

1.1 Program

Kullanıcının kendi labirentini oluşturması ve bu labirentlerden istediğini seçmesi ile seçilen labirentte farenin engellere takılmadan peyniri bulduğu masaüstü uygulaması.

1.2 Girdiler

Labirent Boyutu: Kullanıcının 6x6, 10x8 veya 12x10 boyutunda labirent oluşturma seçenekleri

Renkli Tuğlalar: Labirentte farenin gidiş yoluna konan engeller

Fare: Peyniri bulması gereken nesne

Peynir: Hedef konum

1.3 Mantığı

Kullanıcı ilk önce labirent oluşturmalıdır. Labirent oluşturmak için labirent oluşturma butonuna tıklar ve açılan pencerede önce koyacağı nesneyi daha sonra labirentteki konumunu seçerek. Labirentte nesneleri yükler. Daha sonra oluşturulan labirenti kaydederek ana pencereye döner. Ana pencerede kayıtlı labirentlerden birisini seçip yüklemesi suretiyle uygulama başlama konumuna gelir. Fare engellerin içerisinden geçmeden, engel gördüğünde yönünü değiştirerek ve yol tekrarından kaçınarak peyniri bulmalıdır.

İlk etapta fare mevcut konumuna göre ön, arka, sağ ve sol konumlarını kontrol eder. İlgili yolların boş olma durumuna göre önce ileri sonra sol sonra geri sonra sağ metodlarını kullanarak yönünü bulmaya çalışır. Fare yol tekrarı yapmamak için yol tekrarında yol önceliğini değiştirerek farklı alternatifler oluşturur.

1.4 Program Kodu

```
using System;
using System.Collections.Generic;
using System.Drawing;
using System.Windows.Forms;

namespace projeHepsiForm
{
    public partial class frmAnimasyon : Form
    {
        public frmAnimasyon() { InitializeComponent(); }

        public void btnKayitSil(Button tiklananButon)
        {
            secilmisLab.doluListem[int.Parse(tiklananButon.Name)] = -1;
            tiklananButon.BackgroundImage = null;
        }

        public void btnKayitGoster()
        {

```

```

int tttt= 2;//Borderstyle fixed yapınca 2 piksel kendine ayırıyor
fPanel1.Width = secilmisLab.enBoyAdet.X * secilmisLab.genislik + tttt;
fPanel1.Height = secilmisLab.enBoyAdet.Y * secilmisLab.yukseklik + tttt;

fPanel1.Controls.Clear();
for (int i = 0; i < secilmisLab.doluPictureListem.Length; i++)
{
    //Varsayılan Değerlerde Buton Oluşturma
    Button btn = new Button { Width = secilmisLab.genislik, Height = secilmisLab.yukseklik,
Margin = new Padding(0), BackgroundImageLayout = ImageLayout.Stretch, Name = i.ToString() };
    fPanel1.Controls.Add(btn);
    if (secilmisLab.doluListem[i]>0)
    {
        if (secilmisLab.doluListem[i] == 4) { btn.BackgroundImage =
btnFare.BackgroundImage; }
        else if(secilmisLab.doluListem[i] == 5) { btn.BackgroundImage =
btnPeynir.BackgroundImage; }
        else if(secilmisLab.doluListem[i] == 1) { btn.BackgroundImage =
btnSari.BackgroundImage; }
        else if (secilmisLab.doluListem[i] == 2) { btn.BackgroundImage =
btnMavi.BackgroundImage; }
        else if (secilmisLab.doluListem[i] == 3) { btn.BackgroundImage =
btnKirmizi.BackgroundImage; }
        else {
            MessageBox.Show("O ne ki?");
        }
    }
}

}
public void labirentlisteTazele()
{
    listLabirentYukle.Items.Clear();
    foreach (classLabo item in lablar)
    {
        listLabirentYukle.Items.Add(item.ad);
    }
}
List<classLabo> lablar;

classLabo secilmisLab;
private void frmAnimasyon_Shown(object sender, EventArgs e)
{
    btnSari.BackgroundImage = Image.FromFile(Application.StartupPath + @"\Resimler\" +
"DugarSari" + ".png");
    btnMavi.BackgroundImage = Image.FromFile(Application.StartupPath + @"\Resimler\" +
"DugarMavi" + ".png");
}

