ESKİŞEHİR OSMANGAZİ ÜNİVERSİTESİ
Mühendislik Mimarlık Fakültesi
İnşaat Mühendisliği Bölümü
E-Posta: ogu.ahmet.topcu@gmail.com
Web: http://mmf2.ogu.edu.tr/atopcu

Bilgisayar Destekli

Nümerik Analiz

Ders notları 2014

Ahmet TOPÇU

$$I = \int_{a}^{b} f(x)dx$$

$$I = \int_{xa}^{xb} [\int_{ya(x)}^{yb(x)} f(x,y)\, dy]\, dx$$

$$I = \int_{xa}^{xb} \{ \int_{ya(x)}^{yb(x)} [\int_{za(x,y)}^{zb(x,y)} f(x,y,z)\ dz]\, dy \} dx$$

39

PROGRAMLAR: Belirli integral hesabı

- Simpson
- Romberg
- GaussLegendre
- AdapteSimpson
- RecursiveAdapteSimpson
- TanhKurali
- Yamuk2
- Simpson2
- Romberg2
- GaussLegendre2
- GaussLegendre3

39. PROGRAMLAR: Tek ve çok katlı belirli integral hesabı

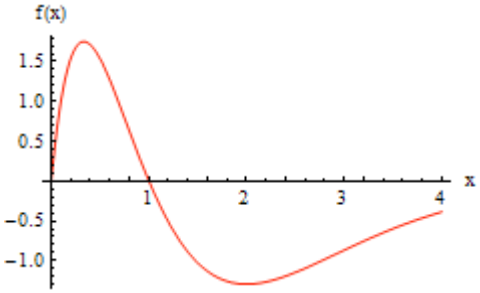
Bu bölümde uygulamada kullanılan integrasyon metotlarının QBASIC programları ve çözüm örnekleri verilecektir. Her hangi bir fonksiyon için biri iyi sonuç verirken bir diğeri çözüm vermeyebilir.

1. **Simpson :** $I=\int_a^b f(x)\,dx$ integralini Simpson kuralı ile hesaplar.
2. **Romberg:** $I=\int_a^b f(x)\,dx$ integralini Romberg metodu ile hesaplar.
3. **GaussLegendre:** $I=\int_a^b f(x)\,dx$ integralini Gauss-Legendre metodu ile hesaplar.
4. **AdabteSimpson:** $I=\int_a^b f(x)\,dx$ integralini adabte(adabtive) Simpson metodu ile hesaplar.
5. **RecursiveAdapteSimpson:** $I=\int_a^b f(x)\,dx$ integralini Recursive adabte Simpson metodu ile hesaplar.
6. **TanhKurali:** $I=\int_a^b f(x)\,dx$ integralini Tanh kuralı ile hesaplar.
7. **Yamuk2:** $I=\int_a^b [\int_{ya(x)}^{yb(x)} f(x,y)\,dy]\,dx$ integralini yamuk kuralı ile hesaplar.
8. **Simpson2:** $I=\int_a^b [\int_{ya(x)}^{yb(x)} f(x,y)\,dy]\,dx$ integralini Simpson metodu ile hesaplar.
9. **Romberg2:** $I=\int_a^b [\int_{ya(x)}^{yb(x)} f(x,y)\,dy]\,dx$ integralini Romberg metodu ile hesaplar.
10. **GaussLegendre2:** $I=\int_{xa}^{xb} [\int_{ya(x)}^{yb(x)} f(x,y)\,dy]\,dx$ integralini Gauss-Legendre metodu ile hesaplar.
11. **GaussLegendre3:** $I=\int_{xa}^{xb} \{\int_{ya(x)}^{yb(x)} [\int_{za(x,y)}^{zb(x,y)} f(x,y,z)\,dz]\,dy\}\,dx$ integralini Gauss-Legendre metodu ile hesaplar.

Tek katlı integral örnekleri:

Yukarıda adı verilen programlar ile birçok fonksiyonun tek katlı integralleri hesaplanarak sonuçlar karşılaştırılmıştır. Karşılaştırmayı kolaylaştırabilmek için integrali hesaplanan fonksiyonun ayrıca grafiği de çizilmiştir. Tüm metotlarda [a,b] aralığı 20 ye bölünmüş, hassasiyet=0.000001 alınmıştır(virgülden sonraki 6. hane yuvarlanmış). Aynı integraller Mathematica[1] ile de hesaplanmış, Mathematica sonuçları doğru varsayılarak, her metodun hatalı haneleri koyu kırmızı olarak işaretlenmiştir.

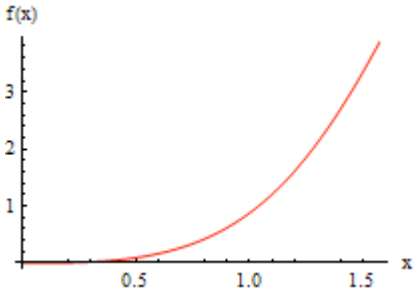
$$I_1=\int_1^4 \frac{13(x-x^2)}{\sqrt{e^{3x}}}\,dx$$



Metot	I_1
Simpson	-2.630988
Romberg	-2.630988
GaussLegendre	-2.630988
AdapteSimpson	-2.630988
RecursiveAdapteSimpson	-2.630988
TanhKurali	-2.630987
Mathematica 4	-2.630988

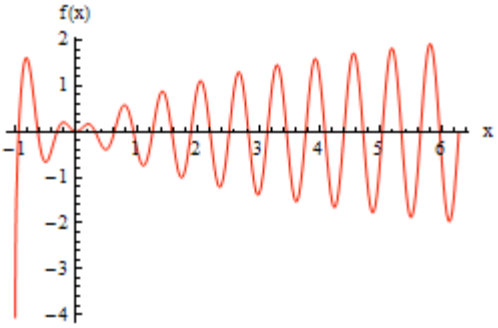
[1] Mathematica 4: Wolfram Reserch tarafından geliştirilmiş, analitik ve nümerik hesap yapabilen profesyonel yazılım

$$I_2 = \int_0^{\pi/2} \frac{x^3}{\sqrt{1+Cos^2(x)}}\,dx$$



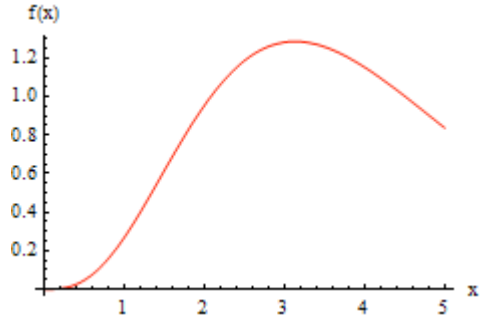
Metot	I_2
Simpson	1.437642
Romberg	1.437642
GaussLegendre	1.437642
AdapteSimpson	1.437642
RecursiveAdapteSimpson	1.437642
TanhKurali	1.437642
Mathematica 4	1.437642

$$I_3 = \int_0^{2\pi} Ln(1+x)Sin(10x)\,dx$$



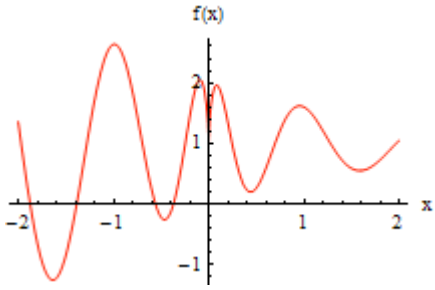
Metot	I_3
Simpson	-0.197627
Romberg	-0.197627
GaussLegendre	-0.197627
AdapteSimpson	-0.19762**8**
RecursiveAdapteSimpson	-0.197627
TanhKurali	-0.197627
Mathematica 4	-0.197627

$$I_4 = \int_0^{5} \frac{x^3}{e^x+1}\,dx$$



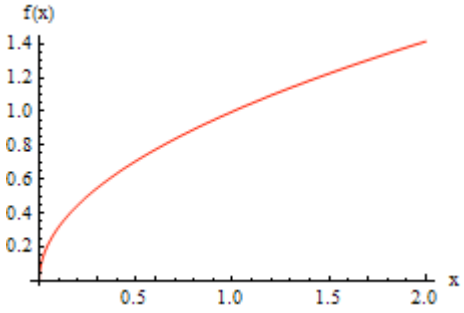
Metot	I_4
Simpson	4.095902
Romberg	4.095902
GaussLegendre	4.095902
AdapteSimpson	4.095902
RecursiveAdapteSimpson	4.095902
TanhKurali	4.09590**1**
Mathematica 4	4.095902

