YMH412 YAZILIM KALİTE GÜVENCESİ VE TESTİ – BEYAZ KUTU TESTİ

Prof. Dr. Resul Daş
Arş. Gör. T. Burak Alakuş

BEYAZ KUTU TESTİ NEDİR?

- Literatürde beyaz kutu, cam kutu, saydam kutu gibi anlamlarla ifade edilen beyaz kutu testi, sistemin iç yapısını bilen kişiler tarafından yapılan yazılım testidir.
- Genellikle yazılım mühendisleri ve geliştiriciler tarafından yapılır.

BEYAZ KUTU TESTİ NEDİR?

- Geliştiriciler ya da yazılım mühendisleri, **birim testler** hazırlayarak beyaz kutu test sürecini başlatırlar.
- Beyaz kutu testinde, projenin **hem kaynak kodu hem de derlenmiş kodu** test edilir.

BEYAZ KUTU TESTİ NEDİR?

- Beyaz kutu testi, uygulama kodunun iç mantığı üzerindeki bilgiye bağlıdır.
- Yazılım kodundaki **deyimler, akış denetimleri, koşullar** vb elemanlar sınanır.

BEYAZ KUTU TESTİ NEDİR?

- Beyaz kutu test tasarım tekniği **veri akışlarına, kontrol akışlarına, ifade kapsama, dal kapsama** gibi konulara odaklanır.

BEYAZ KUTU TESTİ HANGİ SEVİYELERDE GERÇEKLEŞTİRİLİR?

- Beyaz kutu testleri **birim, tümleştirme ve sistem test** seviyelerinde gerçekleştirilebilir.
- **Birim test seviyesinde gerçekleştirilen beyaz kutu testleri** birim tümleştirme öncesinde **birimdeki hataları bulmayı amaçlar.**

BEYAZ KUTU TESTİ HANGİ SEVİYELERDE GERÇEKLEŞTİRİLİR?

- Tümleştirme seviyesindeki beyaz kutu testlerinde ise, **modüllerin birbiri ile iletişiminde** ortaya çıkabilecek olan hataları bulmak hedeflenir.

BEYAZ KUTU TESTİ HANGİ SEVİYELERDE GERÇEKLEŞTİRİLİR?

- Sistem seviyesinde gerçekleştirilen beyaz kutu testlerinde ise **amaç kapsama analizlerinin (ifade, kod, gereksinim kapsama vb) gerçekleştirilmesidir.**

BEYAZ KUTU TESTİ NE ZAMAN YAPILIR?

- Beyaz kutu testlerinde en önemli **nokta kod bilgisidir.**
- Bu testlerin yazılım yaşam döngüsünün erken safhalarında yapılması ayrı önem taşımaktadır.

BEYAZ KUTU TESTİ NE ZAMAN YAPILIR?

- Beyaz kutu testleri yaşam döngüsünün erken aşamalarında gerçekleştiği için yazılım içindeki hatalarda erken safhada bulunmuş olur.
- Bu sayede **düzeltme maliyeti düşük olmaktadır.**

BEYAZ KUTU TESTİNİN AVANTAJLARI VE DEZAVANTAJLARI

- Beyaz kutu testinin avantajları;
  - Kaynak kodun yan etkileri saptanır.
  - Kod optimizasyonu yapılır.
  - Geliştiricilere kendilerini geliştirmek için fırsat verir.

BEYAZ KUTU TESTİNİN AVANTAJLARI VE DEZAVANTAJLARI

- Beyaz kutu testinin dezavantajları ise;

- Testi yapan kişi testi yaparken **önyargılı bir şekilde davranabilir.** Bu durumda gözden kaçan ya da değerlendirilmek istenmeyen hatalar testin güvenirliğini sarsabilir.

BEYAZ KUTU TESTİNİN AVANTAJLARI VE DEZAVANTAJLARI

- Testi yapan kişi test mühendisi olacaksa, kaynak kodun ona anlatılması gerekir ve bu durumda **bilgi transferi maliyetli olmaktadır.**

BEYAZ KUTU TESTİ VE KARA KUTU TESTİ ARASINDAKİ FARKLAR

- Kara kutu ve beyaz kutu testinde sistemin yapısı, tasarımı ve uygulama tekniği test edilmektedir.
- Ancak kara kutu testinde testi yapan kişinin bunların içeriğini bilmesine gerek yokken, beyaz kutu testinde testi yapan kişinin uygulamaya tamamen hakim olması gerekmektedir.

