

---

# **Sage Turu**

***Release 7.3***

**The Sage Development Team**

**Aug 16, 2016**



## CONTENTS

<b>1</b>	<b>Hesap Makinesi Olarak Sage</b>	<b>3</b>
<b>2</b>	<b>Sage ile Güçlü Hesaplamalar</b>	<b>5</b>
<b>3</b>	<b>Sage’de Algoritmaların Kullanımı</b>	<b>7</b>



Bu tur, Mathematica Book başında bulunan Mathematica turuna oldukça benzerdir.



## HESAP MAKİNESİ OLARAK SAGE

Sage komut satırında `sage:` kendiliğinden oluşur; bunu eklemeniz gerekmez. Eğer Sage defteri kullanıyorsanız herşeyi `sage:` ibaresinin devamına yazın ve hesaplanması için shift-enter tuşlarına basın.

```
sage: 3 + 5
8
```

Şapka işareti “kuvvetini almak” anlamına gelir.

```
sage: 57.1 ^ 100
4.60904368661396e175
```

$2 \times 2$  bir matrisin tersini alıyoruz.

```
sage: matrix([[1,2], [3,4]])^(-1)
[ -2      1]
[ 3/2 -1/2]
```

Burada basit bir fonksiyonun integralini alıyoruz.

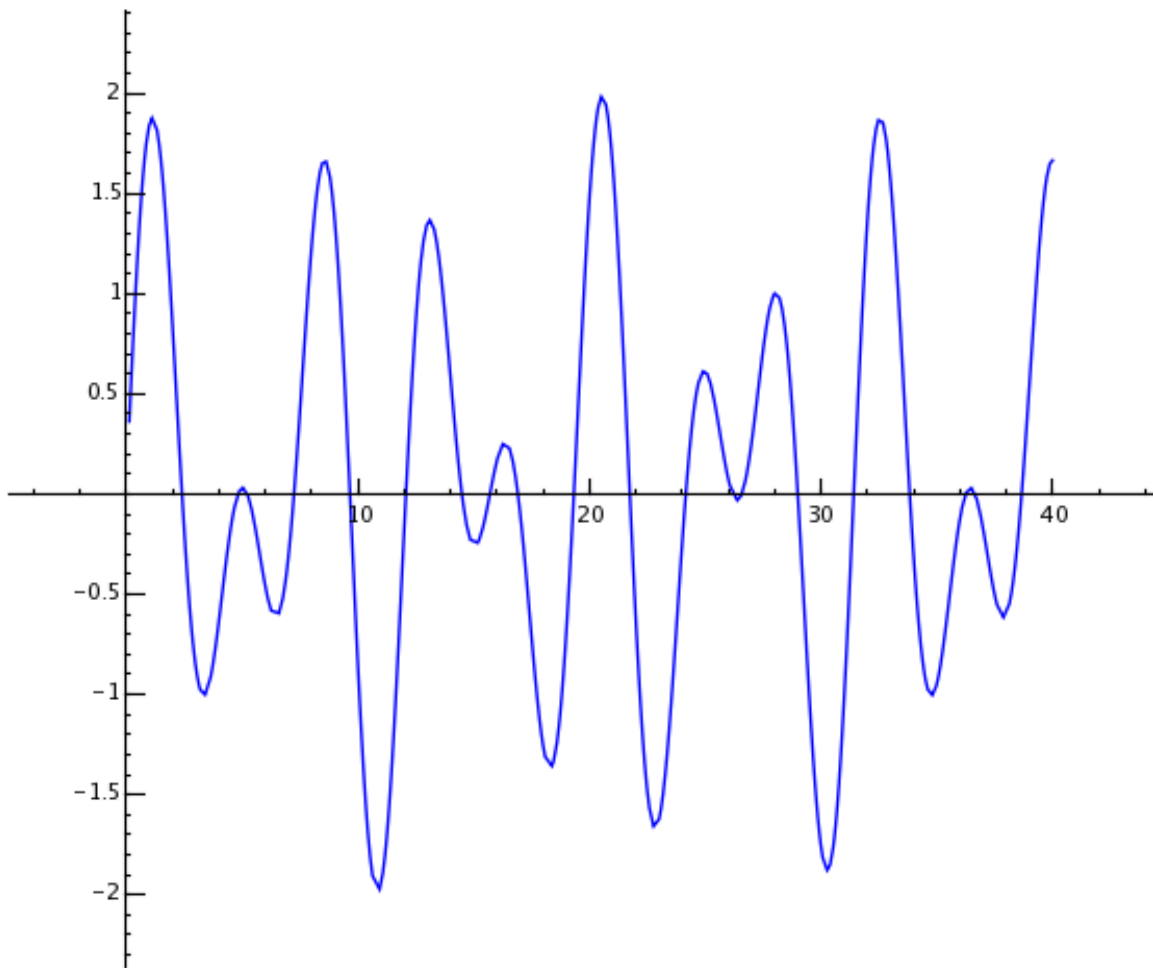
```
sage: x = var('x') # değişkeni sembolik olarak yaratıyoruz
sage: integrate(sqrt(x)*sqrt(1+x), x)
1/4*((x + 1)^(3/2)/x^(3/2) + sqrt(x + 1)/sqrt(x))/((x + 1)^2/x^2 - 2*(x + 1)/x + 1) -
↳ 1/8*log(sqrt(x + 1)/sqrt(x) + 1) + 1/8*log(sqrt(x + 1)/sqrt(x) - 1)
```