$$I_5 = \int_{-2}^{2} \frac{Sin(8\sqrt[3]{x^2})}{\sqrt{e^x}} + 1\,dx$$



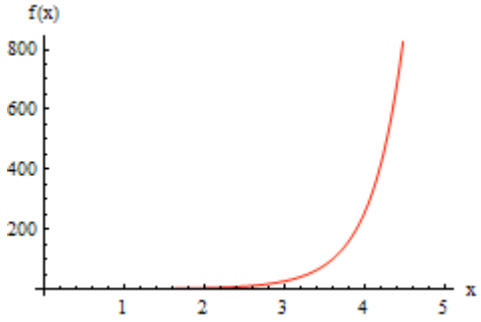
Metot	I_5
Simpson	3.38886**8**
Romberg	3.38886**8**
GaussLegendre	3.388869
AdapteSimpson	3.38886**8**
RecursiveAdapteSimpson	3.38886**8**
TanhKurali	3.38886**8**
Mathematica 4	3.388869

$$I_6 = \int_0^{1} \sqrt{x}\,dx$$



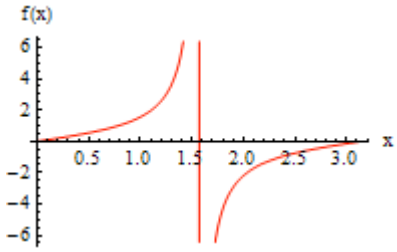
Metot	I_6
Simpson	0.66666**6**
Romberg	0.66666**6**
GaussLegendre	0.666667
AdapteSimpson	0.666667
RecursiveAdapteSimpson	0.666667
TanhKurali	0.666667
Mathematica 4	0.666667

$$I_7 = \int_0^5 x^x dx$$



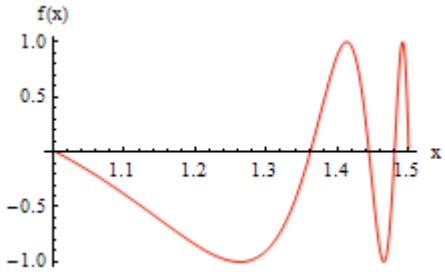
Metot	I_7
Simpson	1241.816715
Romberg	1241.816715
GaussLegendre	1241.816715
AdapteSimpson	1241.816719
RecursiveAdapteSimpson	1241.816719
TanhKurali	1241.816714
Mathematica 4	1241.816715

$$I_8 = \int_0^{\pi/2} Tan(x)dx$$



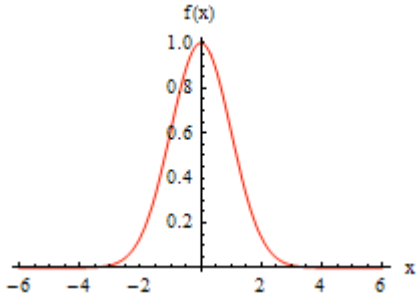
Metot	I_8
Simpson	Yakınsamadı
Romberg	Yakınsamadı
GaussLegendre	Yakınsamadı
AdapteSimpson	**≈10^13**
RecursiveAdapteSimpson	Yığın taşması (Stack overflow)
TanhKurali	Yakınsamadı
Mathematica 4	Yakınsamadı

$$I_9 = \int_1^{1.5} Cos(Tan(x))dx$$



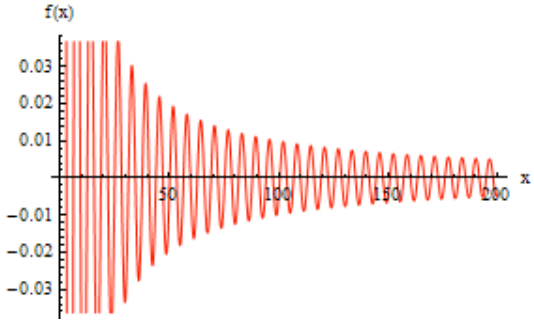
Metot	I_9
Simpson	-0.166312
Romberg	-0.166312
GaussLegendre	-0.166312
AdapteSimpson	-0.166312
RecursiveAdapteSimpson	-0.166312
TanhKurali	-0.166312
Mathematica 4	-0.166312

$$I_{10} = \int_{-2}^{2} e^{-\frac{x^2}{2}} dx$$



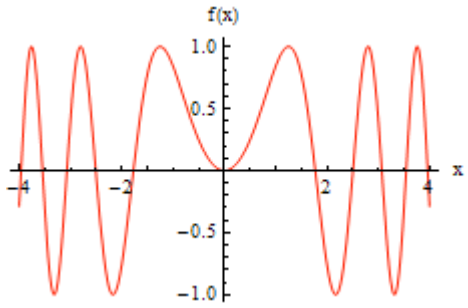
Metot	I_{10}
Simpson	2.392576
Romberg	2.392576
GaussLegendre	2.392576
AdapteSimpson	2.392576
RecursiveAdapteSimpson	2.392576
TanhKurali	2.392575
Mathematica 4	2.392576

$$I_{11} = \int_0^{200} \frac{Sin(x)}{x} dx$$



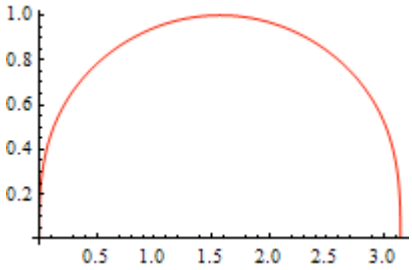
Metot	I_{11}
Simpson	**Sayı taşması**
Romberg	**Sayı taşması**
GaussLegendre	1.568382
AdapteSimpson	**Sayı taşması**
RecursiveAdapteSimpson	**Sayı taşması**
TanhKurali	1.568380
Mathematica 4	1.568320

$$I_{12} = \int_{-4}^{4} Sin(x^2)dx$$



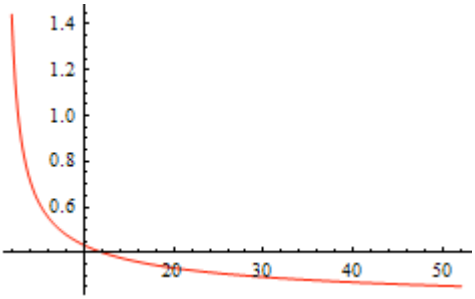
Metot	I_{12}
Simpson	1.494268
Romberg	1.494268
GaussLegendre	1.494268
AdapteSimpson	1.494267
RecursiveAdapteSimpson	1.494267
TanhKurali	1.494267
Mathematica 4	1.494268

$$I_{13} = \int_{0}^{\pi} \sqrt[3]{Sin(x)}dx$$



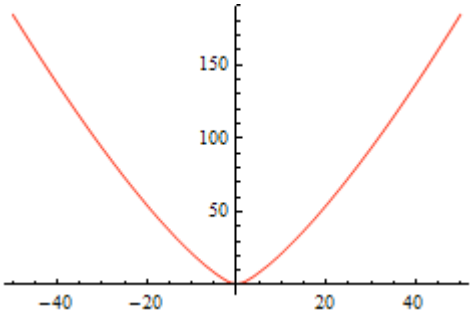
Metot	I_{13}
Simpson	2.587109
Romberg	2.587109
GaussLegendre	2.587110
AdapteSimpson	2.586384
RecursiveAdapteSimpson	Yığın taşması (Stack overflow)
TanhKurali	2.587109
Mathematica 4	2.587110

$$I_{14} = \int_{2}^{50} \frac{1}{Ln(x)}dx$$



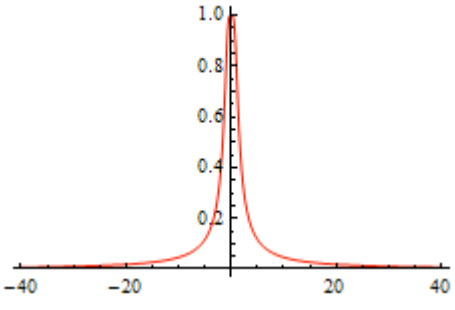
Metot	I_{14}
Simpson	17.423533
Romberg	17.423533
GaussLegendre	17.423533
AdapteSimpson	17.423533
RecursiveAdapteSimpson	17.423534
TanhKurali	17.423524
Mathematica 4	17.423533

$$I_{15} = \int_{-40}^{40} \sqrt[3]{x^4+1}dx$$



Metot	I_{15}
Simpson	4691.846448
Romberg	4691.846447
GaussLegendre	4691.846447
AdapteSimpson	4691.846451
RecursiveAdapteSimpson	4691.846450
TanhKurali	4691.846435
Mathematica 4	4691.846447

$$I_{16} = \int_{-40}^{40} \frac{1}{\sqrt[3]{x^4+1}}dx$$



Metot	I_{16}
Simpson	6.027112
Romberg	6.027112
GaussLegendre	6.027112
AdapteSimpson	6.027113
RecursiveAdapteSimpson	6.027114
TanhKurali	6.027099
Mathematica 4	6.027099