BEYAZ KUTU TESTİ VE KARA KUTU TESTİ ARASINDAKİ FARKLAR

- **Hangi seviyelerde gerçekleştirilir?**
- Kara kutu testi yüksek seviyelerde (Kullanıcı Kabul Testi, Sistem Testi) uygulanırken,
- Beyaz kutu testi düşük seviyelerde (Birim Testi, Tümleştirme Testi, Sistem Testi) uygulanmaktadır.

BEYAZ KUTU TESTİ VE KARA KUTU TESTİ ARASINDAKİ FARKLAR

- **Testi kimler gerçekleştirmektedir?**
- Kara kutu testi genellikle **bağımsız yazılım testçileri** tarafından gerçekleştirilirken,
- Beyaz kutu testi genellikle **yazılım geliştiricileri** tarafından gerçekleştirilir.

BEYAZ KUTU TESTİ VE KARA KUTU TESTİ ARASINDAKİ FARKLAR

- **Programlama bilgisi**
- Kara kutu testinde test yapılırken herhangi bir **programlama bilgisine gerek yokken**,
- Beyaz kutu testinde **programlama bilgisine gerek vardır.**

BEYAZ KUTU TESTİ VE KARA KUTU TESTİ ARASINDAKİ FARKLAR

- **Uygulama bilgisi**
- Kara kutu testinde test yapılırken **herhangi bir uygulama bilgisine gerek yokken,**
- Beyaz kutu testinde **uygulama bilgisine gerek vardır.**

BEYAZ KUTU TESTİ VE KARA KUTU TESTİ ARASINDAKİ FARKLAR

- **Test senaryoları neye dayanır?**
- Kara kutu testinde test senaryoları, **gereksinim özelliklerine** dayanırken,
- Beyaz kutu testinde, **detaylı tasarıma** dayanmaktadır.

BEYAZ KUTU TESTİ NASIL YAPILIR?

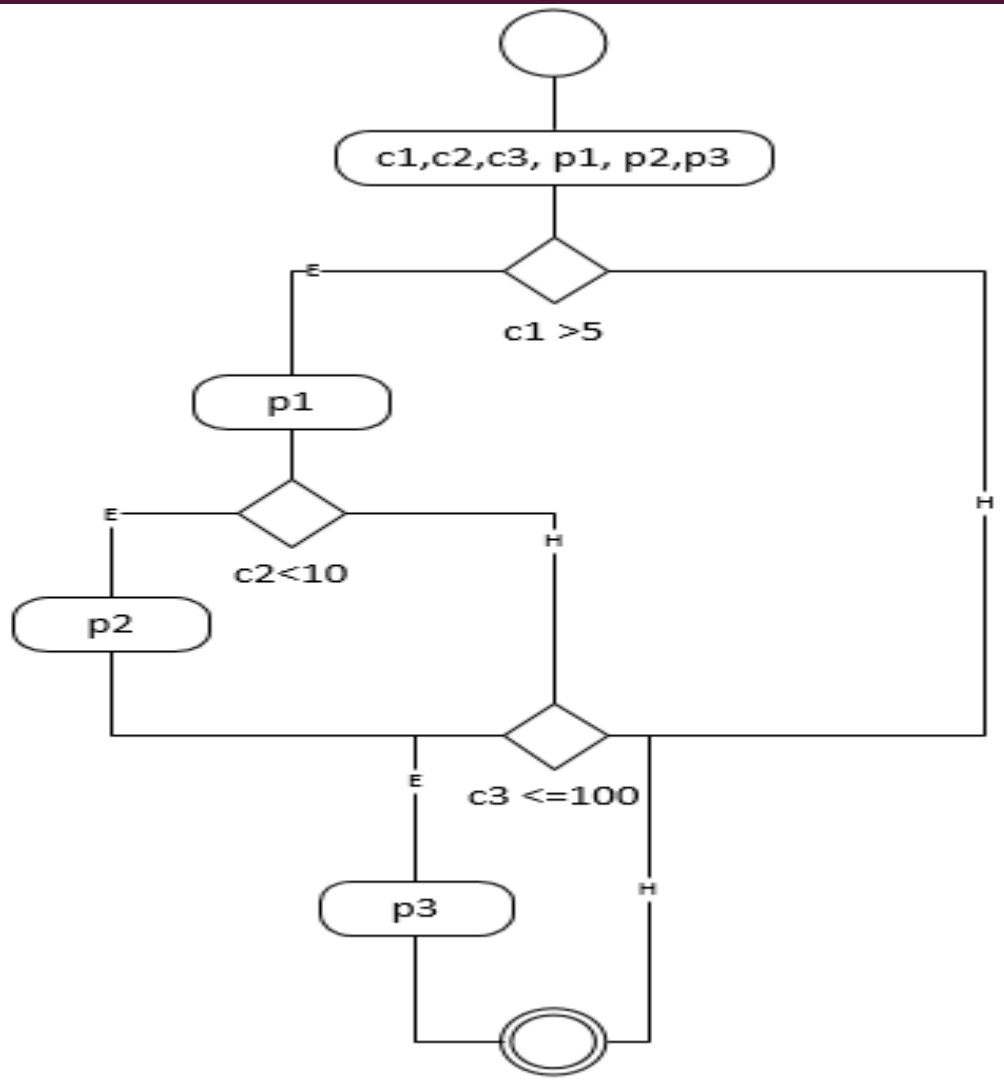
- İlk olarak test edilecek kodun akış diyagram oluşturulur.
- Ardından tüm yollar belirlenir ve yollara değerler verilir.
- Son olarak belirlenen yollar gerekli değerlerle test edilir.