```

```
btnKirmizi.BackgroundImage = Image.FromFile(Application.StartupPath + @"\Resimler\" +
"DugarKirmizi" + ".png");
```

```
btnFare.BackgroundImage = Image.FromFile(Application.StartupPath + @"\Resimler\" +
"mice_tailnot" + ".png");
```

```
btnPeynir.BackgroundImage = Image.FromFile(Application.StartupPath + @"\Resimler\" +
"mice_cheese" + ".png");
```

```
lablar = new List<classLabo>();
```

```
int testa = 12;
```

```
int testb = 10;
```

```
testLab = new classLabo() { ad = "TestLab1", enBoyAdet = new Point(testa, testb), genislik
= 70, yukseklik = 70, doluListem = new int[testa * testb], doluPictureListem = new Button[testa *
testb] };
```

```
int tempindis = testa * testb - 1;
```

```
testLab.doluListem[tempindis] = 4;
```

```
testLab.doluPictureListem[tempindis] = new Button() { Width = testLab.genislik, Height =
testLab.yukseklik, Margin = new Padding(0), BackgroundImageLayout = ImageLayout.Stretch,
Name = tempindis.ToString() };
```

```
tempindis = testa * testb - 2;
```

```
testLab.doluListem[tempindis] = 2;
```

```
testLab.doluPictureListem[tempindis] = new Button() { Width = testLab.genislik, Height =
testLab.yukseklik, Margin = new Padding(0), BackgroundImageLayout = ImageLayout.Stretch,
Name = tempindis.ToString() };
```

```
tempindis = 25;
```

```
testLab.doluListem[tempindis] = 5;
```

```
testLab.doluPictureListem[tempindis] = new Button() { Width = testLab.genislik, Height =
testLab.yukseklik, Margin = new Padding(0), BackgroundImageLayout = ImageLayout.Stretch,
Name = tempindis.ToString() };
```

```
tempindis = 13;
```

```
testLab.doluListem[tempindis] = 3;
```

```
testLab.doluPictureListem[tempindis] = new Button() { Width = testLab.genislik, Height =
testLab.yukseklik, Margin = new Padding(0), BackgroundImageLayout = ImageLayout.Stretch,
Name = tempindis.ToString() };
```

```
lablar.Add(testLab);
```

```
labirentlisteTazele();
```

```
listLabirentYukle.SelectedIndex = 0;
```

```
btnYukle_Click(null, null);
```

```

}
#region LAB OLUŞTUR FORMU ÇAĞIR
classLabo kapananLab;
frmOlustur forumOlustur;
classLabo testLab;
private void btnLabirent_Click(object sender, EventArgs e)
{
    forumOlustur = new frmOlustur();
    forumOlustur.FormClosing += LabOlustur_FormClosing;
    forumOlustur.Show();
    this.Hide();
}

private void LabOlustur_FormClosing(object sender, FormClosingEventArgs e)
{
    if (forumOlustur.alBeni)
    {
        kapananLab = forumOlustur.lab1;
        lablar.Add(kapananLab);
        labirentlisteTazele();
        MessageBox.Show("Labirentiniz Kaydedildi");
    }
    else
    {
        kapananLab = null;
        MessageBox.Show("Labirentiniz Kaydedilmedi");
    }

    this.Show();
}
#endregion

private void btnYukle_Click(object sender, EventArgs e)
{
    int seciliLabIndeks = listLabirentYukle.SelectedIndex;
    if (seciliLabIndeks != -1)
    {
        secilmisLab = lablar[seciliLabIndeks];
        btnKayitGoster();
    }
    else
    {
        MessageBox.Show("Önce Labirent Seciniz");
    }
}

private void btnIptal_Click(object sender, EventArgs e)

```

```

    {
        listLabirentYukle.SelectedItem = null;
    }

    private void btnKapat_Click(object sender, EventArgs e)
    {
        this.Close();
    }

    private void btnBasla_Click(object sender, EventArgs e)
    {
        classHareket hareket = new classHareket(secilmisLab);
        classYonler yon = new classYonler(secilmisLab);
        yon.yonleriBul();
    }
}