Karşılaştırma

	Simpson	Romberg	GaussLegendre	AdapteSimpson	RecursiveAdapteSimpson	TanhKuralı	Mathematica
I_1	-2.630988	-2.630988	-2.630988	-2.630988	-2.630988	-2.630987	-2.630988
I_2	1.437642	1.437642	1.437642	1.437642	1.437642	1.437642	1.437642
I_3	-0.197627	-0.197627	-0.197627	-0.197628	-0.197627	-0.197627	-0.197627
I_4	4.095902	4.095902	4.095902	4.095902	4.095902	4.095901	4.095902
I_5	3.388868	3.388868	3.388869	3.388638	3.388868	3.388868	3.388869
I_6	0.666666	0.666666	0.666667	0.666667	0.666667	0.666667	0.666667
I_7	1241.816715	1241.816715	1241.816715	1241.816719	1241.816719	1241.816714	1241.816715
I_8	Yakınsamadı	Yakınsamadı	Yakınsamadı	**≈10^{13}**	**Yığın taşması (Stack overflow)**	Yakınsamadı	Yakınsamadı
I_9	-0166312	-0166312	-0166312	-0166312	-0166312	-0166312	-0166312
Analitik çözümü olmayan integraller:							
I_{10}	2.392576	2.392576	2.392576	2.392576	2.392576	2.392575	2.392576
I_{11}	**Sayı Taşması**	**Sayı Taşması**	1.568382	**Sayı Taşması**	**Sayı Taşması**	1.568380	1.568320
I_{12}	1.494268	1.494268	1.494268	1.494267	1.494266	1.494267	1.494268
I_{13}	2.587109	2.587109	2.587110	2.586384	**Yığın taşması (Stack overflow)**	2.587109	2.587110
I_{14}	17.423533	17.423533	17.423533	17.423533	17.423534	17.423524	17.423533
I_{15}	4691.846448	4691.846447	4691.846447	4691.846451	4691.846450	4691.846435	4691.846447
I_{16}	6.027112	6.027112	6.027112	6.027113	6.027114	6.027099	6.027112

YORUM:
Yukarıdaki tablo incelendiğinde, tüm metotların, uygulama açısından, tatminkâr sonuç verdiği söylenebilir. Her integrali hesaplayabilen ideal bir metot yoktur. Bir sıralama yapmak gerekirse; GaussLegendre, Simpson, Romberg en iyi integrasyon metodudur denilebilir.

Özellikle $I_8 = \int_0^{\pi/2} Tan(x)dx$ integraline dikkat çekmek gerekir. Bu integralin teorik sonucu $I_8 = \int_0^{\pi/2} Tan(x)dx = +\infty$ dur, bilgisayarda hesaplamak mümkün değildir. Çünkü bilgisayarda ∞ büyük yoktur. AdapteSimpson **≈10^{13}** gibi, uygulama açısından oldukça büyük bir sayı vermiş, diğerleri yakınsamamıştır. Bu sonuç, teorik olarak yanlış olmakla birlikte, AdapteSimpson metodunun [a,b] aralığında sürekli olan, fakat eğimi ani ve hızla değişen fonksiyonlarda daha iyi sonuç vereceği anlamına gelir.

$I_{11} = \int_0^{200} \frac{Sin(x)}{x} dx$ integralinde bazı metotların **Sayı Taşması** vermesi beklenen bir sonuçtur. Çünkü integralin alt sınırı için sıfır bölüm oluşmaktadır. Aynı integral için GaussLegendre ve Tanh metotlarının sonuç vermesi de doğaldır, çünkü bu metotlar integral aralığını değiştirerek hesaplarlar.

İki katlı integral örnekleri:

Aşağıdaki iki katlı integraller adı verilen programlar ile hesaplanmış, sonuçlar karşılaştırılmıştır. Hassasiyet=0.01 alınmıştır. Aynı integraller Mathematica ile de hesaplanmış, Mathematica sonuçları doğru varsayılarak, her metodun hatalı hane sayısı koyu kırmızı olarak işaretlenmiştir.

	Yamuk2	Simpson2	Romberg2	GaussLegendre2	Mathematica 4
$I_1 = \int_{-2}^{2}\int_{-1}^{1} e^{x^2y^2}\,dydx$	5.94	5.94	5.94	5.94	5.94
$I_2 = \int_{0}^{\pi/2}\int_{0}^{1} (1-y^2)Cos(x)^2\,dydx$	0.52	0.52	0.52	0.52	0.52
$I_3 = \int_{0}^{2}\int_{0}^{\sqrt{(4-x^2)/2}} (4-x^2-2y^2)dydx$	4.44	4.44	4.44	4.44	4.44
$I_4 = \int_{0}^{2}\int_{1-x}^{e^{-x}} \sqrt{x+y}\,dydx$	1.00	1.00	1.00	1.00	1.00
$I_5 = \int_{0}^{\pi/2}\int_{Tan(x/2)}^{Sin(x)} (x+y)dydx$	0.43	0.43	0.43	0.43	0.43
$I_6 = \int_{0}^{2}\int_{-x^x}^{x^x} (x^2+y^2)dydx$	20.80	20.80	20.80	20.80	20.80
$I_7 = \int_{-4}^{4}\int_{-\sqrt{\lvert16-x^2\rvert}}^{\sqrt{\lvert16-x^2\rvert})} \sqrt{\lvert16-x^2-y^2\rvert}\,dydx$	134.0**3**	134.0**3**	134.04	134.04	134.04
$I_8 = \int_{0}^{2}\int_{x^2}^{\sqrt{8x}} e^{-x\sqrt{y}}\,dydx$	1.0**2**	1.0**1**	1.0**1**	1.0**2**	1.00

YORUM:
Yukarıdaki tablo incelendiğinde, tüm metotların, uygulama açısından, iyi sonuç verdiği söylenebilir. GaussLegendre2 ve Simpson2 hızlı yakınsama sergilerken Yamuk2 ve özellikle Romberg2 programlarının yavaş kaldığı gözlenmiştir.

İki katlı integral örneklerinin tümünde hassasiyet=0.01 alınmıştır. Hassasiyetin daha küçük seçilmesi, örneğin: hassasiyet=0.00001, doğru hane sayısını hemen hiç değiştirmemektedir. Sonuç olarak, iki katlı integrallerde, hassasiyetin küçük seçilmesi hiçbir kazanç sağlamazken hesap süresinin aşırı uzamasına neden olmaktadır.

Üç katlı integral örnekleri:

Aşağıdaki üç katlı integraller Hassasiyet=0.01 alınarak GaussLegendre3 programı ile hesaplanmış, sonuçlar gerçek değerler ile karşılaştırılmıştır.

	GaussLegendre3	Gerçek değer
$I_1 = \int_0^1 \int_0^2 \int_0^3 x^2 y^2 z^2 dz\, dy\, dx$	8.00	8.00
$I_2 = \int_0^2 \int_0^3 \int_0^{\sqrt{9-y^2}} z^2 dz\, dy\, dx$	31.81	31.81
$I_3 = \int_0^4 \int_0^{\sqrt{4x-x^2}} \int_0^{2x^2 y} dz\, dy\, dx$	51.20	51.20
$I_4 = \int_{-2}^2 \int_{-\sqrt{4-x^2}}^{\sqrt{4-x^2}} \int_0^{\sqrt{4-x^2-y^2}} z\ dz\, dy\, dx$	12.57	12.57
$I_5 = \int_0^6 \int_0^{(12-2x)/3} \int_0^{(12-2x-3y)/6} dz\, dy\, dx$	8.00	8.0
$I_6 = \int_{-\pi/2}^{\pi} \int_{-Sin(x)^2}^{Sin(x)^2} \int_{-x-y}^{x+y} Sin(x)+Cos(x)+Sin(z) dz\, dy\, dx$	17.30	17.30

YORUM:

Üç katlı integral örneklerinin tümünde hassasiyet=0.01 alınmıştır. Hassasiyetin daha küçük seçilmesi, örneğin: hassasiyet=0.00001, doğru hane sayısını hemen hiç değiştirmemektedir. Sonuç olarak, çok katlı integrallerde, hassasiyetin küçük seçilmesi hiçbir kazanç sağlamazken hesap süresinin aşırı uzamasına neden olmaktadır.

```
'-----------------------Ana program Simpson-----------------------
' f(x) fonksiyonun [a,b] aralığında tek katlı entegralini hesaplar
' Ahmet TOPÇU, Osmangazi Üniversitesi, Eskişehir, 1994
' Metot: Simpson

' Veri:
'  a: integralın alt sınırı
'  b: integralın üst sınırı
'  f(x) : integrali hesaplanacak fonksiyon
'  Hassasiyet: hassasiyet

'  f(x) fonksiyonu ana programda DEF fnF(x)=... ile tanımlanmalıdır.

' çıktı:
'  Toplamintegral: Hesaplanan integral değeri
'  iHata=0 : integral yakınsadı
'  iHata<>0 : İntegral yakınsamadı

' çağrılan alt program: Simpson
'-----------------------------------------------------------------

DEFINT A-I
DEFDBL A-H, O-Z
DECLARE SUB Simpson (ai, bi, Hassasiyet, Altintegral, iHata)

' Aralığın ve f(x) fonksiyonunun tanımlanması
   Pi = 4 * ATN(1)  ' Pi=3.14... sayısı

   a = 1: b = 4: DEF fnF (x) = 13 * (x - x * x) * EXP(-3 * x / 2)
   'a = 0: b = Pi / 2: DEF fnF (x) = x ^ 3 / SQR(1 + COS(x) ^ 2)
   'a = 0: b = 2 * pi: DEF fnF (x) = LOG(1 + x) * SIN(10 * x)
   'a = 0: b = 5: DEF fnF (x) = x ^ 3 / (EXP(x) + 1)
   'a = -2: b = 2: DEF fnF (x) = EXP(-.5 * x) * SIN(8 * x ^ 2 ^ (1 / 3)) + 1
   'a = 0: b = 1: DEF fnF (x) = SQR(x)
   'a = 0: b = 5: DEF fnF (x) = x ^ x
   'a = 0: b = Pi / 2: DEF fnF (x) = TAN(x)
   'a = 1: b = 1.5: DEF fnF (x) = COS(TAN(x))

' analitik çözümü olmayan integraller:
   'a = -2: b = 2: DEF fnF (x) = EXP(-x ^ 2 / 2)
   'a = 0: b = 1: DEF fnF (x) = SIN(x) / x
   'a = -4: b = 4: DEF fnF (x) = SIN(x ^ 2)
   'a = 0: b = Pi: DEF fnF (x) = SIN(x) ^ (1 / 3)
   'a = 2: b = 50: DEF fnF (x) = 1 / LOG(x)
   'a = -40: b = 40: DEF fnF (x) = (1 + x ^ 4) ^ (1 / 3)
   'a = -40: b = 40: DEF fnF (x) = 1 / (1 + x ^ 4) ^ (1 / 3)

   CLS

   Hassasiyet = .000001' değiştirilebilir,(1E-4 - 1E-12 arası normal)
   MaxAralik = 20      ' değiştirilebilir,(2-100 arası normal)

   Delta = (b - a) / MaxAralik
   bi = a

   Toplamintegral = 0
   FOR iAralik = 1 TO MaxAralik
    ai = bi
    bi = ai + Delta
    IF iAralik = MaxAralik THEN bi = b

     CALL Simpson(ai, bi, Hassasiyet, Altintegral, iHata)

     Toplamintegral = Toplamintegral + Altintegral

     IF iHata <> 0 THEN
      PRINT "İntegral yakınsamadı(Simpson)"
      PRINT "Hesaplanan yaklaşık değer="; Toplamintegral
      END
     END IF

   NEXT iAralik

   PRINT "İntegral(Simpson)="; Toplamintegral

END ' Simpson ana sonu
```

Tek katlı integral:
Simpson integrasyon metodu
(ana program)

İntegrali hesaplanacak fonksiyonlar

' işareti olmayan fonksiyon hesaplanır. Bu işareti içeren fonksiyon devre dışıdır.

Tek katlı integral:
Simpson integrasyon metodu
(alt program)

```
SUB Simpson (ai, bi, Hassasiyet, Altintegral, iHata)
'---------------------------------------------------------------------
' f(x) fonksiyonunun [ai,bi] aralığında tek katlı integralini hesaplar
' Ahmet TOPÇU, Osmangazi Üniversitesi, Eskişehir, 1994
' kullanılan metod: Simpson

' Veri:
'  a: integralın alt sınırı
'  b: integralın üst sınırı
'  f(x) : integrali hesaplanacak fonksiyon
'  Hassasiyet: hassasiyet

'  f(x) fonksiyonu çağıran programda DEF fnF(x)=... ile tanımlanmalıdır.

' çıktı:
'  Toplamintegral: Hesaplanan integral değeri
'  iHata=0 : integral yakınsadı
'  iHata<>0 : İntegral yakınsamadı

' çağrılan program: yok
'---------------------------------------------------------------------
    iHata = 0
    m = 14  ' m>14 tam sayı taşmasına neden olabilir

    h = .5 * (bi - ai)
    s = 2 * fnF(ai + h)
    r = fnF(ai) + fnF(bi)
    c = h / 3
    EskiAltintegral = c * (r + 2 * s)

    FOR i = 2 TO m
     h = .5 * h
     r = r + s

     s = 0
     iAltaralikSayisi = 2 ^ i
     FOR iAltAralikNo = 1 TO iAltaralikSayisi STEP 2
      x = ai + h * iAltAralikNo
      IF iAltAralikNo = iAltaralikSayisi THEN x = bi
      s = s + fnF(x)
     NEXT iAltAralikNo

     s = 2 * s
     c = .5 * c
     Altintegral = c * (r + 2 * s)

     Hata = ABS(Altintegral - EskiAltintegral)
     IF i > 2 AND Hata <= Hassasiyet THEN EXIT SUB

     EskiAltintegral = Altintegral

    NEXT i

    iHata = 1

END SUB ' Simpson sonu
```

Tek katlı integral:
Romberg integrasyon metodu
(ana program)

```
'----------------------Ana Program Romberg-------------------------
' f(x) fonksiyonunun [a,b] aralığında tek katlı integralini hesaplar
' Ahmet TOPÇU, Osmangazi Üniversitesi, Eskişehir, 1994
' kullanılan metod: Romberg

' Veri:
'  a: integralın alt sınırı
'  b: integralın üst sınırı
'  f(x) : integrali hesaplanacak fonksiyon
'  Hassasiyet: hassasiyet

'  f(x) fonksiyonu ana programda DEF fnF(x)=... ile tanımlanmalıdır.

' çıktı:
'  Toplamintegral: Hesaplanan integral değeri
'  iHata=0 : integral yakınsadı
'  iHata<>0 : İntegral yakınsamadı

' Çağrılan alt program: Romberg
'-----------------------------------------------------------------
DEFINT I-N
DEFDBL A-H, O-Z
DECLARE SUB Romberg (ai, bi, Hassasiyet, Altintegral, iHata)

' Aralığın ve f(x) fonksiyonunun tanımlanması
   Pi = 4 * ATN(1)  ' Pi=3.14... sayısı

   a = 1: b = 4: DEF fnF (x) = 13 * (x - x * x) * EXP(-3 * x / 2)
   'a = 0: b = Pi / 2: DEF fnF (x) = x ^ 3 / SQR(1 + COS(x) ^ 2)
   'a = 0: b = 2 * pi: DEF fnF (x) = LOG(1 + x) * SIN(10 * x)
   'a = 0: b = 5: DEF fnF (x) = x ^ 3 / (EXP(x) + 1)
   'a = -2: b = 2: DEF fnF (x) = EXP(-.5 * x) * SIN(8 * x ^ 2 ^ (1 / 3)) + 1
   'a = 0: b = 1: DEF fnF (x) = SQR(x)
   'a = 0: b = 5: DEF fnF (x) = x ^ x
   a = 0: b = Pi / 2: DEF fnF (x) = TAN(x)
   a = 1: b = 1.5: DEF fnF (x) = COS(TAN(x))

' analitik çözümü olmayan integraller:
   'a = -2: b = 2: DEF fnF (x) = EXP(-x ^ 2 / 2)
   'a = 0: b = 1: DEF fnF (x) = SIN(x) / x
   'a = -4: b = 4: DEF fnF (x) = SIN(x ^ 2)
   'a = 0: b = Pi: DEF fnF (x) = SIN(x) ^ (1 / 3)
   'a = 2: b = 50: DEF fnF (x) = 1 / LOG(x)
   'a = -40: b = 40: DEF fnF (x) = (1 + x ^ 4) ^ (1 / 3)
   'a = -40: b = 40: DEF fnF (x) = 1 / (1 + x ^ 4) ^ (1 / 3)

   CLS

   Hassasiyet = .000001' değiştirilebilir, (1E-4 - 1E-12 arası normal)
   MaxAralik = 20 ' değiştirilebilir, (2-100 arası normal)

   Delta = (b - a) / MaxAralik
   bi = a

   Toplamintegral = 0
   FOR iAralikNo = 1 TO MaxAralik
    ai = bi: bi = ai + Delta
    IF iAralikNo = MaxAralik THEN bi = b

    CALL Romberg(ai, bi, Hassasiyet, Altintegral, iHata)

    Toplamintegral = Toplamintegral + Altintegral

    IF iHata <> 0 THEN
     PRINT "İntegral yakınsamadı(Romberg)"
     PRINT "hesaplananyaklaşık değer="; Toplamintegral
     END
    END IF

   NEXT iAralikNo

   PRINT "İntegral(Romberg)="; Toplamintegral

END ' Romberg ana
```