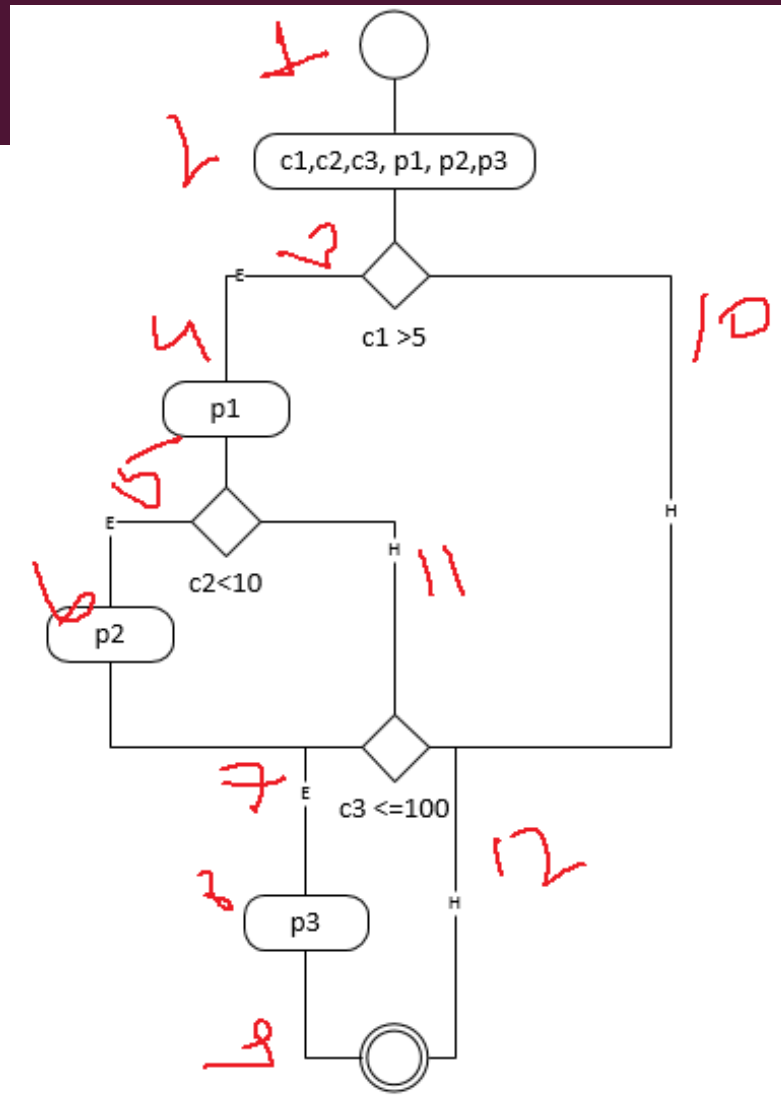
ÖRNEK UYGULAMALAR #1

```
static void Main(string[] args)
{
    Console.WriteLine("Lütfen üç adet tamsayı değeri giriniz: ");
    string a, b, c;
    a = Console.ReadLine();
    b = Console.ReadLine();
    c = Console.ReadLine();
    int c1 = Convert.ToInt32(a);
    int c2 = Convert.ToInt32(b);
    int c3 = Convert.ToInt32(c);

    int p1, p2, p3;

    if (c1 > 5)
    {
        p1 = c1 + 7;
        if (c2 < 10)
            p2 = c2 - 5;
    }
    if (c3 <= 100)
        p3 = c3 * 40;
}
```