```

```

using System;
using System.Drawing;
using System.Linq;
using System.Windows.Forms;

namespace projeHepsiForm
{
    public partial class frmOlustur : Form
    {
        public frmOlustur() { InitializeComponent(); }

        int seciliRakam;
        private void buttonlar_Click(object sender, EventArgs e)
        {
            seciliRakam = -1;

            string strSecimAd = (sender as Button).Name;
            if (strSecimAd == "btnSari")
            {
                rSari.Checked = true;
                rMavi.Checked = false;
                rKirmizi.Checked = false;
                rFare.Checked = false;
                rPeynir.Checked = false;
                seciliRakam = 1;
            }
            else if (strSecimAd == "btnMavi")

```

```

{
    rSari.Checked = false;
    rMavi.Checked = true;
    rKirmizi.Checked = false;
    rFare.Checked = false;
    rPeynir.Checked = false;
    seciliRakam = 2;
}
else if(strSecimAd == "btnKirmizi")
{
    rSari.Checked = false;
    rMavi.Checked = false;
    rKirmizi.Checked = true;
    rFare.Checked = false;
    rPeynir.Checked = false;
    seciliRakam = 3;
}
else if(strSecimAd == "btnFare")
{
    rSari.Checked = false;
    rMavi.Checked = false;
    rKirmizi.Checked = false;
    rFare.Checked = true;
    rPeynir.Checked = false;
    seciliRakam = 4;
}
else if(strSecimAd == "btnPeynir")
{
    rSari.Checked = false;
    rMavi.Checked = false;
    rKirmizi.Checked = false;
    rFare.Checked = false;
    rPeynir.Checked = true;
    seciliRakam = 5;
}
else
{
    MessageBox.Show("Nasil?");
}
btnGoster.BackgroundImage = (sender as Button).BackgroundImage;
}

```

```

private void radioButtonlar_Click(object sender, EventArgs e)
{

```

```

    rSari.Checked = false;
    rMavi.Checked = false;
    rKirmizi.Checked = false;
    rFare.Checked = false;

```

```

        rPeynir.Checked = false;

        MessageBox.Show("Lütfen Buton Üzerine Tıklayınız");
    }

    public class Labo lab1; public bool alBeni;
    private void LabirentClick(object sender, EventArgs e)
    {

        string strSecimAd = (sender as Button).Name;
        if (strSecimAd == "btnLab1")
        {
            lab1 = new class Labo() { enBoyAdet = new Point(8, 6), genislik = 70, yukseklik = 70,
            doluListem = new int[8 * 6], doluPictureListem = new Button[8 * 6] };
        }
        else if (strSecimAd == "btnLab2")
        {
            lab1 = new class Labo() { enBoyAdet = new Point(10, 8), genislik = 70, yukseklik = 70,
            doluListem = new int[10 * 8], doluPictureListem = new Button[10 * 8] };
        }
        else
        {
            lab1 = new class Labo() { enBoyAdet = new Point(12, 10), genislik = 70, yukseklik = 70,
            doluListem = new int[12 * 10], doluPictureListem = new Button[12 * 10] };
        }

        flowLayoutPanel1.Width = lab1.enBoyAdet.X * lab1.genislik;
        flowLayoutPanel1.Height = lab1.enBoyAdet.Y * lab1.yukseklik;

        flowLayoutPanel1.Controls.Clear();
        for (int i = 0; i < lab1.enBoyAdet.X * lab1.enBoyAdet.Y; i++)
        {
            //Varsayılan Değerlerde Buton Oluşturma
            Button btn = new Button { Width = lab1.genislik, Height = lab1.yukseklik, Margin = new
            Padding(0), BackgroundImageLayout = ImageLayout.Stretch, Name = i.ToString() };
            flowLayoutPanel1.Controls.Add(btn);
            lab1.doluListem[i] = -1;
            lab1.doluPictureListem[i] = btn;

            btn.MouseDown += Btn_MouseDown;
        }
    }

    public void btnKayitEkle(Button tiklananButon)
    {

        int indeks = int.Parse(tiklananButon.Name);
        if (indeks == -1) throw new Exception("seçilmemiş");
        else
        {

