en yakın noktaya koyulup koyulmadığının kontrolü

4-Girilen taşların diğer taşlarla yatay/dikey/çapraz olarak temas etmesinin kontrolü

5-Zaten taş olan bir koordinatın üzerine farklı bir taş koymaya çalışınca uyarı vermesi

6- Renk değişimi için, yatay/dikey/çapraz sıranın bir açık ucuna koyulan taş ile o sıradaki aynı renkteki diğer en yakın taşın arasındaki diğer renklerinin tamamını koyulan taşın rengine döndürme

7- En son hangi oyuncunun kazandığını ve hangi renkten kaç adet olduğunu ekrana yazdırma

Video linki: https://www.youtube.com/watch?v=cKRJg4EToHU

KULLANICIDAN OYUN TAHTASININ BOYUTLARINI ALMA VE EKRANA YAZDIRMA

```
printf("kare oyun tahtasinin satir sayisini giriniz :");
scanf("%d",&n);



for(i=0;i<n;i++){
    for(j=0;j<n;j++){
        matris[i][j]=nokta;
    }
}
printf("    ");
for(i=0;i<n;i++){
    printf(" %2d",i);
}
printf("\n");
 for(i=0;i<n;i++){
    printf(" %2d ",i);
    for(j=0;j<n;j++){

        printf("%2c ",matris[i][j]);
    }
    printf("\n");
}
```

```
kare oyun tahtasinin satir sayisini giriniz :8
     0  1  2  3  4  5  6  7
  0  .  .  .  .  .  .  .  .
  1  .  .  .  .  .  .  .  .
  2  .  .  .  .  .  .  .  .
  3  .  .  .  .  .  .  .  .
  4  .  .  .  .  .  .  .  .
  5  .  .  .  .  .  .  .  .
  6  .  .  .  .  .  .  .  .
  7  .  .  .  .  .  .  .  .
kirmizi oyuncu, satir ve sutun giriniz:
```

2-While ve For döngüsü yardımıyla kullanıcıdan sırayla kırmızı,mavi ve sarı taşların koordinatlarını isteme

```
for(j=0;j<3;j++){
if(j==0){
oyuncu[0] = 'k';
oyuncu[1] = 'i';
oyuncu[2] = 'r';
oyuncu[3] = 'm';
oyuncu[4] = 'i';
oyuncu[5] = 'z';
oyuncu[6] = 'i';
oyuncu[7] = '\0';
k='K';
}else if(j==1){
oyuncu[0] = 's';
oyuncu[1] = 'a';
oyuncu[2] = 'r';
oyuncu[3] = 'i';
oyuncu[4] = '\0';
k='S';
    }
else if(j==2){
oyuncu[0] = 'm';
oyuncu[1] = 'a';
oyuncu[2] = 'v';
oyuncu[3] = 'i';
oyuncu[4] = '\0';
k='M';
    }

    if(i==0&&j==0){


//ilk koyulan taşın merkeze en yakın nokta olup olmadığını kontrol etme
while(dogruKoordinat==0){
printf("%s oyuncu, satir ve sutun giriniz:\n", oyuncu);
scanf("%d %d",&a,&b);
```

3-ilk koyulan taşın merkeze en yakın nokta olup olmadığını kontrol etme

```
//ilk koyulan taşın merkeze en yakın nokta olup olmadığını kontrol etme
while(dogruKoordinat==0){
printf("%s oyuncu, satir ve sutun giriniz:\n", oyuncu);
scanf("%d %d",&a,&b);
if(i==0&&j==0){

    if(n%2==0){
        if ((a == n / 2 || a == (n / 2) - 1) && (b == n / 2 || b == (n / 2) - 1)){
            dogruKoordinat=1;
        }else{
            printf("hatali koordinat!,tekrar giriniz\n");


        }

    }else{
            if(a==n/2&&b==n/2){
                dogruKoordinat=1;
            }else{
            printf("hatali koordinat!,tekrar giriniz\n");

        }
    }

}
}
```

Burada eğer yanlış koordinat girerseniz sürekli hatalı koordinat diyerek sizden yeniden kırmızı taş için koordinat girmenizi isteyecek ta ki siz doğru koordinatı girene kadar .Doğru koordinatı girdiğinizde ise while döngüsünden çıkıp diğer sürece geçecek.

4-Girilen taşların diğer taşlarla yatay/dikey/çapraz olarak temas etmesinin kontrolü

```
//konulan taşın  diğer taşlara yatay/dikey/çapraz olup olmadığını kontrol etme
while(dogru2==0){

printf("%s oyuncu, satir ve sutun giriniz:\n", oyuncu);
scanf("%d %d",&a,&b);
for(c=(a-1);c<a+2;c++){
    for(z=b-1;z<b+2;z++){
        if(matris[c][z]=='K' ||matris[c][z]=='S' || matris[c][z]=='M' ){
            rakam=1;
        }
    }
}

if(rakam==1){
    dogru2=1;
}else{
    printf("hatali koordinat!,tekrar giriniz.\n");
}
```

Burda da 3. adımdaki çözüm ile aynı mantık var.Eğer girilen koordinatın etrafında K,M veya S taş yoksa hatalı koordinat uyarısı veriyor ve tekrar koordinat girmenizi istiyor.