c1,c2,c3, p1, p2,p3
E
c1 >5
H
p1
E
c2<10
H
p2
E
c3 <=100
H
p3


c1,c2,c3, p1, p2,p3
c1 >5
E
H
p1
c2<10
E
H
p2
c3 <=100
E
H
p3

ÖRNEK UYGULAMALAR #1

- 1-2-3-4-5-6-7-8-9
- 1-2-3-10-12-9
- 1-2-3-4-11-7-8-9
- 1-2-3-4-11-12-9

ÖRNEK UYGULAMALAR #1

- 1-2-3-4-5-6-7-8-9
- **6, 8,100**
- 1-2-3-10-12-9
- **4, 8,120**

ÖRNEK UYGULAMALAR #1

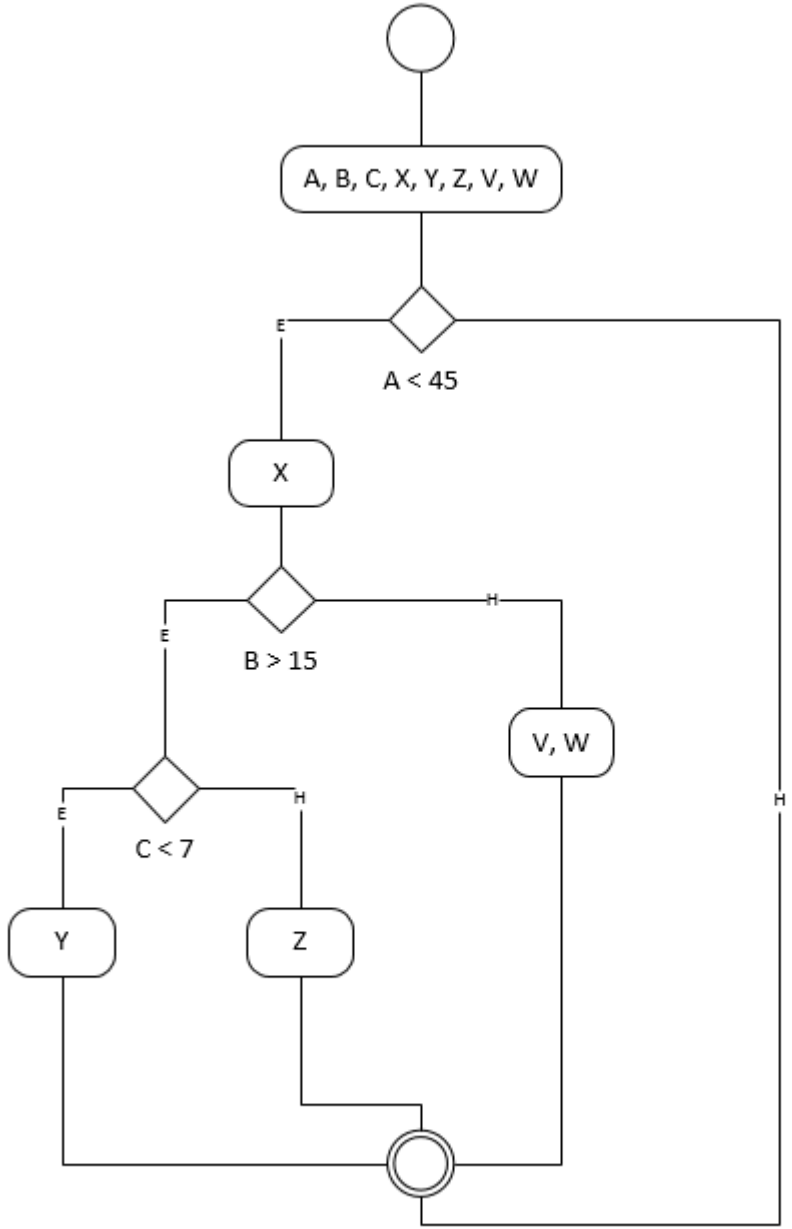
- 1-2-3-4-11-7-8-9
- **6, 12, 89**
- 1-2-3-4-11-12-9
- **6,12, 120**

