

**BURSA TEKNİK ÜNİVERSİTESİ**

**MÜHENDİSLİK FAKÜLTESİ**

**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

**BİLGİSAYAR MİMARİSİ DERSİ**

**DÖNEM PROJESİ**

**ADI VE SOYADI** Burak AYTAŞ

**Öğr. No** 18360859030

**ADI VE SOYADI**  Nil DUMAN

**Öğr. No**  18360859026

# Mayıs 2020

**MARİE SİMÜLATÖRÜ**

Simülasyon içinde kaydettiğimiz C# kodları mevcuttur.

Text kısmına ilk önce program yüklenmelidir. Program yüklemeden çalıştır a basılırsa, ekranda Önce Program Yüklenmelidir yazan bir mesaj kutusu belirir. Bu özellik de aşağıdaki kod ile sağlanmıştır.

***if(dg\_Memory.Rows.Count == 0)***

***{***

***MessageBox.Show("Önce Program Yüklenmelidir");***

***return;***

***}***

Text kısmına program yüklendikten sonra bellek ve etiket kısımları oluşur. Etiket kısmında etiketin adresi ve o adresteki karşılığı gösterilir.

Program yüklendikten sonra ihtiyaç duyulursa random sayı oluşturulur. Program yüklenmeden bellek oluşamayacağı için, ilk önce programın yüklenip daha sonra random sayı oluşturulması gerekmektedir. Sayının belleğini hangi aralığında bulunması isteniliyorsa girilir ve sonra text kısmındaki assembly kodu çalıştırılır.

Random sayı üretme button kodu aşağıdaki şekildedir:

***private void btn\_CreateRandom\_Click(object sender, EventArgs e)***

***{***

***if(dg\_Memory.Rows.Count == 0)***

***{***

***MessageBox.Show("Önce Program Yüklenmelidir");***

***return;***

***}***

***var start = int.Parse(txt\_RndStart.Text, System.Globalization.NumberStyles.HexNumber);***

***var end = int.Parse(txt\_RndEnd.Text, System.Globalization.NumberStyles.HexNumber);***

***var rnd = new Random();***

***for (int i = start; i <= end; i++)***

***{***

***Memories.First(c => c.Address == i).Value = rnd.Next(0, 10).ToString("X");***

***}***

***showMemory();***

***}***

**ORG:** Programın kaç numaralı bellekten itibaren yerleşeceğini gösteren anahtar kelime

**END:** Programın en altına, kodun bittiğini gösteren yazı

**KARŞILAŞTIĞIMIZ SORUNLAR VE ÇÖZÜM YÖNTEMLERİ:**

1. Oluşturduğumuz simülatörü test etmek için yazdığımız programlarda, rastgele sayı üretmemiz gerekiyordu. Programların içine bu sayıları üretebilecek kodlar yazabilirdik fakat bu kısa vadeli bir çözüm olup, bir sonraki diğer bütün programlarda aynısını yapmamızı zorunlu kılardı. Bu yüzden simülatöre rastgele sayı üretebilecek bir buton eklemeye karar verdik. Bu buton dediğimiz kısımda; sayı oluşturulması istenilen aralıklar girilerek program çalıştırılmalıdır. Buton dediğimiz kısmın içinde ise 0-10 arası rastgele sayı atama fonksiyonu mevcuttur.

***{***

***Memories.First(c => c.Address == i).Value = rnd.Next(0, 10).ToString("X");***

***}***

1. Çarpma komutunun eksikliğini toplama işlemi ile kapattık.

**TEST PROGRAMLARI**

*NOT: Noktalı virgül (;) comment ifadelerini belirtir.*

1. **Bellekteki (210)16 - (22F)16 alanları arasındaki sayıların her birini 8 ile çarpıp üzerine yazdıran MARIE programı**

**ORG 0**

L1 Dec 0 ; *Döngü sonucunda L1 e dönebilmemiz için herhangi bir değere atamamız gerekiyordu. Aynı mantık L2 için de geçerli*