Bu komut Sage’e ikinci derece denklemi çözdürür. `==` sembolü Sage’de eşitlik anlamına gelir.

```
sage: a = var('a')
sage: S = solve(x^2 + x == a, x); S
[x == -1/2*sqrt(4*a + 1) - 1/2, x == 1/2*sqrt(4*a + 1) - 1/2]
```

Sonuç olarak eşitlikler listesi döndürülür.

```
sage: S[0].rhs()
-1/2*sqrt(4*a + 1) - 1/2
sage: show(plot(sin(x) + sin(1.6*x), 0, 40))
```





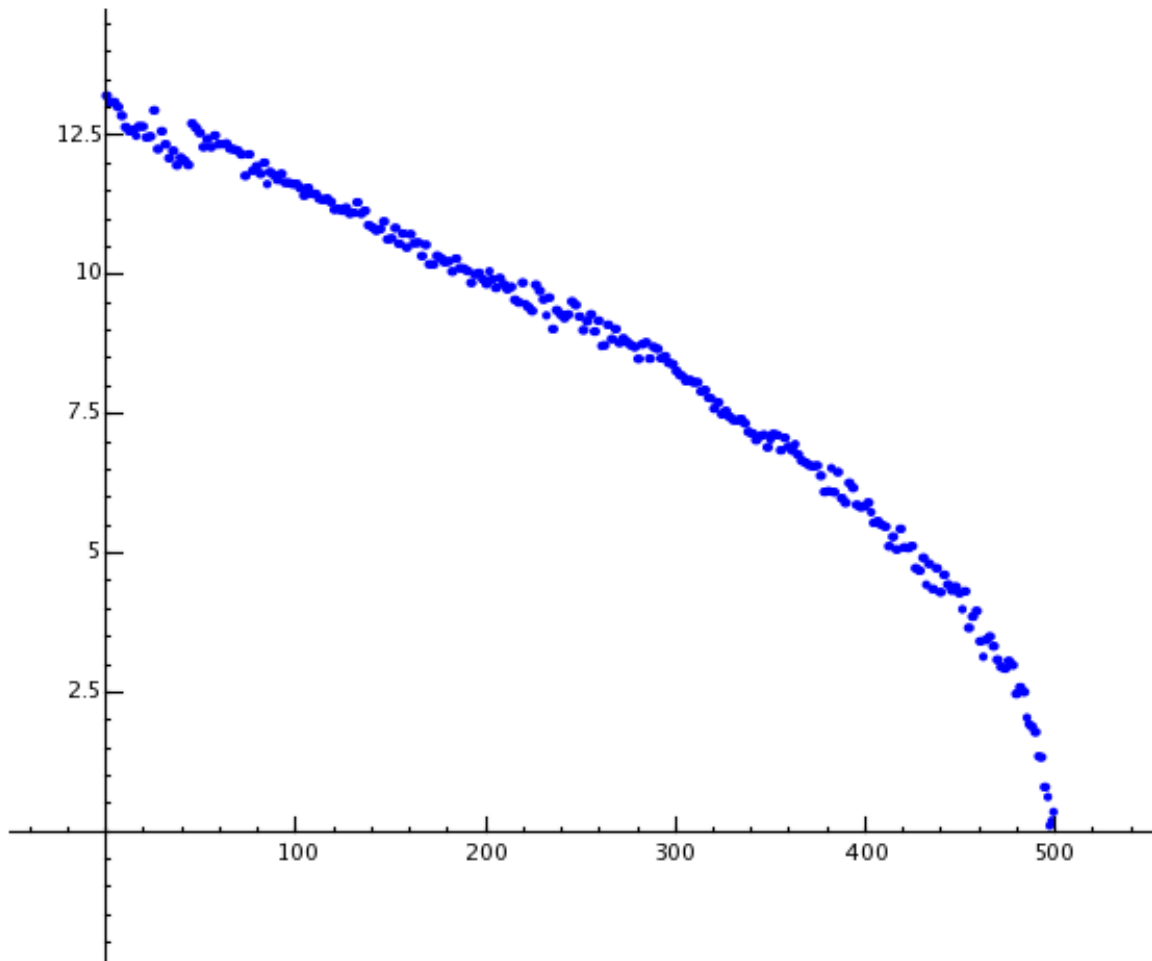
## SAGE İLE GÜÇLÜ HESAPLAMALAR

Önce rasgele sayılardan oluşan  $500 \times 500$  boyutlu bir matris oluşturuyoruz.

```
sage: m = random_matrix(RDF, 500)
```

Sage, bu matrisin özdeğerlerini birkaç saniyede bulup bunları çizdirir.

```
sage: e = m.eigenvalues() # yaklaşık 2 saniye  
sage: w = [(i, abs(e[i])) for i in range(len(e))]  
sage: show(points(w))
```



GNU Multiprecision Library (GMP) sayesinde Sage, rakam adedi milyonları hatta milyarları bulan sayılarla başa çıkabilir.

```
sage: factorial(100)
9332621544394415268169923885626670049071596826438162146859296389521759999322991560894146397615651828
sage: n = factorial(1000000) # yaklaşık 2.5 saniye
```

Bu komutla  $\pi$  sayısının en az 100 rakamı hesaplanır.

```
sage: N(pi, digits=100)
3.