```

```

        if (seciliRakam == 4)
        {
            if (lab1.doluListem.Contains(4))
            {
                for (int i = 0; i < lab1.doluPictureListem.Length; i++)
                {
                    if (lab1.doluListem[i] == 4)
                    {
                        lab1.doluPictureListem[i].BackgroundImage = null;
                        lab1.doluListem[i] = -1;
                    }
                }
            }
        }
        else if (seciliRakam == 5)
        {
            if (lab1.doluListem.Contains(5))
            {
                for (int i = 0; i < lab1.doluPictureListem.Length; i++)
                {
                    if (lab1.doluListem[i] == 5)
                    {
                        lab1.doluPictureListem[i].BackgroundImage = null;
                        lab1.doluListem[i] = -1;
                    }
                }
            }
        }

        lab1.doluListem[indeks] = seciliRakam;
        tiklananButon.BackgroundImage = btnGoster.BackgroundImage;

    }
}

public void btnKayitSil(Button tiklananButon)
{
    lab1.doluListem[int.Parse(tiklananButon.Name)] = -1;
    tiklananButon.BackgroundImage = null;
}

private void Btn_MouseDown(object sender, MouseEventArgs e)
{
    Button sec = sender as Button;
    int secSayisi = int.Parse(sec.Name);

    if (e.Button == MouseButtons.Left)
    {

```



```

        btnKayitEkle(sec);
    }
    else if (e.Button == MouseButtons.Right)
    {
        btnKayitSil(sec);
    }
}

private void Form1_Shown(object sender, EventArgs e)
{
    alBeni = false;
    btnSari.BackgroundImage = Image.FromFile(Application.StartupPath + @"\" + Resimler\" +
    "DuvarSari" + ".png");
    btnMavi.BackgroundImage = Image.FromFile(Application.StartupPath + @"\" + Resimler\" +
    "DuvarMavi" + ".png");
    btnKirmizi.BackgroundImage = Image.FromFile(Application.StartupPath + @"\" + Resimler\" +
    "DuvarKirmizi" + ".png");

    btnFare.BackgroundImage = Image.FromFile(Application.StartupPath + @"\" + Resimler\" +
    "mice_tailnot" + ".png");
    btnPeynir.BackgroundImage = Image.FromFile(Application.StartupPath + @"\" + Resimler\" +
    "mice_cheese" + ".png");

    btnKaydet.BackgroundImage = Image.FromFile(Application.StartupPath + @"\" + Resimler\" +
    "iconkaydet" + ".png");
    btnIptal.BackgroundImage = Image.FromFile(Application.StartupPath + @"\" + Resimler\" +
    "iconsupur" + ".png");
    LabirentClick(btnLab1, null);
    buttonlar_Click(btnMavi, null);
}
string labAd = "";
private void kaydet_Click(object sender, EventArgs e)
{
    labAd = "";
    bool fare = false;
    bool peynir = false;

    for (int i = 0; i < lab1.doluPictureListem.Length; i++)
    {
        if (lab1.doluListem[i] == 4){ fare = true; }
        else if (lab1.doluListem[i] == 5){ peynir = true; }
    }
    labAd = txtLabAd.Text;
    if (labAd == "")
    {
        MessageBox.Show("Labirent Adını Giriniz");
        alBeni = fare;
    }
    else
    {

```

```

        if (fare && peynir)
        {
            lab1.ad = labAd;
            alBeni = true; Close();
        }

        else
        {
            MessageBox.Show("Labirente fare ve peynir Ekleyiniz");
            alBeni = false;
        }
    }
}

private void temizle_Click(object sender, EventArgs e)
{
    for (int i = 0; i < lab1.doluPictureListem.Length; i++)
    {
        lab1.doluPictureListem[i].BackgroundImage = null;
        lab1.doluListem[i] = -1;
    }
    txtLabAd.Text = "";
    alBeni = false;
}
}
}

```

```

using System.Drawing;
using System.Windows.Forms;