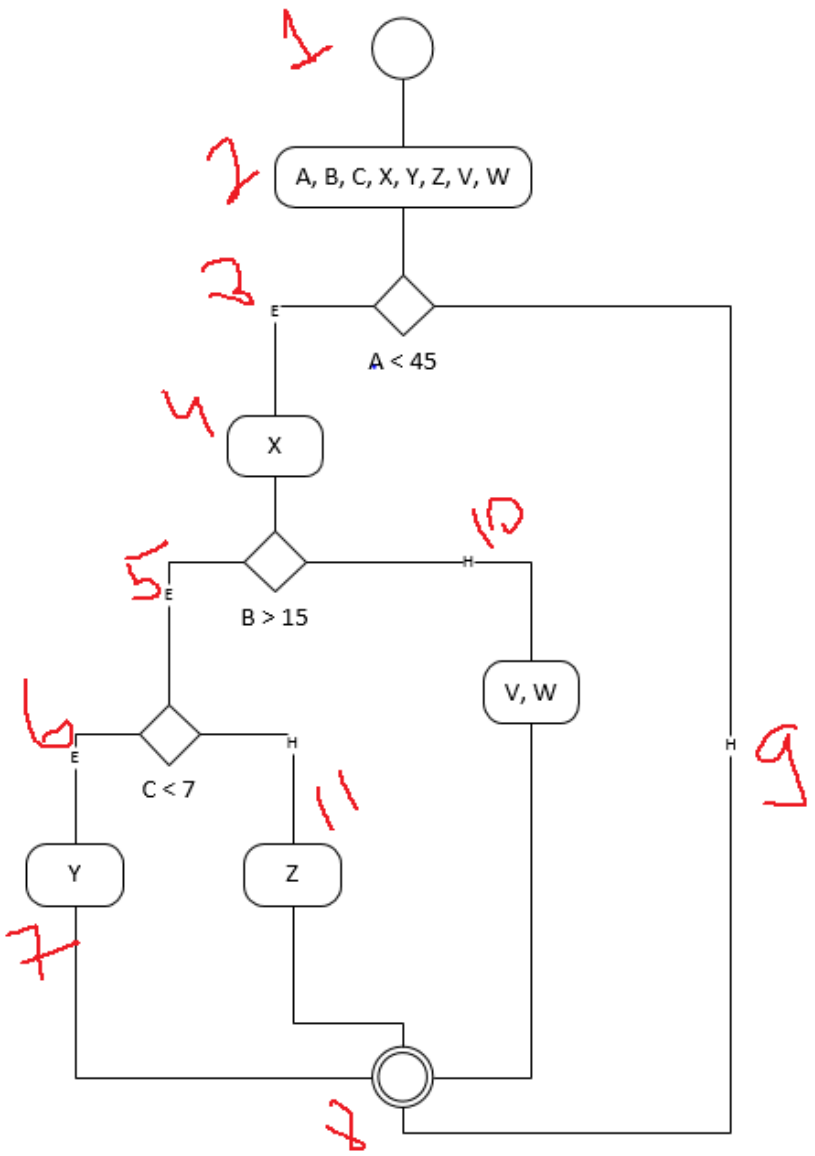
ÖRNEK UYGULAMALAR #2

```
0 başvuru
static void Main(string[] args)
{
    Console.WriteLine("Lütfen üç adet tam sayı değeri giriniz: ");
    string a, b, c;
    a = Console.ReadLine();
    b = Console.ReadLine();
    c = Console.ReadLine();
    int A = Convert.ToInt32(a);
    int B = Convert.ToInt32(b);
    int C = Convert.ToInt32(c);
    int x = 0;
    int Y = 0;
    int Z = 1;
    int V = 2;
    int W = 3;
```

```
void fonksiyonF1()
{
    while (A < 45)
    {
        x = A + x;

        if (B > 15)
        {
            if (C < 7)
                Y = Y + C;
            else
                Z = Z * 2;
        }
        else
        {
            V = V + A + B;
            W = W - Z;
        }
    }
}
```