5-Zaten taş olan bir koordinatın üzerine farklı bir taş koymaya çalışınca uyarı vermesi

```
//taş olan yeri girince hata vermesi
if(matris[a][b]!='.'){
    printf("hatali koordinat!,tekrar giriniz.\n");
    dogru2=0;
}

}
```

6- Renk değişimi için, yatay/dikey/çapraz sıranın bir açık ucuna koyulan taş ile o sıradaki aynı renkteki diğer en yakın taşın arasındaki diğer renklerinin tamamını koyulan taşın rengine döndürme

Ben, burada her düzlemi ayrı ayrı kontrol etmeyi tercih ettim

```
 //yatay sağ kontrol
 int count=0;
 p=b+1;
while(p>=0&&p<n&&count==0){

 if(matris[a][p]!='.'){
     if(matris[a][p]==k){
         for(c=p;c>b;c--){
             matris[a][c]=k;
            }
            count=1;
        }else{
         p++;
        }
    }else{
    count=1;
    }
}
```

```
   //yatay sol kontrol
count=0;
    p=b-1;
  while(p>=0&&p<n&&count==0){

   if(matris[a][p]!='.'){
       if(matris[a][p]==k){
           for(c=p;c<b;c++){
               matris[a][c]=k;
              }
              count=1;
          }else{
           p--;
          }
      }else{
      count=1;
      }
  }
```

```
    //dikey üst kontrol
count=0;
    p=a+1;
   while(p>=0&&p<n&&count==0){

    if(matris[p][b]!='.'){
        if(matris[p][b]==k){
            for(c=p;c>a;c--){
                matris[c][b]=k;
               }
               count=1;
           }else{
            p++;
           }
       }else{
       count=1;
       }
   }
```

```
    //dikey alt kontrol
count=0;
    p=a-1;
   while(p>=0&&p<n&&count==0){

    if(matris[p][b]!='.'){
        if(matris[p][b]==k){
            for(c=p;c<a;c++){
                matris[c][b]=k;
               }
               count=1;
           }else{
            p--;
           }
       }else{
       count=1;
       }
   }
```

```
      //sağ çapraz üst kontrol
count=0;
    p=a-1;
    l=b+1;
   while(p>=0&&p<n&&count==0){

    if(matris[p][l]!='.'){
        if(matris[p][l]==k){

               c=p;
               z=l;
               while(c<a&&z>b){
                matris[c][z]=k;
                c++;
                z--;
               }
               count=1;
           }else{
            p--;
            l++;
           }
       }else{
       count=1;
       }
   }
```

```
      //sağ çapraz alt kontrol
count=0;
    p=a+1;
    l=b-1;
   while(p>=0&&p<n&&count==0){

    if(matris[p][l]!='.'){
        if(matris[p][l]==k){

               c=p;
               z=l;
               while(c>a&&z<b){
                matris[c][z]=k;
                c--;
                z++;
               }
               count=1;
           }else{
            p++;
            l--;
           }
       }else{
       count=1;
       }
   }
```

```
        //sol çapraz alt kontrol
count=0;
    p=a+1;
    l=b+1;
   while(p>=0&&p<n&&count==0){

    if(matris[p][l]!='.'){
        if(matris[p][l]==k){

                c=p;
                z=l;
                while(c>a&&z>b){
                 matris[c][z]=k;
                 c--;
                 z--;
                }
                count=1;
            }else{
             p++;
             l++;
            }
       }else{
       count=1;
       }
   }
```

```
        //sol çapraz üst kontrol
count=0;
    p=a-1;
    l=b-1;
   while(p>=0&&p<n&&count==0){

    if(matris[p][l]!='.'){
        if(matris[p][l]==k){

                c=p;
                z=l;
                while(c<a&&z<b){
                 matris[c][z]=k;
                 c++;
                 z++;
                }
                count=1;
            }else{
             p--;
             l--;
            }
       }else{
       count=1;
       }
   }
```

Burdaki mantık taşın dikey/yatay/çapraz olarak en yakın aynı renkli taşı bulup aradaki renkleri kendi rengine çevirmek.

Oyunun Bitmesi

Oyun tahtasında boş yer kalmadığı anda oyun biter ve sonuçlar ekrana yazdırılır.

```
while(asilsayi==0){

for(j=0;j<3;j++){
```

```
    asilsayi=1;
  for(c=0;c<n;c++){
    for(z=0;z<n;z++){
        if(matris[c][z]==nokta) {

                asilsayi=0;
            }
    }
}
if(asilsayi==1)
j=3;
```

Burda da tahtada boş yer kalıp kalmadığının kontrolünü yapıyoruz ve eğer boş yer kalmamışsa program while döngüsünden çıkıp sonuçları yazdırma kısmına giriyor.

7- En son hangi oyuncunun kazandığını ve hangi renkten kaç adet olduğunu ekrana yazdırma

```
//hangi renkten kaç tane olduğunu bulma
}
int kirmizi=0,sari=0,mavi=0;
 for(i=0;i<n;i++){
    for(j=0;j<n;j++){
    if(matris[i][j]=='K')
        kirmizi++;
    if(matris[i][j]=='M')
        mavi++;
    if(matris[i][j]=='S')
        sari++;

    }
}
```

```
//en çok olan rengi bulma
int max;
max=kirmizi;
if(mavi>max)
max=mavi;

if(sari>max)
max=sari;
```

Burda renklerin adetlerini ve kazanan rengi buluyoruz.

Ve son olarak ekrana bulduklarımızı yazdırma:

```
char dizi5[8] = "kirmizi";
char dizi6[5]="mavi";
char dizi7[5]="sari";

//kazananı ve sonuçları yazdırma

if(max==kirmizi){
printf("KAZANAN RENK :%s\n\n",dizi5);
    }

if(max==mavi){
printf("KAZANAN RENK :%s\n\n",dizi6);
    }

if(max==sari)
printf("KAZANAN RENK :%s\n\n",dizi7);


printf("%d  adet sari bulunmaktadır.\n",sari);
    printf("%d  adet kirmizi bulunmaktadır.\n",kirmizi);
    printf("%d  adet mavi bulunmaktadır.",mavi);
```