```

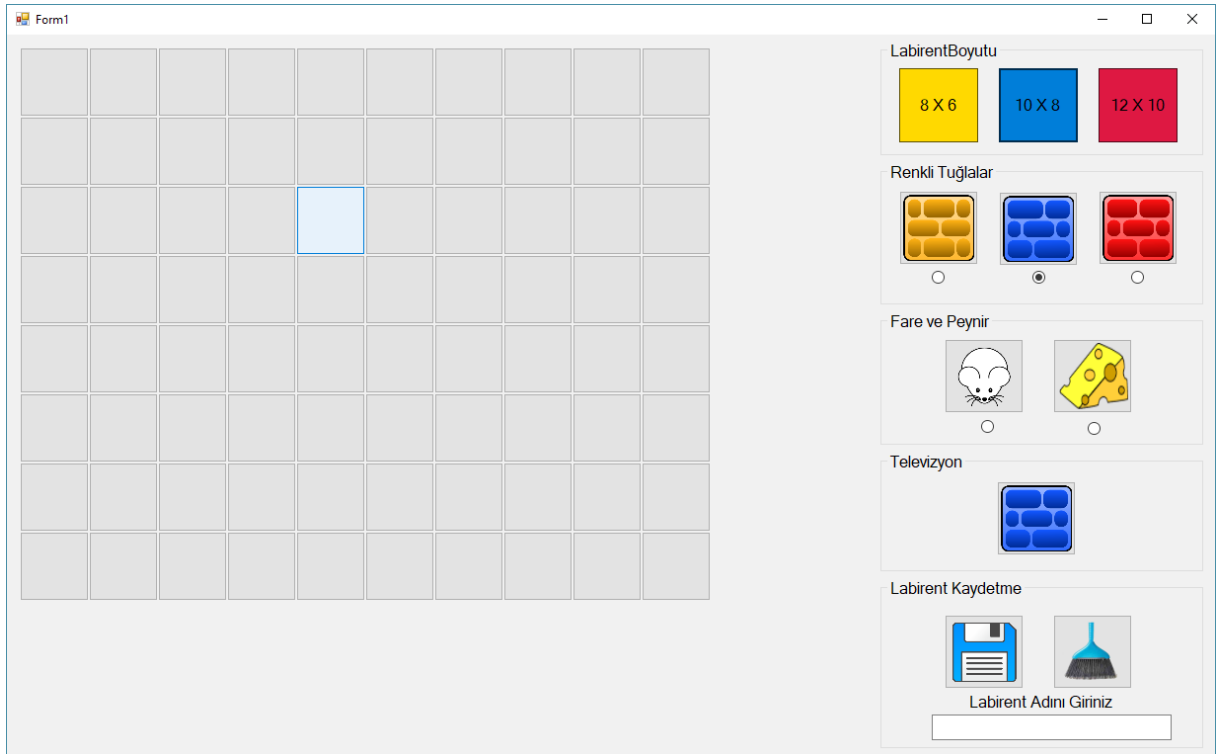
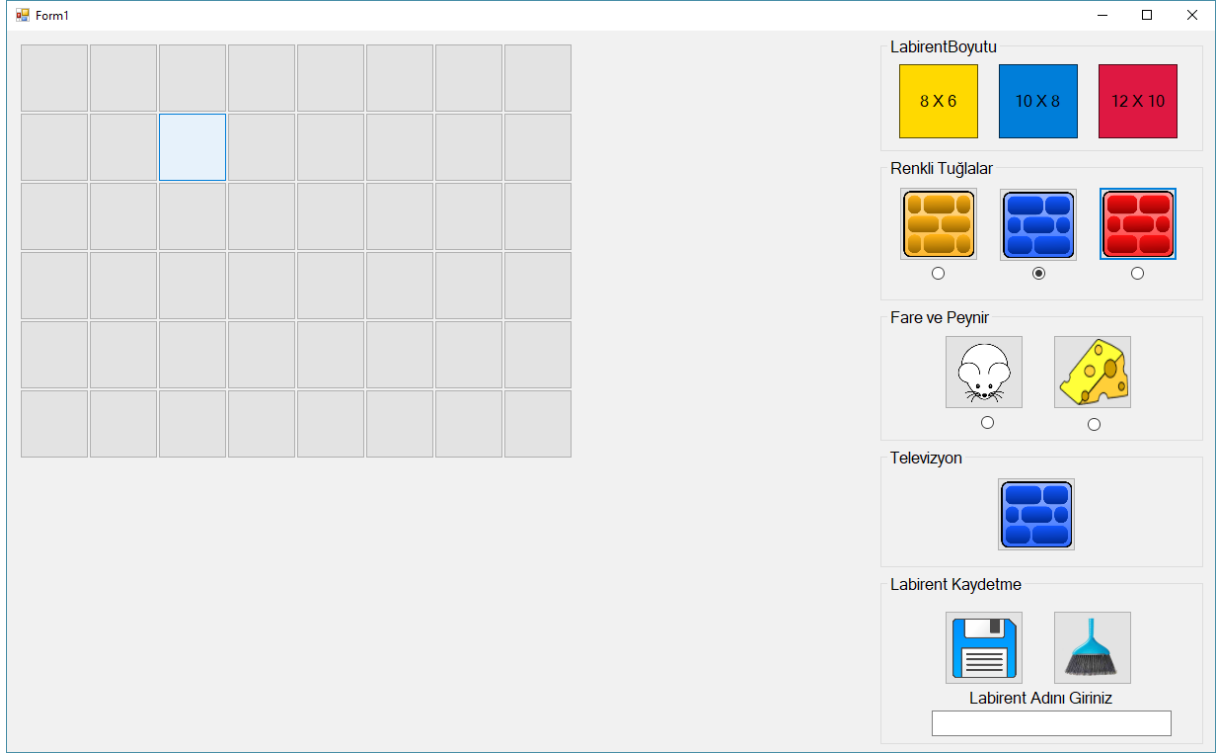
```

namespace projeHepsiForm
{
    public class classLabo
    {
        public string ad;
        public Point enBoyAdet;
        public int[] doluListem;