A, B, C, X, Y, Z, V, W
E
A < 45
H
X
E
B > 15
H
V, W
E
C < 7
H
Y
Z


1
2
A, B, C, X, Y, Z, V, W
3
E
A < 45
H
9
4
X
5
E
B > 15
H
10
V, W
6
E
C < 7
H
11
Y
Z
7
8

ÖRNEK UYGULAMALAR #2

- 1-2-3-4-5-6-7-8
- 1,2,9,8
- 1-2-3-4-10-8
- 1-2-3-4-5-11-8

ÖRNEK UYGULAMALAR #2

- 1-2-3-4-5-6-7-8
- **44, 18, 6**
- 1,2,9,8
- **46, 3, 4**

ÖRNEK UYGULAMALAR #2

- 1-2-3-4-10-8
- **43, 12, 7**
- 1-2-3-4-5-11-8
- **35, 21, 8**

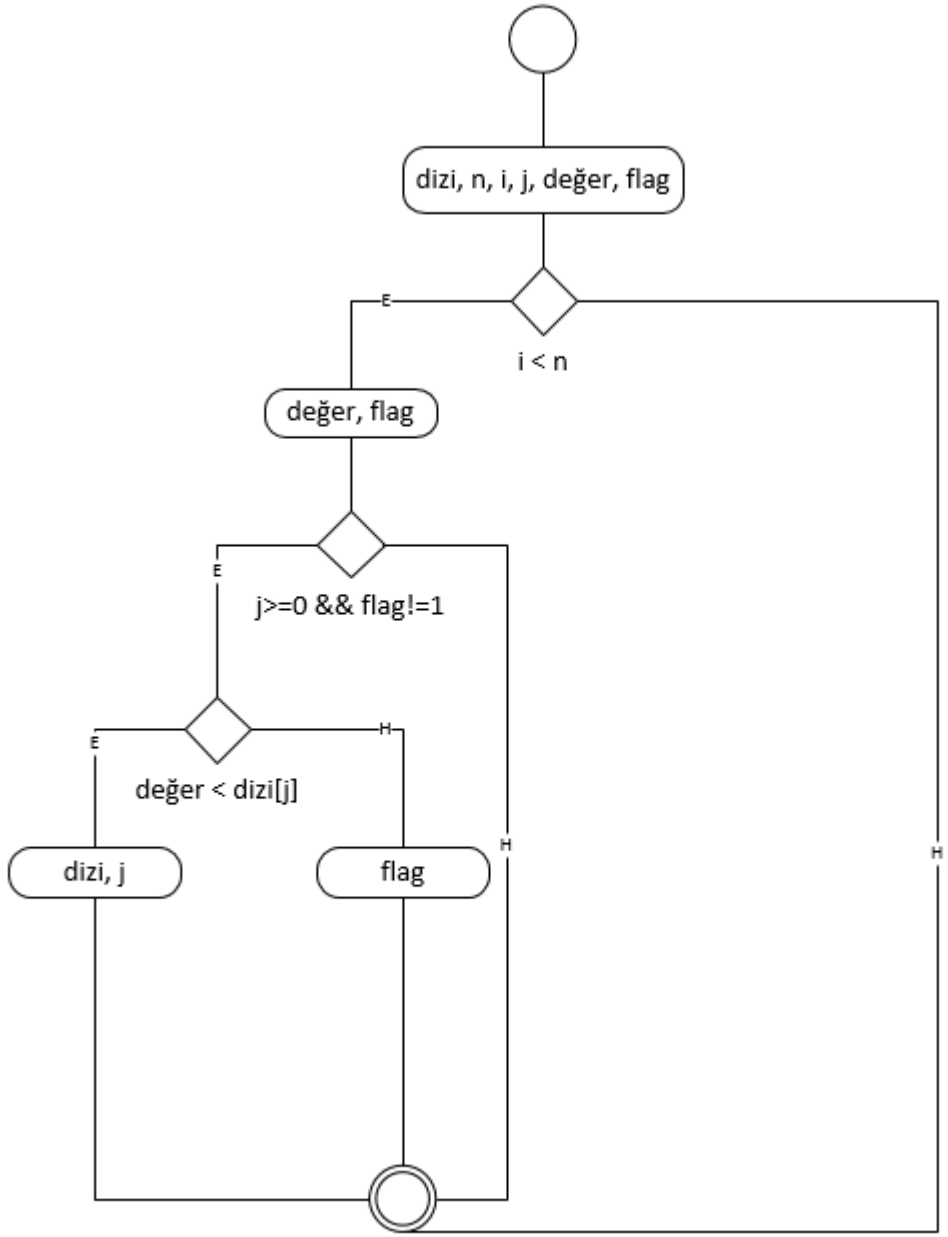
ÖRNEK UYGULAMALAR #3

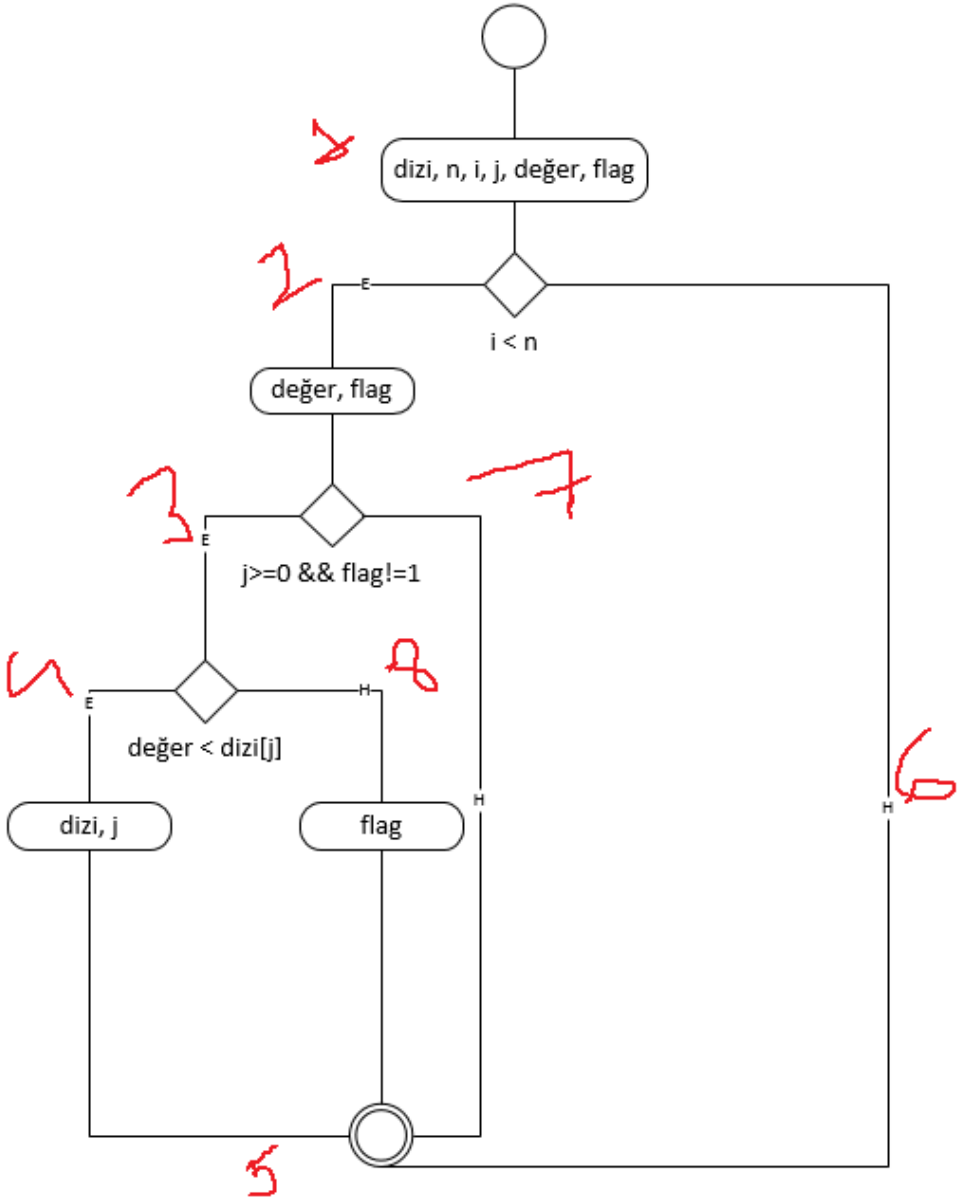
```
static void Main(string[] args)
{
    int[] dizi = new int[10] { 23, 9, 85, 12, 99, 34, 60, 15, 100, 1 };
    int n = 10, i, j, değer, flag;

    Console.Write("\nSırasız değerler: ");
    for (i = 0; i < n; i++)
    {
        Console.Write(dizi[i] + " ");
    }

    for (i = 1; i < n; i++)
    {
        değer = dizi[i];
        flag = 0;
        for (j = i - 1; j >= 0 && flag != 1;)
        {
            if (değer < dizi[j])
            {
                dizi[j + 1] = dizi[j];
                j--;
                dizi[j + 1] = değer;
            }
            else flag = 1;
        }
    }
```

```
Console.Write("\nSıralı değerler: ");
for (i = 0; i < n; i++)
{
    Console.Write(dizi[i] + " ");
}
Console.Write("\n");
```