Tek katlı integral:
Romberg integrasyon metodu
(alt program)

```
SUB Romberg (ai, bi, Hassasiyet, Altintegral, iHata)
'---------------------------------------------------------------------
' f(x) fonksiyonunun [ai,bi] aralığında tek katlı integralini hesaplar
' Ahmet TOPÇU, Osmangazi Üniversitesi, Eskişehir, 1994
' kullanılan metod: Romberg

' Veri:
'  a: integralın alt sınırı
'  b: integralın üst sınırı
'  f(x) : integrali hesaplanacak fonksiyon
'  Hassasiyet: hassasiyet

'  f(x) fonksiyonu çağıran programda DEF fnF(x)=... ile tanımlanmalıdır.

' çıktı:
'  Toplamintegral: Hesaplanan integral değeri
'  iHata=0 : integral yakınsadı
'  iHata<>0 : İntegral yakınsamadı

' çağrılan program: yok
'---------------------------------------------------------------------
   DIM Sayici1 AS LONG  ' 4 Byte(32 bit) integer
   DIM Sayici2 AS LONG
   DIM i AS LONG, m AS LONG

   Maxiterasyon = 10 ' Romberg iterasyon sayısı(4-12 arası normal)

   IF Maxiterasyon < 4 THEN Maxiterasyon = 4
   n = Maxiterasyon + 1
   n = (n * n - n) / 2 + n

   DIM Tablo(n)  ' Romberg tablosu
   m = 2 ^ 12' Maksimum alt aralık sayısı


   Delta = bi - ai
   Deltax = Delta
   Sayici1 = 1
   Sayici2 = 2

   EskiAltintegral = 0
   Tablo(1) = .5 * (fnF(ai) + fnF(bi))

' Yamuk kuralı
   FOR iterasyonNo = 1 TO Maxiterasyon
    x = ai + .5 * Deltax
    fx = 0
     FOR i = 1 TO Sayici1
      fx = fx + fnF(x)
      x = x + Deltax
    NEXT i
    t1 = .5 * Tablo(1) + fx / Sayici2

' Romberg tablosu
    FOR i = 1 TO iterasyonNo
     t2 = t1 + (t1 - Tablo(i)) / (4 ^ i - 1)
     Tablo(i) = t1
     t1 = t2
    NEXT i

    Altintegral = Delta * t1
    Hata = ABS(Altintegral - EskiAltintegral)
    IF iterasyonNo > 2 AND Hata <= Hassasiyet GOTO 1

    EskiAltintegral = Altintegral
    Tablo(iterasyonNo + 1) = t1
    Sayici1 = 2 * Sayici1
    Sayici2 = 2 * Sayici2
    Deltax = .5 * Deltax

    IF Sayici1 > m OR Sayici2 > m THEN EXIT FOR
   NEXT iterasyonNo

   iHata = 1
   EXIT SUB

1  iHata = 0

END SUB ' Romberg sonu
```

Tek katlı integral: GaussLegendre integrasyon metodu (ana program)

```
'----------------------Ana program GaussLegendre----------------------
' f(x) fonksiyonun [a,b] aralığında tek katlı integralini hesaplar
' Ahmet TOPÇU, Osmangazi Üniversitesi, Eskişehir, 1994
' Kullanılan metot: Gauss-Legendre

' Veri:
'  a: integralın alt sınırı
'  b: integralın üst sınırı
'  f(x) : integrali hesaplanacak fonksiyon
'  Hassasiyet: hassasiyet

'  f(x) fonksiyonu ana programda DEF fnF(x)=... ile tanımlanmalıdır.

' çıktı:
'  Toplamintegral: Hesaplanan integral değeri
'  iHata=0 : integral yakınsadı
'  iHata<>0 : İntegral yakınsamadı

' çağrılan alt program: GaussLegendre
'---------------------------------------------------------------------------

DEFINT I-N
DEFDBL A-H, O-Z
DECLARE SUB GaussLegendre (a, b, Hassasiyet, Toplamintegral, iHata)
DECLARE SUB GauLeg (Ai, Bi, x(), w(), n)

' Aralığın ve f(x) fonksiyonunun tanımlanması
   Pi = 4 * ATN(1)  ' Pi=3.14... sayısı

   a = 1: b = 4: DEF fnf (x) = 13 * (x - x * x) * EXP(-3 * x / 2)
   'a = 0: b = Pi / 2: DEF fnf (x) = x ^ 3 / SQR(1 + COS(x) ^ 2)
   'a = 0: b = 2 * pi: DEF fnf (x) = LOG(1 + x) * SIN(10 * x)
   'a = 0: b = 5: DEF fnf (x) = x ^ 3 / (EXP(x) + 1)
   'a = -2: b = 2: DEF fnf (x) = EXP(-.5 * x) * SIN(8 * x ^ 2 ^ (1 / 3)) + 1
   'a = 0: b = 1: DEF fnf (x) = SQR(x)
   'a = 0: b = 5: DEF fnf (x) = x ^ x
   'a = 0: b = Pi / 2: DEF fnf (x) = TAN(x)
   'a = 1: b = 1.5: DEF fnf (x) = COS(TAN(x))

' analitik çözümü olmayan integraller:
   'a = -2: b = 2: DEF fnf (x) = EXP(-x ^ 2 / 2)
   'a = 0: b = 200: DEF fnf (x) = SIN(x) / x
   'a = -4: b = 4: DEF fnf (x) = SIN(x ^ 2)
   'a = 0: b = Pi: DEF fnf (x) = SIN(x) ^ (1 / 3)
   'a = 2: b = 50: DEF fnf (x) = 1 / LOG(x)
   'a = -40: b = 40: DEF fnf (x) = (1 + x ^ 4) ^ (1 / 3)
   'a = -40: b = 40: DEF fnf (x) = 1 / (1 + x ^ 4) ^ (1 / 3)

   CLS

   Hassasiyet = .000001' değiştirilebilir(1E-4 -1E-12 arası normal)

   CALL GaussLegendre(a, b, Hassasiyet, Toplamintegral, iHata)

   IF iHata <> 0 THEN
    PRINT "İntegral yakınsamadı(GaussLegendre)"
    PRINT "Hesaplanan yaklaşık değer="; Toplamintegral
   ELSE
    PRINT "İntegral(GaussLegendre)="; Toplamintegral
   END IF

END ' GaussLegendre ana sonu
```

Tek katlı integral: Gauss-Legendre integrasyon metodu (alt program)

```
SUB GaussLegendre (a, b, Hassasiyet, Toplamintegral, iHata)
'-------------------------------------------------------------------------------
' f(x) fonksiyonun [a,b] aralığında tek katlı integralini hesaplar
' Ahmet TOPÇU, Osmangazi Üniversitesi, Eskişehir, 1994
' Kullanılan metot: Gauss-Legendre

' Veri:
'  a: integralın alt sınırı
'  b: integralın üst sınırı
'  f(x) : integrali hesaplanacak fonksiyon
'  Hassasiyet: hassasiyet

'  f(x) fonksiyonu çağıran programda DEF fnF(x)=... ile tanımlanmalıdır.

' çıktı:
'  Toplamintegral: Hesaplanan integral değeri
'  iHata=0 : integral yakınsadı
'  iHata<>0 : İntegral yakınsamadı

' çağrılan alt program: GauLeg
'-------------------------------------------------------------------------------
   MaxAralik = 20     ' değiştirilebilir(2-100 arası normal)
   MaxNokta = 1024    ' değiştirilebilir(300-3000 arası normal)
' MaxNokta alt üst sınırını kontrol et
   IF MaxNokta < 2 THEN MaxNokta = 8
   IF MaxNokta > 2 ^ 13 THEN MaxNokta = 2 ^ 13
   m = LOG(MaxNokta) / LOG(2)
   MaxNokta = 2 ^ m
   DIM x(MaxNokta), w(MaxNokta)

   Delta = (b - a) / MaxAralik
   Bi = a

   Toplamintegral = 0
   FOR iAralik = 1 TO MaxAralik
    Ai = Bi: Bi = Ai + Delta
    IF iAralik = MaxAralik THEN Bi = b

    Eskiintegral = 0
    FOR i = 1 TO m
     n = 2 ^ i

     CALL GauLeg(Ai, Bi, x(), w(), n)

     Altintegral = 0
     FOR j = 1 TO n
      Altintegral = Altintegral + w(j) * fnf(x(j))
     NEXT j

     IF ABS(Altintegral - Eskiintegral) <= Hassasiyet GOTO 1
     Eskiintegral = Altintegral
    NEXT i

    iHata = 1
    EXIT SUB

1   Toplamintegral = Toplamintegral + Altintegral
   NEXT iAralik


END SUB ' GaussLegendre sonu
```