Load L2\_temp

Store L2\_sayac *; AC ye aktarma*

Load Toplam\_temp

Store Toplam

L2 Dec 0

Load Toplam

AddI Adres *;Adres etiketini gösterdiği değere eklendi*

Store Toplam *;Toplam değerini RAM’e gönderme*

Load L2\_sayac

Subt One *; 1 eksiltme*

Store L2\_sayac ;*sayaç Ram e gönderilsin*

Skipcond 400 *;Skipcond da döngünün kontrolü yapılır. 8 tekrardan sonra döngü kırılır*

Jump L2 *; L2 ye dönmesi için*

Load Toplam *;döngü bittiğinde hedeflenen çarpma işlemine ulaşılır ve AC’ye aktarılır*

StoreI Adres

Load Adres

Add One *; birer arttırarak bir sonraki bellek gözüne geçiliyor*

Store Adres

Load L1\_sayac

Subt One

Store L1\_sayac

Skipcond 400

Jump L1 *; L1 e dönmesi için (o değil ise)*

Halt *; 0 ise program amacına ulaşmış demektir ve sonlanır*

L2\_sayac Dec 8 ;*sayılar 8 kez toplanacak burada çarpma işlemi olmadığından toplama işlemi kullanıldı.*

L2\_temp Dec 8 *;sadece ilk sekiz sayının çarpılmasını önlemek için geçici bir değişken atanıyor*

L1\_sayac Dec 32 *;kaç sayının çarpıldığını gösteriyor*

Adres Hex 210

One Dec 1

Toplam Dec 0

Toplam\_temp Dec 0

**END**

NOT: Döngüler (Loop, L) 3 programda da aynı şekilde çalışır.

1. **Fibonacci serisinin ilk 10 elemanını (4FF)16 adresinden başlayan bellek gözlerine kaydeden MARIE programı**

ORG 4E5

Loop Dec 0

LoadI Adres1 *;Adres1 in işaret ettiği yer AC’ye aktarılıyor*

Store Sayi *;Sayi değeri AC’ ye aktarılıyor*

LoadI Adres2

Add Sayi

Store Temp

Load Adres1

Add One

Store Adres1

Load Adres2

Add One *; Adres2 kendinden sonraki göze aktarılıyor*

Store Adres2

Load Temp

StoreI Adres2

Load Sayac

Subt One *; sayaç değeri 1 azaltılır*

Store Sayac

Skipcond 400

Jump Loop

Halt *;program bitirme fonksiyonunu çalıştırıyoruz*

Sayi Dec 0

Adres1 Hex 4FF *;yapılacak işlem ve bulunduğu RAM gözü belirtiliyor*

Adres2 Hex 500

Temp Dec 0

One Dec 1

Sayac Dec 8

F1 Dec 1

F2 Dec 1

**END**

1. **Bellekteki (350)16 - (36F)16 alanları arasındaki sayıların kaçının pozitif, kaçının negatif ve kaçının sıfır olduğunu bularak istediğiniz ardışık 3 bellek gözüne yazan MARIE programı**

**ORG 0**

Loop Dec 0

LoadI Adres *;Adres’in işaret ettiği yer AC’ye aktarılıyor*

Skipcond 000 *;konrol için yazılmıştır. Sayı 0 ise alttaki işlemlere gerek yoktur.*

Jump Pozıtıfkontrol

Load Neg\_sayac

Add One *;neg\_sayac değerini 1 arttırıyoruz*

Store Neg\_sayac *;RAM’e aktardık*

Jump Atla *;Bir sonraki RAM gözüne gidiyoruz*

Pozıtıfkontrol Dec 0

Skipcond 400

Jump Pozitif *;sayinin değeri 0 değilse pozitif döngüsüne girilir*

Load Sıfır\_sayac

Add One

Store Sıfır\_sayac

Jump Atla

Pozitif Dec 0

Load Poz\_sayac

Add One

Store Poz\_sayac

Atla Dec 0

Load Adres

Add One

Store Adres

Load Sayac

Subt One

Store Sayac

Skipcond 400

Jump Loop

Halt *;program bitirme fonksiyonun çalıştırıyoruz*

Adres Hex 350 *;RAM de bulunduğumuz göz*

Sayac Dec 32 *; kaç RAM gözünün işleme ekleneceğini gösteriyor*

Sayac Dec 32

One Dec 1 *;1 arttırma ya da çıkarma işlemi yapar*

Neg\_sayac Dec 0

Poz\_sayac Dec 0

Sıfır\_sayac Dec 0

**END**