

MANİSA CELAL BAYAR ÜNİVERSİTESİHASAN FERDİ TURGUTLU TEKNOLOJİ FAKÜLTESİ  
Yazılım Mühendisliği

YZM 3111 – Yazılım Sınama  
Ödev Dokümantasyonu

Ödevin Adı

Ödev 1

Hazırlayan

Hüseyin Yasin DÖNMEZ 202802003

Tuba AKBAŞ 212802004

EKİM –2021

1. **Amaç**

Satranç tahtası üzerinde Piyonların aşağıdaki diziliş kurallarına göre kaç farklı şekilde dizileceğinin tespit edilmesi.

1. İlk piyon siyah olamaz
2. İki siyah piyon yan yana gelemez
3. İki beyaz piyon yan yana gelebilir
4. Bir siyah piyon ve bir beyaz piyon yan yana gelebilir tam tersi de geçerlidir.
5. **Girdiler**

static int birSatirIcinIstenilenOlasilikSayisi = 0;// kurallara uyan olasiliklarin sayisini tutmak ve her yerden erismek icin kullanildi

int i sayi; // dizisinin içerisindeki elemanları arasında gezinmek için kullanıldı

int kalan; //cevirme icin kullanilan sayinin 2'ye kalanini tutmak icin kullanildi ikili sisteme çevirmek için gerekli

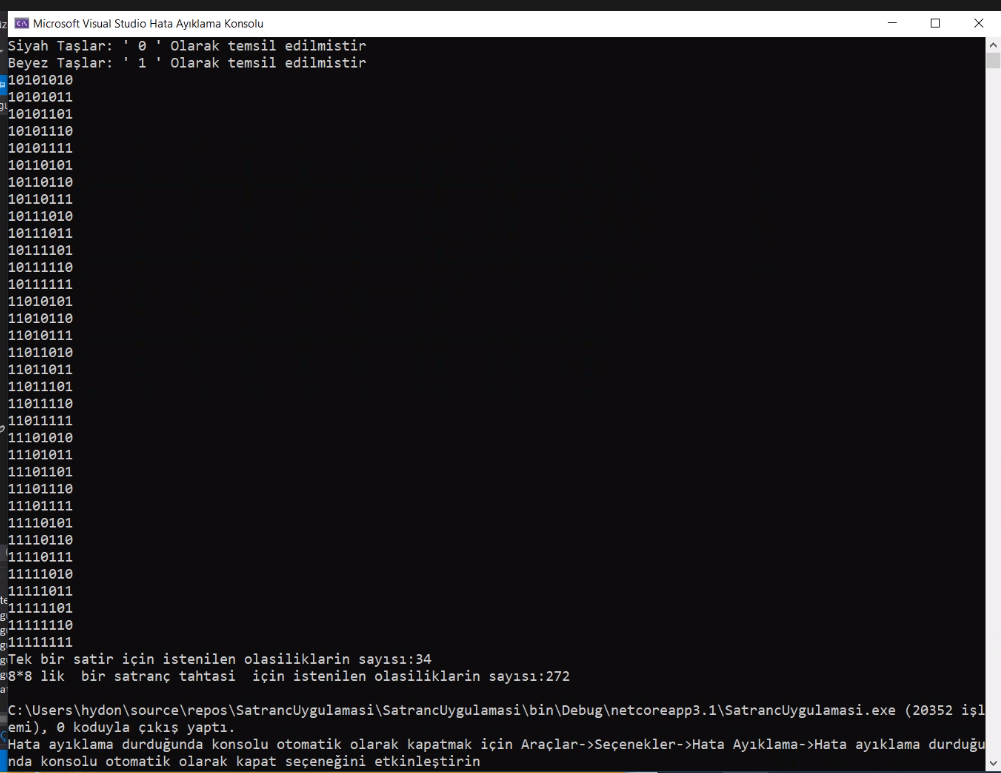
int kacinci = 0; //cevrilen sayinin kacinci kalani oldugunu belirlemek için

int tumOlasiliklar=2^n n=sutun sayisi olmak üzere tüm olasılıkları ifade eder başlangicta tüm olasılıklar 0 dan başlar ve 8 kare olduğu için 256 sayısına kadar devam eder

