

```
(kali@kali)-[~/Desktop/micro/hw/MU0]
$ ls
CommandLine.png  mu0.py  test1.asm  test.asm
```

```
(kali@kali)-[~/Desktop/micro/hw/MU0]
$ python3 mu0.py test.asm
### Parsing source file ...
Recognized: INI 0x03A 0xA ; 1. sayi
Recognized: INI 0x03B 0xB ; 2. sayi
Recognized: INI 0x03C 0xC ; 3. sayi
Recognized: INI 0x03D 0xD ; 4. sayi
Recognized: INI 0x03E 0x0 ; toplam
Recognized: INI 0x03f 0xf ; en küçük sayi
Recognized: LDA 0x03A; ACC = 1.sayi
Recognized: ADD 0x03B; ACC =ACC + 2.sayi
Recognized: ADD 0x03C; ACC =ACC + 3.sayi
Recognized: ADD 0x03D; ACC =ACC + 4.sayi
Recognized: STO 0x03E; Toplami sakla
```

```
### Memory dump before program execution:
@0x03a: 0x00a (dec: 10)
@0x03b: 0x00b (dec: 11)
@0x03c: 0x00c (dec: 12)
@0x03d: 0x00d (dec: 13)
@0x03e: 0x000 (dec: 0)
@0x03f: 0x00f (dec: 15)
```

```
### Running the program ...
```

```
Reached end of instructions.
```

```
### Memory dump after program end:
@0x03a: 0x00a (dec: 10)
@0x03b: 0x00b (dec: 11)
@0x03c: 0x00c (dec: 12)
@0x03d: 0x00d (dec: 13)
@0x03e: 0x02e (dec: 46)
@0x03f: 0x00f (dec: 15)
```

INI komutları ile bellek adreslerine başlangıç değerleri atanır.

INI 0x03A 0xA; ile bellek adresi 0x03A'ya 10 atanır.

0x03B 11

0x03C 12

0x03D 13

0x03E adresine toplam atanır

0x03F adresine de en küçük sayı atanır.

LDA komutları: ACC'ye (accumulator) değerler yüklenir.

ADD komutları: ADD komutu ile verilen değerlerin toplam karşılığı bulunur. Ve değiştirilen her ADD değişkenine göre ACC güncellenir.

STO komutları: ACC'nin değeri belirtilen bellek adresine kaydedilir. STO 0x03E; ile ACC'nin değeri tüm sayıların toplamı ile bulunur ve bellek adresi 0x03E'e kaydedilir.

```

(kali㉿kali)-[~/Desktop/micro/hw/MU0]
$ python3 mu0.py test.asm
### Parsing source file ...
Recognized: INI 0x03A 0xA ; 1. sayi
Recognized: INI 0x03B 0xB ; 2. sayi
Recognized: INI 0x03C 0xC ; 3. sayi
Recognized: INI 0x03D 0xD ; 4. sayi
Recognized: INI 0x03E 0x0 ; toplam
Recognized: INI 0x03f 0xf ; en kucuk sayi
Recognized: LDA 0x03A; ACC = 1.sayi
Recognized: STO 0x03F; En kucuk sayiyi 03F adresine kaydet
Recognized: LDA 0x03B; ACC = 2.sayi
Recognized: SUB 0x03F; En kucuk sayi ile karsilastir.
Recognized: JGE 0x08; eger acc buyuk ve esit ise sonraki adrese atla
Recognized: STO 0x03F; eger acc kucuk ise 03F adresine kaydet
Recognized: LDA 0x03C; ACC = 3.sayi
Recognized: SUB 0x03F; En kucuk sayi ile karsilastir.
Recognized: JGE 0x0E; eger acc buyuk ve esit ise sonraki adrese atla
Recognized: STO 0x03F; eger acc kucuk ise 03F adresine kaydet
Recognized: ADD 0x03D; ACC = 4.sayi
Recognized: SUB 0x03F; En kucuk sayi ile karsilastir.
Recognized: JGE 0x14; eger acc buyuk ve esit ise sonraki adrese atla
Recognized: STO 0x03F; eger acc kucuk ise 03F adresine kaydet
Recognized: JMP 0x1A; Sonraki adrese gec
Recognized: STP; programi durdur.

### Memory dump before program execution:
@0x03a: 0x00a (dec: 10)
@0x03b: 0x00b (dec: 11)
@0x03c: 0x00c (dec: 12)
@0x03d: 0x00d (dec: 13)
@0x03e: 0x000 (dec: 0)
@0x03f: 0x00f (dec: 15)

### Running the program ...

### Memory dump after program end:
@0x03a: 0x00a (dec: 10)
@0x03b: 0x00b (dec: 11)
@0x03c: 0x00c (dec: 12)
@0x03d: 0x00d (dec: 13)
@0x03e: 0x000 (dec: 0)
@0x03f: 0x00a (dec: 10)

```

NI, LDA, STO, Komutları yukarıda verildi.

SUB komutları: ACC'nin değeri bir başka bellek adresindeki değerle çıkarılır.

SUB 0x03F; ile ACC'nin değeri en küçük sayıdan çıkarılır. Bu, en küçük sayıyı bulmamıza yardımcı olur.

JGE komutları: ACC'nin değeri, bir başka bellek adresindeki değere göre kontrol edilir. JGE komutları, ACC'nin değeri en küçük sayıdan büyükse veya eşitse, belirtilen adrese atlama sağlar.

JMP ve STP komutları: Program akışını yönlendirir ve sonlandırır. JMP 0x1A; ile sonraki adrese atlanır.

STP; ile program sonlandırılır.