GauLeg alt programı GaussLegendre programı tarafından çağrılır

```
SUB GauLeg (Ai, Bi, x(), w(), n)
'-------------------------------------------------------------------
' Gauss-Legendre integrasyon ordinatları ve ağırlıkları hesaplanır
' ai: integralin alt sınırı
' bi: integralin üst sınırı
' x(n): Gauss-Legendre ordinatlarının depolandığı vektör
' w(n): Gauss-Legendere ağırlıklarının depolandığı vektör
' n: Gaus-Legendre integral nokta sayısı

' çağrılan alt program: yok

' Fortran kodu http://www.haoli.org/nr/bookf/f4-5.ps den alınmıştır
'-------------------------------------------------------------------
   x1 = Ai
   x2 = Bi
   Pi = 4 * ATN(1)' Pi=3.14... sayısı
   Hassasiyet = 3E-14: ' Gauleg için hassasiyet(değiştirmeyiniz!)

   m = (n + 1) / 2
   xm = .5 * (x2 + x1)
   x1 = .5 * (x2 - x1)

   FOR i = 1 TO m
    z = COS(Pi * (i - .25) / (n + .5))
' # işareti DOUBLE anlamındadır
    z1 = 0
    WHILE ABS(z - z1) > Hassasiyet
     p1 = 1
     p2 = 0
     FOR j = 1 TO n
      p3 = p2
      p2 = p1
      p1 = ((2 * j - 1) * z * p2 - (j - 1) * p3) / j
     NEXT j
     pp = n * (z * p1 - p2) / (z * z - 1)
     z1 = z
     z = z1 - p1 / pp
    WEND

    x(i) = xm - x1 * z
    x(n + 1 - i) = xm + x1 * z
    w(i) = 2 * x1 / ((1 - z * z) * pp * pp)
    w(n + 1 - i) = w(i)
   NEXT i

END SUB ' Gauleg sonu
```

Tek katlı integral: Adapte Simpson integrasyon metodu (ana program)

```
'----------------------Ana program Adapte Simpson--------------------
' Ahmet Topçu, Eskişehir Osmangazi Üniversitesi, 2010
' f(x) fonksiyonunun [a,b] aralığıdaki tek katlı inegralini hesaplar

' Kullanılan metot: Adapte(adaptive) Simpson kuralı

' Veri:
'  a: integralın alt sınırı
'  b: integralın üst sınırı
'  f(x) : integrali hesaplanacak fonksiyon
'  Hassasiyet: hassasiyet

'  f(x) fonksiyonu ana programda DEF fnF(x)=... ile tanımlanmalıdır.

' çıktı:
'  Toplamintegral: Hesaplanan integral değeri
'  iHata=0 : integral yakınsadı
'  iHata<>0 : İntegral yakınsamadı

' çağrılan alt program: AdapteSimpson
'-----------------------------------------------------------------------

DEFINT I-N
DEFDBL A-H, O-Z

TYPE AraBilgiler
 a AS DOUBLE
 c AS DOUBLE
 b AS DOUBLE
 s AS DOUBLE
 Hassasiyet AS DOUBLE
 Kontrol AS INTEGER
END TYPE

DECLARE SUB AdapteSimpson (a, b, Hassasiyet, Toplamintegral, iHata)
DECLARE SUB SimpsonKurali (a, b, YariHassasiyet, AraBilgiVektoru AS AraBilgiler)
DECLARE SUB AralikYarila (iAralikNo, AraBilgiMatrisi() AS AraBilgiler, iAltAralikSayaci, itrasyonDevamEdiyor)

' aralığın ve f(x) fonksiyonunun tanımlanması
   Pi = 4 * ATN(1)  ' Pi=3.14... sayısı

   a = 1: b = 4: DEF fnF (x) = 13 * (x - x * x) * EXP(-3 * x / 2)
   'a = 0: b = Pi / 2: DEF fnF (x) = x ^ 3 / SQR(1 + COS(x) ^ 2)
   'a = 0: b = 2 * pi: DEF fnF (x) = LOG(1 + x) * SIN(10 * x)
   'a = 0: b = 5: DEF fnF (x) = x ^ 3 / (EXP(x) + 1)
   'a = -2: b = 2: DEF fnF (x) = EXP(-.5 * x) * SIN(8 * x ^ 2 ^ (1 / 3)) + 1
   'a = 0: b = 1: DEF fnF (x) = SQR(x)
   'a = 0: b = 5: DEF fnF (x) = x ^ x
   'a = 0: b = Pi / 2: DEF fnF (x) = TAN(x)
   'a = 1: b = 1.5: DEF fnF (x) = COS(TAN(x))

' analitik çözümü olmayan integraller:
   'a = -2: b = 2: DEF fnF (x) = EXP(-x ^ 2 / 2)
   'a = 0: b = 1: DEF fnF (x) = SIN(x) / x
   'a = -4: b = 4: DEF fnF (x) = SIN(x ^ 2)
   'a = 0: b = Pi: DEF fnF (x) = SIN(x) ^ (1 / 3)
   'a = 2: b = 50: DEF fnF (x) = 1 / LOG(x)
   'a = -40: b = 40: DEF fnF (x) = (1 + x ^ 4) ^ (1 / 3)
   'a = -40: b = 40: DEF fnF (x) = 1 / (1 + x ^ 4) ^ (1 / 3)

   CLS

   Hassasiyet = .000001 ' değiştirilebilir(1E-04 - 1E-12 arası normal)

   CALL AdapteSimpson(a, b, Hassasiyet, Toplamintegral, iHata)

   IF iHata <> 0 THEN
    PRINT "İntegral yakınsamadı(AdapteSimpson)"
    PRINT "Hesaplanan yaklaşık değer="; Toplamintegral
   ELSE
    PRINT "İntegral(AdapteSimpron)="; Toplamintegral
   END IF

END ' Adapte Simpson ana
```

Tek katlı integral:
Adapte Simpson integrasyon metodu (alt program)

```
SUB AdapteSimpson (a, b, Hassasiyet, Toplamintegral, iHata)
'------------------------------------------------------------------------
' f(x) fonksiyonunun [a,b] aralığıdaki inegralini hesaplar
' Ahmet Topçu, EskiŞehir Osmangazi Üniversitesi, 2010

' Kullanılan metot: Adaptive Simpson kuralı
'  1.Mathews, H. J., Numerical Methods for Mathematics, Science &
'    Engineering, Prentice Hall, 1992
'  2.Conte, S., D., Boor, C., Elementary Numerical Analysis, McGraw-Hill, 1980

' Veri:
'  a: integralın alt sınırı
'  b: integralın üst sınırı
'  f(x) : integrali hesaplanacak fonksiyon
'  Hassasiyet: hassasiyet

'  f(x) fonksiyonu çağıran programda DEF fnF(x)=... ile tanımlanmalıdır.

' çıktı:
'  Toplamintegral: integralin hesaplanan değeri
'  iHata=0 : integral yakınsadı
'  iHata<>0 : integral yakınsamadı

' Programın yalancı kodu yukarıda verilen birinci kaynaktan alınmıştır
' Çağrılan alt programlar: AralikYarila, SimpsonKurali
'------------------------------------------------------------------------
    MaxAralik = 20     ' değiştirilebilir(2-100 arası normal)
    MaxAltAralik = 500 ' programda öngörülen maksimum alt aralık sayısı
    DIM AraBilgiMatrisi(MaxAltAralik + 1) AS AraBilgiler
    DIM AraBilgiVektoru AS AraBilgiler

    iHata = 0
    Toplamintegral = 0

    Delta = (b - a) / MaxAralik
    bi = a
    FOR iAralik = 1 TO MaxAralik
     ai = bi
     bi = ai + Delta
     IF iAralik = MaxAralik THEN bi = b

     CALL SimpsonKurali(ai, bi, Hassasiyet, AraBilgiVektoru)
     AraBilgiMatrisi(1) = AraBilgiVektoru

     iAltAralikSayaci = 1
     itrasyonDevamEdiyor = 1  '( =1 TRUE =0 FALSE anlamında )
     WHILE itrasyonDevamEdiyor = 1
      n = iAltAralikSayaci
      FOR iAralikNo = n TO 1 STEP -1
       CALL AralikYarila(iAralikNo, AraBilgiMatrisi(), iAltAralikSayaci, itrasyonDevamEdiyor)

' İzin verilen maksimum aralık sayısına ulaşıldı mı?
       IF iAltAralikSayaci = MaxAltAralik THEN
        iHata = 1
        EXIT SUB
       END IF
      NEXT iAralikNo
     WEND

     Altintegral = 0
     FOR j = 1 TO iAltAralikSayaci
      Altintegral = Altintegral + AraBilgiMatrisi(j).s
     NEXT j

     Toplamintegral = Toplamintegral + Altintegral

    NEXT iAralik

END SUB ' AdapteSimpson sonu
```

```
SUB AralikYarila (iAralikNo, AraBilgiMatrisi() AS AraBilgiler, iAltAralikSayaci, itrasyonDevamEdiyor)
'-------------------------------------------------------------------------------------------------
' f(x) fonsiyonunun eğiminin büyük olduğu bölgelerde küçük aralıklar, eğimin küçük olduğu bölgelerde
' büyük aralıklar otomatik olarak seçilir
'-------------------------------------------------------------------------------------------------
     DIM AraBilgiVektoru0 AS AraBilgiler
     DIM AraBilgiVektoru1 AS AraBilgiler, AraBilgiVektoru2 AS AraBilgiler

     itrasyonDevamEdiyor = 0
     AraBilgiVektoru0 = AraBilgiMatrisi(iAralikNo)

     a = AraBilgiVektoru0.a
     c = AraBilgiVektoru0.c
     b = AraBilgiVektoru0.b
     Hassasiyet = AraBilgiVektoru0.Hassasiyet
     Kontrol = AraBilgiVektoru0.Kontrol

     IF Kontrol = 1 THEN EXIT SUB
     CALL SimpsonKurali(a, c, .5 * Hassasiyet, AraBilgiVektoru1)
     CALL SimpsonKurali(c, b, .5 * Hassasiyet, AraBilgiVektoru2)
     Hata = ABS(AraBilgiVektoru0.s - AraBilgiVektoru1.s - AraBilgiVektoru2.s)

     IF iAltAralikSayaci > 1 AND Hata < 10 * Hassasiyet THEN
      AraBilgiVektoru0.Kontrol = 1  ' kriter sağlandı anlamında


      AraBilgiMatrisi(iAralikNo).s = AraBilgiVektoru1.s + AraBilgiVektoru2.s

     ELSE
      iAltAralikSayaci = iAltAralikSayaci + 1
      FOR j = iAltAralikSayaci TO iAralikNo STEP -1
       AraBilgiMatrisi(j) = AraBilgiMatrisi(j - 1)
      NEXT j

      AraBilgiMatrisi(iAralikNo) = AraBilgiVektoru1
      AraBilgiMatrisi(iAralikNo + 1) = AraBilgiVektoru2

      itrasyonDevamEdiyor = 1
     END IF

END SUB ' AralikYarila sonu
```