        public Button[] doluPictureListem;
        public int genislik;
        public int yukseklik;
        public classLabo()
        {
            ad = "";
            enBoyAdet = new Point() {X = -1, Y=-1 };
        }
    }
}

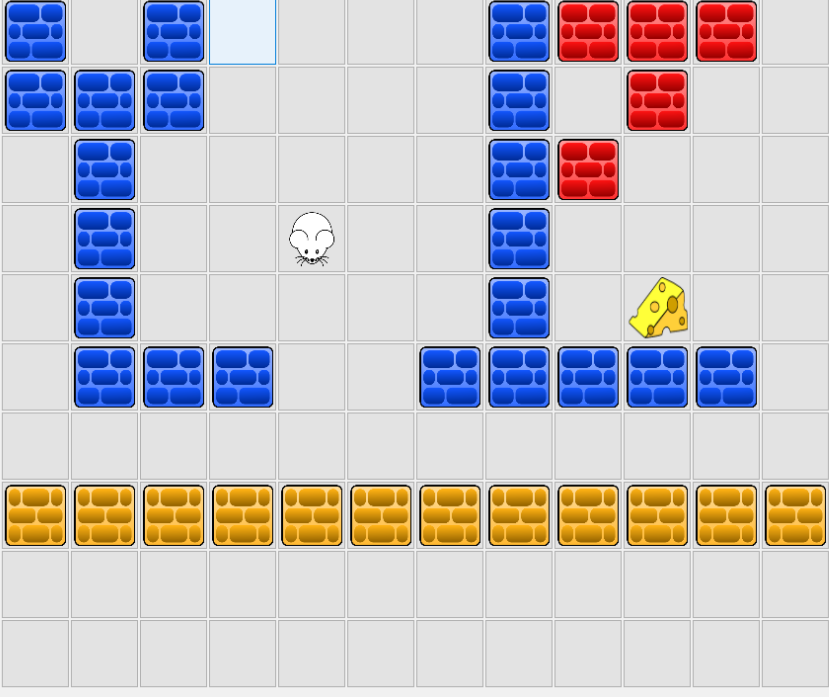
```