1. **Mantığı**

Beyaz taşlar 1 siyah taşlar 0 olarak düşünülmüştür ve sayısal tasarım dersinde doğruluk tablolarının tüm olasılıkları 2n formülünden yararlanılarak bulunduğundan ve satranç tahtasının bir satırı 8 kare olduğu için toplam 28 = 256 ihtimal olduğu hesaplanmıştır. Bu yüzden 0 dan 256 ‘ya kadar olan sayılar bir for döngüsü içinde yazılarak her sayı **IkilikSayiSistemineCevir()** adli fonksiyona parametre olarak gönderilmiştir. Bu fonksiyonda **kalan** ve **kacinci** adli iki adet değişken vardır. **kalan** değişkeni sayının 2 ile bölümünden kalanı tutmaktadır. **kacinci** ise bir diziye kalanları yazabilmek için 0 dan başlatılarak while döngüsü döndüğü sürece artırılmış bu sayede her kalan bir indikse kayıt edilmiştir. Bu kısımda bilinmesi gereken önemli kısım c#’da int dizilerin varsayılan değerlerinin 0 olmasıdır bu sayede 8 den az kalanı olan sayıların diğer kısımları sıfır olarak atanmıştır. Örnek olarak 1 için 00000001 bu işlemin ise bir dizi olarak sayının 2’lik tabanda karşılığını bulmak için sayi 0’ a eşit olana kadar while döngüsüne sokulmuş burada sırasıyla kalan değişkende sayının 2’ile bölümünden kalan hesaplanarak tutulmuştur. Sonrasında sayı 2’ye bölünmüş ve oluşan sonuç **dizinin[kalan]** binary sisteme çevrildikten sonra 8 elemanlı bir tek boyutlu dizide tutularak paremetre olarak integer bir dizi alan **KontrolEt()** fonksiyona gönderilmiştir. Bu fonksiyonda for döngüsü içinde dizinin 0 elemanından başlanılarak dizinin length’ i kadar dönülmüştür. Bu kısımda if yapılarından yararlanılarak dizinin birinci elemanın 0 ve ardışık iki elemanın 0 olmaması şartını sağlayan tüm durumlar ekrana yazdırılmıştır ve şart sağlandığında **birSatirIcinIstenilenOlasilikSayisi** adlı static değişkenin bir değeri artırılmış ve bulunan sonuç 8\*8’ lik bir tahta olduğu için 8 ile çarpılarak **tumSatirIcinIstenilenOlasilikSayisi** değişkenine atanarak ekrana yazdırılmıştır.