Bu alt program AdapteSimpson alt pogramı tarafından çağrılır

```
SUB SimpsonKurali (a, b, YariHassasiyet, AraBilgiVektoru AS AraBilgiler)
'-----------------------------------------------------------------------
' f(x) fonsiyonunun [a,b] alt aralığındaki integralini hesaplar
' Kullanılan metot: adapte Simpson kuralı
'-----------------------------------------------------------------------
     c = (a + b) / 2
     s = (b - a) * (fnF(a) + 4 * fnF(c) + fnF(b)) / 6
     AraBilgiVektoru.a = a
     AraBilgiVektoru.c = c
     AraBilgiVektoru.b = b
     AraBilgiVektoru.s = s
     AraBilgiVektoru.Hassasiyet = YariHassasiyet
     AraBilgiVektoru.Kontrol = 0

END SUB ' SimpsonKurali sonu
```

Bu alt program AdapteSimpson alt pogramı tarafından çağrılır

Tek katlı integral: Recursive adapte simpson integrasyon metodu (ana program)

```
'-----------------Ana program Recursive Adapte Simpson----------------
' Ahmet Topçu, Eskişehir Osmangazi Üniversitesi, 2010
' f(x) fonksiyonunun [a,b] aralığındaki tek katlı inegralini hesaplar

' Kullanılan metot: Recursive Adapte(adaptive) Simpson

' Veri:
'  a: integralın alt sınırı
'  b: integralın üst sınırı
'  f(x) : integrali hesaplanacak fonksiyon
'  Hassasiyet: hassasiyet

'  f(x) fonksiyonu ana programda DEF fnF(x)=... ile tanımlanmalıdır.

' çıktı:
'  Toplamintegral: Hesaplanan integral değeri
'  iHata=0 : integral yakınsadı
'  iHata<>0 : integral yakınsamadı

' çağrılan alt program: RecursiveAdapteSimpson
'-------------------------------------------------------------------

DEFINT I-N
DEFDBL A-H, O-Z

DECLARE SUB RecursiveAdapteSimpson (ai, bi, Hassasiyet, iAltaralikSayisi, Simpsonintegrali, iHata)

' Aralığın ve f(x) fonksiyonunun tanımlanması
   Pi = 4 * ATN(1)  ' Pi=3.14... sayısı

   a = 1: b = 4: DEF fnf (x) = 13 * (x - x * x) * EXP(-3 * x / 2)
   'a = 0: b = Pi / 2: DEF fnf (x) = x ^ 3 / SQR(1 + COS(x) ^ 2)
   'a = 0: b = 2 * pi: DEF fnf (x) = LOG(1 + x) * SIN(10 * x)
   'a = 0: b = 5: DEF fnf (x) = x ^ 3 / (EXP(x) + 1)
   'a = -2: b = 2: DEF fnf (x) = EXP(-.5 * x) * SIN(8 * x ^ 2 ^ (1 / 3)) + 1
   'a = 0: b = 1: DEF fnf (x) = SQR(x)
   'a = 0: b = 5: DEF fnf (x) = x ^ x
   'a = 0: b = Pi / 2: DEF fnf (x) = TAN(x)
   'a = 1: b = 1.5: DEF fnf (x) = COS(TAN(x))

' analitik çözümü olmayan integraller:
   'a = -2: b = 2: DEF fnf (x) = EXP(-x ^ 2 / 2)
   'a = 0: b = 1: DEF fnf (x) = SIN(x) / x
   'a = -4: b = 4: DEF fnf (x) = SIN(x ^ 2)
   'a = 0: b = Pi: DEF fnf (x) = SIN(x) ^ (1 / 3)
   'a = 2: b = 50: DEF fnf (x) = 1 / LOG(x)
   'a = -40: b = 40: DEF fnf (x) = (1 + x ^ 4) ^ (1 / 3)
   'a = -40: b = 40: DEF fnf (x) = 1 / (1 + x ^ 4) ^ (1 / 3)

   CLS

   Hassasiyet = .000001 ' değiştirilebilir(1E-04 - 1E-12 arası normal)
   MaxAralik = 20       ' değiştirilebilir(2-100 arası normal)

   iHata = 0
   Toplamintegral = 0

   Delta = (b - a) / MaxAralik
   bi = a
   FOR iAralik = 1 TO MaxAralik
    ai = bi
    bi = ai + Delta
    IF iAralik = MaxAralik THEN bi = b

    iAltaralikSayisi = 0
    CALL RecursiveAdapteSimpson(ai, bi, Hassasiyet, iAltaralikSayisi, Simpsonintegrali, iHata)

    Toplamintegral = Toplamintegral + Simpsonintegrali

    IF iHata <> 0 THEN
     PRINT "İntegral yakınsamadı(RecursiveAdapteSimpson)"
     PRINT "Hesaplanan yaklaşık değer="; Toplamintegral
     END
    END IF

   NEXT iAralik

   PRINT "İntegral(RecursiveAdapteSimpson)="; Toplamintegral

END ' Recursive adapte Simpson ana sonu
```

Tek katlı integral: Recursive adapte simpson integrasyon metodu (alt program)

```
SUB RecursiveAdapteSimpson (ai, bi, Hassasiyet, iAltaralikSayisi, Simpsonintegrali, iHata)
'------------------------------------------------------------------------------
' f(x) fonksiyonunun [ai,bi] aralığıdaki inegralini hesaplar

' Kullanılan metot: Recursive Adaptive Simpson kuralı
'  Cheney, W., Kincaid D., Numerical Mathematics and Computing,
'  Books/Cole Publishing Co., California, 1994

' Veri:
'  a: integralın alt sınırı
'  b: integralın üst sınırı
'  f(x) : integrali hesaplanacak fonksiyon
'  Hassasiyet: hassasiyet

'  f(x) fonksiyonu çağıran programda DEF fnF(x)=... ile tanımlanmalıdır.

' çıktı:
'  Toplamintegral: Hesaplanan integral değeri
'  iHata=0 : integral yakınsadı
'  iHata<>0 : İntegral yakınsamadı

' Yalancı program kodu yukarıda verilen kaynaktan alınmıştır

' çağrılan alt programlar: yok
'------------------------------------------------------------------------------
   MaxAltaralik = 500  ' programda öngörülen maksimum alt aralık sayısı

   iAltaralikSayisi = iAltaralikSayisi + 1
   h = bi - ai
   c = .5 * (ai + bi)
   Fai = fnf(ai)
   Fbi = fnf(bi)
   Fc = fnf(c)
   d = .5 * (ai + c)
   e = .5 * (c + bi)
   Fd = fnf(d)
   Fe = fnf(e)
   OneSimpson = h / 6 * (Fai + 4 * Fc + Fbi)
   TwoSimpson = h / 12 * (Fai + 4 * Fd + 2 * Fc + 4 * Fe + Fbi)
   IF iAltaralikSayisi > MaxAltaralik THEN
    iHata = 1
    EXIT SUB
   ELSE
    IF ABS(TwoSimpson - OneSimpson) < 10 * Hassasiyet THEN
     Simpsonintegrali = TwoSimpson
    ELSE
     CALL RecursiveAdapteSimpson(ai, c, .5 * Hassasiyet, iAltaralikSayisi, ALeftSimpson, iHata)
     CALL RecursiveAdapteSimpson(c, bi, .5 * Hassasiyet, iAltaralikSayisi, RightSimpson, iHata)
     Simpsonintegrali = ALeftSimpson + RightSimpson
    END IF
   END IF

END SUB ' RecursiveAdapteSimpson sonu
```