}

1.5 Ekran Görüntüleri



Form1



LabirentBoyutu

8 X 6 10 X 8 12 X 10

Renkli Tuğlalar

☒ ☐ ☐

Fare ve Peynir

☐ ☐

Televizyon

☐

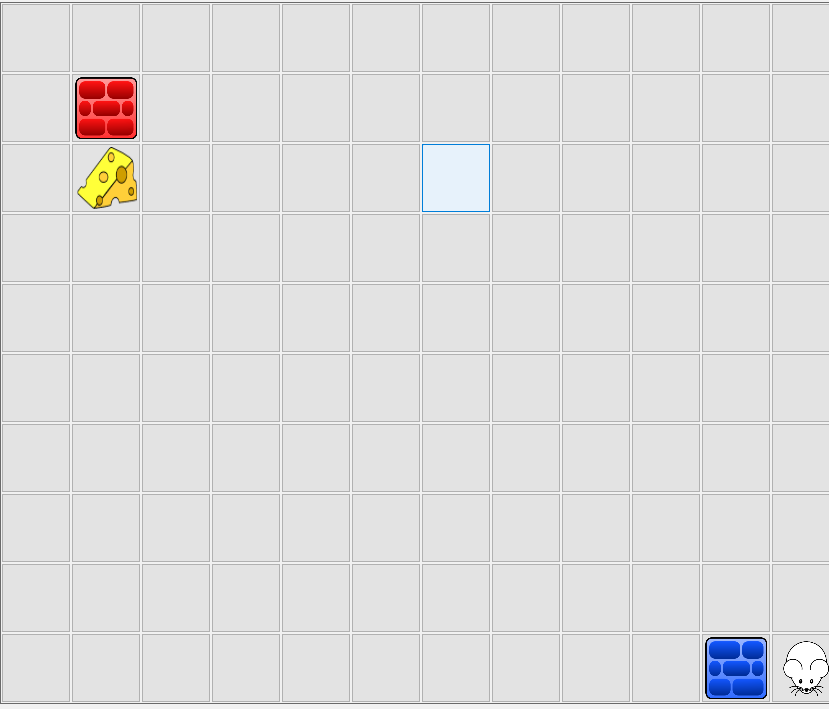
Labirent Kaydetme

☐ ☐

Labirent Adını Giriniz

Labirent2

LABİRENT OLUŞTUR



Labirent Yükle

TestLab1

Yükle İptal

Animasyon

Lab Oluştur Başla Dur

İçindekiler

☒ ☐ ☐

☐ ☐

×

Labirentiniz Kaydedilmedi

Tamam

×

Labirentiniz Kaydedildi

Tamam

1.6 Test Case

1.6.1

Level	Orta
Purpose	Kullanıcının belirlediği labirent boyutunda labirent oluşması
Inputs	8x6 10x8 12x10 varsayılan ekran değerleri
Excepted Output	Seçilen boyutta labirent oluşturulması
Passed Critical	Seçilen butondaki boyutlarda haritanın oluşturulması
Failed Critical	Daha büyük veya daha küçük bir haritanın oluşturulması,seçim yapılmasına rağmen haritanın oluşturulmaması
Test Procedure	Test kullanıcısı,yazılımı,desteklenen sistem ve cihazlarda çalıştırarak belirtilen girdilerle testi gerçekleştirmelidir. Test işlemi tamamlandıktan sonra test sonucu pass/fail olarak belirtilerek nedenleriyle birlikte raporlamalıdır.

1.6.2

Level	Yüksek
Purpose	Labirent oluşturma ekranında fare girişi
Inputs	Kullanıcının oluşturulan labirentte seçtiği konuma fare koyması
Excepted Output	Fare konulmaz ise labirent kaydının başarısız olması ve kullanıcıya hata mesajı iletilmesi
Passed Critical	Fare eklenmeden kaydet butonuna tıklandığında kayıt işleminin gerçekleştirilmemesi
Failed Critical	Labirentin kaydedilip seçime hazır hale gelmesi
Test Procedure	Test kullanıcısı,yazılımı,desteklenen sistem ve cihazlarda çalıştırarak belirtilen girdilerle testi gerçekleştirmelidir. Test işlemi tamamlandıktan sonra test sonucu pass/fail olarak belirtilerek nedenleriyle birlikte raporlamalıdır.

1.6.3

Level	Yüksek
Purpose	Labirent oluşturma ekranında fare girişi
Inputs	Kullanıcının oluşturulan labirentte seçtiği konumlara fare koyması
Excepted Output	Kullanıcının birden fazla fare koymaya çalışması durumunda farenin konumunun değiştirilmesi
Passed Critical	Kullanıcının ikinci fare konumu seçiminde eski farenin silinip yeni konumda fare oluşturulması
Failed Critical	Tek labirent içerisinde birden fazla fare konulması
Test Procedure	Test kullanıcısı,yazılımı,desteklenen sistem ve cihazlarda çalıştırarak belirtilen girdilerle testi gerçekleştirmelidir. Test işlemi tamamlandıktan sonra test sonucu pass/fail olarak belirtilerek nedenleriyle birlikte raporlamalıdır.