↪ 14159265358979323846264338327950288419716939937510582097494459230781640628620899862803482534211706
```

Bu komutla Sage, iki değişkenden oluşan polinomu çarpanlarına ayırır.

```
sage: R.<x,y> = QQ[]
sage: F = factor(x^99 + y^99)
sage: F
(x + y) * (x^2 - x*y + y^2) * (x^6 - x^3*y^3 + y^6) *
(x^10 - x^9*y + x^8*y^2 - x^7*y^3 + x^6*y^4 - x^5*y^5 +
x^4*y^6 - x^3*y^7 + x^2*y^8 - x*y^9 + y^10) *
(x^20 + x^19*y - x^17*y^3 - x^16*y^4 + x^14*y^6 + x^13*y^7 -
x^11*y^9 - x^10*y^10 - x^9*y^11 + x^7*y^13 + x^6*y^14 -
x^4*y^16 - x^3*y^17 + x*y^19 + y^20) * (x^60 + x^57*y^3 -
x^51*y^9 - x^48*y^12 + x^42*y^18 + x^39*y^21 - x^33*y^27 -
x^30*y^30 - x^27*y^33 + x^21*y^39 + x^18*y^42 - x^12*y^48 -
x^9*y^51 + x^3*y^57 + y^60)
sage: F.expand()
x^99 + y^99
```

Yüz milyon sayısının kaç farklı biçimde pozitif tamsayıların toplamı olarak yazılabileceğini Sage'de hesaplamak 5 saniyeden kısa sürer.

```
sage: z = Partitions(10^8).cardinality() # yaklaşık 4.5 saniye
sage: str(z)[:40]
'1760517045946249141360373894679135204009'
```

## SAGE'DE ALGORİTMALARIN KULLANIMI

Sage kullanırken dünyanın en geniş açık kaynak hesaplama algoritma koleksiyonlarından biriyle çalışırsınız.