dizi, n, i, j, değer, flag
E
i < n
değer, flag
E
j>=0 && flag!=1
E
H
değer < dizi[j]
dizi, j
flag
H
H


dizi, n, i, j, değer, flag
E
i < n
değer, flag
E
j>=0 && flag!=1
E
H
değer < dizi[j]
dizi, j
flag
H
H

ÖRNEK UYGULAMALAR #3

- 1-2-3-4-5
- 1-2-3-8-5
- 1-2-7-5
- 1-2-6

UYGULAMA SORULARI

- Quick sort algoritmasının kodunu yazıp **(10p)** üzerinde beyaz kutu testi uygulayınız **(20p).**
- Girilen sayıyı ikili kod (binary) haline çeviren programı yazıp **(10p)** üzerinde beyaz kutu testi uygulayınız **(20p).**

UYGULAMA SORULARI

- Şekilde verilen kod için beyaz kutu testi uygulayınız **(20p).**

```
PROGRAM  maxsum ( maxint, value : INT )
                    INT    result := 0 ;   i := 0 ;
                    IF   value < 0
                    THEN   value  :=  - value ;
                    WHILE   ( i < value )  AND  ( result <= maxint )
                    DO            i  :=  i + 1 ;
                                  result  :=  result + i ;
                    OD;
                    IF   result <= maxint
                    THEN   OUTPUT ( result )
                    ELSE   OUTPUT ( “too large” )
          END.
```

UYGULAMA SORULARI

- Şekilde verilen kod için

beyaz kutu testi uygulayınız **(20p).**

```
integer a,b,count=0;
input a,b;
if (a == 0)
    while (b > 0)
        {b = b-1;
         count++;}
else if (a > 0)
    while (b < 0)
        {b = b+1;
         count--;}
else
    a = b;
output a,b,count;
```