1.6.4

Level	Yüksek
Purpose	Labirent oluşturma ekranında peynir girişi
Inputs	Kullanıcının oluşturulan labirentte seçtiği konuma peynir koyması
Excepted Output	Kullanıcının peynir koymaması durumunda labirentin

	kaydedilmemesi ve kullanıcıya hata mesajı döndürmesi
Passed Critical	Kullanıcının peynir koymaması durumunda labirent kaydının gerçekleştirilememesi
Failed Critical	Peynir konulmadan labirent kaydının gerçekleştirilmesi
Test Procedure	Test kullanıcısı,yazılımı,desteklenen sistem ve cihazlarda çalıştırarak belirtilen girdilerle testi gerçekleştirmelidir. Test işlemi tamamlandıktan sonra test sonucu pass/fail olarak belirtilerek nedenleriyle birlikte raporlamalıdır.

1.6.5

Level	Yüksek
Purpose	Labirent oluşturma ekranında peynir girişi
Inputs	Kullanıcının oluşturulan labirentte seçtiği konumlara peynir koyması
Excepted Output	Kullanıcının birden fazla peynir koymaya çalışması durumunda farenin konumunun değiştirilmesi
Passed Critical	Kullanıcının ikinci peynirkonumu seçiminde peynirin farenin silinip yeni konumda fare oluşturulması
Failed Critical	Tek labirent içerisinde birden fazla peynir konulması
Test Procedure	Test kullanıcısı,yazılımı,desteklenen sistem ve cihazlarda çalıştırarak belirtilen girdilerle testi gerçekleştirmelidir. Test işlemi tamamlandıktan sonra test sonucu pass/fail olarak belirtilerek nedenleriyle birlikte raporlamalıdır.

1.6.6

Level	Yüksek
Purpose	Labirentin temizlenmesi
Inputs	Kullanıcının labirenti temizle butonuna tıklaması
Excepted Output	Kullanıcının oluşturacağı labirentin temizlenmesi
Passed Critical	Hiçbir şey kalmadan oluşturulacak labirentin temizlenmesi
Failed Critical	Kullanıcının labirenti temizle seçeneğini seçmesi halinde labirentte hala mevcut eleman bulunması
Test Procedure	Test kullanıcısı,yazılımı,desteklenen sistem ve cihazlarda çalıştırarak belirtilen girdilerle testi gerçekleştirmelidir. Test işlemi tamamlandıktan sonra test sonucu pass/fail olarak belirtilerek nedenleriyle birlikte raporlamalıdır.

1.6.7

Level	Yüksek
Purpose	Farenin hareketlerinin döngüye girmesi
Inputs	Labirent
Excepted Output	Farenin hareket değişimi yaparak döngüyü kırması
Passed Critical	Farenin hareket değiştirmesi
Failed Critical	Farenin sürekli aynı hareketleri tekrarlayıp hiçbir zaman peyniri bulamaması
Test Procedure	Test kullanıcısı,yazılımı,desteklenen sistem ve cihazlarda çalıştırarak belirtilen girdilerle testi gerçekleştirmelidir. Test işlemi tamamlandıktan sonra test sonucu pass/fail olarak belirtilerek nedenleriyle birlikte raporlamalıdır.

1.6.8

Level	Yüksek
-------	--------

Purpose	Farenin peyniri bulamamadı
Inputs	Labirent
Excepted Output	Kullanıcıya peynir bulunamadı mesajının döndürülmesi
Passed Critical	Farenin tüm yolların denendiğini ve peynir için yol kalmadığını anlaması.
Failed Critical	Programın sürekli çalışması
Test Procedure	Test kullanıcısı,yazılımı,desteklenen sistem ve cihazlarda çalıştırarak belirtilen girdilerle testi gerçekleştirmelidir. Test işlemi tamamlandıktan sonra test sonucu pass/fail olarak belirtilerek nedenleriyle birlikte raporlamalıdır.