1. **Kodlar**
2. using System;
3. namespace SatrancUygulamasi
4. {
5. class Program
6. {
7. static int birSatirIcinIstenilenOlasilikSayisi = 0;// kurallara uyan olasiliklarin sayisini tutmak ve her yerden erismek icin kullanildi
8. static public void KontrolEt(int[] dizi) //olusturulan sayi dizisi parametre olarak alinip istenilen kurallara uyup uymadigi kontrol ediliyor
9. {
10. if (dizi[0] == 0) //dizinin ilk elemaninin 0 olması ve iki sifirin yan yana gelmesi istenmiyor diger tum durumlar istenilen durumlardir
11. {
12. }
13. else if (dizi[0] == 0 && dizi[1] == 0)
14. {
15. }
16. else if (dizi[1] == 0 && dizi[2] == 0)
17. {
18. }
19. else if (dizi[2] == 0 && dizi[3] == 0)
20. {
21. }
22. else if (dizi[3] == 0 && dizi[4] == 0)
23. {
24. }
25. else if (dizi[4] == 0 && dizi[5] == 0)
26. {
27. }
28. else if (dizi[5] == 0 && dizi[6] == 0)
29. {
30. }
31. else if (dizi[6] == 0 && dizi[7] == 0)
32. {
33. }
34. else
35. {
36. for (int i = 0; i < dizi.Length; i++)
37. {
38. Console.Write(dizi[i]);//kurallara uyan dizinin elemanlari ekrana yazdirilir
39. }
40. Console.WriteLine("");
41. birSatirIcinIstenilenOlasilikSayisi++;//kurallara uyan siralislarin sayisini bir artiriliyor
42. }
43. }
44. static public int[] IkilikSayiSistemineCevir(int sayi)
45. {
46. int kalan; //cevirme icin kullanilan sayinin 2'ye kalanini tutmak icin kullanildi ikili sisteme çevirmek için gerekli
47. int[] sayilar = new int[8];// cevrilen sayinin 2' lik halinde islem yapmak icin diziye atilmasi gerekir girilmeyen değerler default olarak 0 dir
48. int kacinci = 7; //cevrilen sayinin kacinci kalani oldugunu belirlemek icin
49. while (sayi != 0)//sayin 1'den kücük olduğunda çevirme işlemi tamamlanmıştır
50. {
51. kalan = sayi % 2;
52. sayi = sayi / 2;
53. sayilar[kacinci] = kalan; //2' lik sistemde kalanlar sirayla diziye atılır
54. kacinci--;//bir sonraki indikse atlanir
55. }
56. return sayilar; //elde edilen 2'lik sisteme donusturulmus sayi dizisi geri gonderilir
57. }
58. static void Main(string[] args)
59. {
60. Console.WriteLine("Siyah Taşlar: ' 0 ' Olarak temsil edilmistir");
61. Console.WriteLine("Beyez Taşlar: ' 1 ' Olarak temsil edilmistir");
62. for (int tumOlasiliklar = 0; tumOlasiliklar < 256; tumOlasiliklar++)
63. {
64. KontrolEt(IkilikSayiSistemineCevir(tumOlasiliklar)); //tek bir satir icin doğruluk tablosundan yararlanılarak 2^8 den =256 olasılık olduğu tespit edilmistir
65. //sayı 0 den başlayarak 256'ya kadar sayilar 2'lik sayi sistemine cevrildi
66. }
67. int tumistenilenolasilikdurumlari = birSatirIcinIstenilenOlasilikSayisi \* 8;//Bir satir için istenilen olasilik sayisi 8 ile carpilirsa tum tahta icin olasilik sayisi bulunur
68. Console.WriteLine("Tek bir satir için istenilen olasiliklarin sayısı:" + birSatirIcinIstenilenOlasilikSayisi);
69. Console.WriteLine("8\*8 lik bir satranç tahtasi için istenilen olasiliklarin sayısı:" + tumistenilenolasilikdurumlari);
70. }
71. } }
72. **Ekran Çıktısı**



1. **Test Case**

|  |  |
| --- | --- |
| **Risk Level-Risk Seviyesi** | Yüksek |
| **Purpose-Amaç** | Gerçekleştirilen programın doğru bir şekilde çalıştığının gösterilmesi |
| **Inputs- Girdiler** | **-** |
| **Expected Output-Beklenen Sonuç** | Tek bir satır için= 34 Tüm tahta için 272 |
| **Pass Criterias- Geçme Kriterleri** | Tek bir satır için= 34 Tüm tahta için 272 |
| **Fail Criterias- Kalma Kriterleri** | Tek bir satır için= 34 ve Tüm tahta için 272 dışındaki tüm sonuçlar |
| **Test Procedure** | 1. Programı çalıştır. 2. Sonuca bak 3. Sonucu beklenen sonuçla karşılaştır. 4. Beklenen sonuç ile çıkan sonuç aynı ise testi başarılı olarak isimlendir. |
| **Test Durumu** | **Geçti** |