Tek katlı integral: Tanh integrasyon kuralı (ana program)

```
'----------------------Ana Program Tanh Kuralı-----------------------
' f(x) fonksiyonunun [a,b] aralığında tek katlı integralini hesaplar
' Ahmet TOPÇU, Osmangazi Üniversitesi, Eskişehir, 1994
' kullanılan metod: Tanh kuralı

' Veri:
'  a: integralın alt sınırı
'  b: integralın üst sınırı
'  f(x) : integrali hesaplanacak fonksiyon
'  Hassasiyet: hassasiyet

'  f(x) fonksiyonu ana programda DEF fnF(x)=... ile tanımlanmalıdır.

' çıktı:
'  Toplamintegral: Hesaplanan integral değeri
'  iHata=0 : integral yakınsadı
'  iHata<>0 : İntegral yakınsamadı

' Çağrılan alt program: TanhKurali
'--------------------------------------------------------------------
DEFINT I-N
DEFDBL A-H, O-Z
DECLARE SUB TanhKurali (ai, bi, hassasiyet, Altintegral, iHata)

' Aralığın ve f(x) fonksiyonunun tanımlanması
   Pi = 4 * ATN(1)  ' Pi=3.14... sayısı

   a = 1: b = 4: DEF fnf (x) = 13 * (x - x * x) * EXP(-3 * x / 2)
   'a = 0: b = Pi / 2: DEF fnf (x) = x ^ 3 / SQR(1 + COS(x) ^ 2)
   'a = 0: b = 2 * pi: DEF fnf (x) = LOG(1 + x) * SIN(10 * x)
   'a = 0: b = 5: DEF fnf (x) = x ^ 3 / (EXP(x) + 1)
   'a = -2: b = 2: DEF fnf (x) = EXP(-.5 * x) * SIN(8 * x ^ 2 ^ (1 / 3)) + 1
   'a = 0: b = 1: DEF fnf (x) = SQR(x)
   'a = 0: b = 5: DEF fnf (x) = x ^ x
   'a = 0: b = Pi / 2: DEF fnf (x) = TAN(x)
   'a = 1: b = 1.5: DEF fnf (x) = COS(TAN(x))

' analitik çözümü olmayan integraller:
   'a = -2: b = 2: DEF fnf (x) = EXP(-x ^ 2 / 2)
   'a = 0: b = 200: DEF fnf (x) = SIN(x) / x
   'a = -4: b = 4: DEF fnf (x) = SIN(x ^ 2)
   'a = 0: b = Pi: DEF fnf (x) = SIN(x) ^ (1 / 3)
   'a = 2: b = 50: DEF fnf (x) = 1 / LOG(x)
   'a = -40: b = 40: DEF fnf (x) = (1 + x ^ 4) ^ (1 / 3)
   'a = -40: b = 40: DEF fnf (x) = 1 / (1 + x ^ 4) ^ (1 / 3)

   CLS

   hassasiyet = .000001' değiştirilebilir, (1E-4 - 1E-12 arası normal)
   MaxAralik = 20 ' değiştirilebilir, (2-100 arası normal)

   Delta = (b - a) / MaxAralik
   bi = a

   Toplamintegral = 0
   FOR iAralikNo = 1 TO MaxAralik

    ai = bi: bi = ai + Delta
    IF iAralikNo = MaxAralik THEN bi = b
    CALL TanhKurali(ai, bi, hassasiyet, Altintegral, iHata)
    Toplamintegral = Toplamintegral + Altintegral

   NEXT iAralikNo

   IF iHata <> 0 THEN
    PRINT "İntegral yakınsamadı(TanhKurali)"
    PRINT "Hesaplanan yaklaşık değer="; Toplamintegral
   ELSE
    PRINT "İntegral(TanhKurali)="; Toplamintegral
   END IF

END ' Tanh ana sonu
```

Tek katlı integral:
Tanh integrasyon kuralı
(alt program)

```
SUB TanhKurali (ai, bi, hassasiyet, Altintegral, iHata)
'---------------------------------------------------------------------
' f(x) fonksiyonunun [a,b] aralığında tek katlı integralini hesaplar
' Ahmet TOPÇU, Osmangazi Üniversitesi, Eskişehir, 1994
' kullanılan metod: Tanh kuralı

' Veri:
'  a: integralın alt sınırı
'  b: integralın üst sınırı
'  f(x) : integrali hesaplanacak fonksiyon
'  Hassasiyet: hassasiyet

'  f(x) fonksiyonu çağıran programda DEF fnF(x)=... ile tanımlanmalıdır.

' çıktı:
'  Toplamintegral: Hesaplanan integral değeri
'  iHata=0 : integral yakınsadı
'  iHata<>0 : İntegral yakınsamadı

' Çağrılan alt program: yok
'---------------------------------------------------------------------

    iHata = 0
    Pi = 4 * ATN(1)   ' Pi=3.14... sayısı
    NoktaSayisi = 100 'değiştirilebilir(20-1000 arası normal)

    Sabit = .5 / NoktaSayisi
    h = Pi * SQR(Sabit) - Sabit
    Delta = .5 * (bi - ai)
    x = .5 * (ai + bi)
    Altintegral = fnf(x)
    Eskiintegral = Altintegral

    FOR i = 1 TO NoktaSayisi
     hi = h * i
     Sinh = .5 * (EXP(hi) - EXP(-hi))
     Cosh = .5 * (EXP(hi) + EXP(-hi))
     Tanh = Sinh / Cosh
     y = Delta * Tanh

     Altintegral = Altintegral + (fnf(x + y) + fnf(x - y)) / Cosh ^ 2

     IF ABS(Eskiintegral - Altintegral) < hassasiyet THEN
      Altintegral = Delta * h * Altintegral
      EXIT SUB
      END IF
     Eskiintegral = Altintegral
    NEXT i
    iHata = 1  ' yakınsamadı

END SUB  ' TanhKurali sonu
```

İki katlı integral:
Yamuk kuralı ile integrasyon
(ana program)

```
'----------------------Ana program Yamuk2-----------------------
' Ahmet TOPÇU, Osmangazi Üniversitesi, Eskişehir, 1998
' F(x,y) fonksiyonunun [Xa,Xb] ve [Ya(x),Yb(x)]  aralığında iki katlı
' integralı yamuk kuralı ile hesaplanır.

'Veri:
' Xa, Xb : x ekseni boyunca integralin alt ve üst sınırı
' Ya(x), Yb(x): y ekseni boyunca alt ve üst sınırın fonksiyonu
' F(x,y) : integralı alınacak fonksiyon
' Hassasiyet: Hassasiyet

' Aralıklar ve f(x,y,z) ana programda aşağıdaki gibi tanımlanmalıdır:
' Xa=...           (sabit sayı)
' Xb=...           (sabit sayı)
' DEF FnYa(x)=...    (sabit sayı veya fonksiyon)
' DEF FnYb(x)=...    (sabit sayı veya fonksiyon)
' Def fnF(x,y)=...   (integralı hesaplanacak fonksiyon)

'Çıktı:
' Dintegral: hesaplanan integral
' iHata=0 : integral yakınsadı
' iHata<>0: integral yakınsamadı

'Çağrılan alt program: Yamuk2
'--------------------------------------------------------------------

DEFINT I-N
DEFDBL A-H, O-Z
DECLARE SUB Yamuk2 (Xa, Xb, Hassasiyet, Dintegral, iHata)
DECLARE SUB YamukKurali (Xa, Xb, iAralikSayisi, Dintegral)
DECLARE SUB Yintegral (x, iAralikSayisi, Yint)

' aralıkların ve F(x,y) fonksiyonunun tanmlanması

    Pi = 4 * ATN(1)   ' Pi=3.14... sayısı

    Xa = -2: Xb = 2
    DEF FnYa (x) = -1: DEF FnYb (x) = 1
    DEF FnF (x, y) = EXP(-x * x * y * y)

    'Xa = 0: Xb = Pi / 2
    'DEF FnYa (x) = 0: DEF FnYb (x) = 1
    'DEF FnF (x, y) = (1 - y * y) * COS(x) ^ 2

    'Xa = 0: Xb = 2
    'DEF FnYa (x) = 0
    'DEF FnYb (x) = SQR((4 - x * x) / 2)
    'DEF FnF (x, y) = 4 - x * x - 2 * y * y

    'Xa = 0: Xb = 2
    'DEF FnYa (x) = 1 - x
    'DEF FnYb (x) = EXP(-x)
    'DEF FnF (x, y) = SQR(x + y)

    'Xa = 0: Xb = Pi / 2
    'DEF FnYa (x) = TAN(x / 2)
    'DEF FnYb (x) = SIN(x)
    'DEF FnF (x, y) = x + y

    'Xa = 0: Xb = 2
    'DEF FnYa (x) = -x ^ x: DEF FnYb (x) = x ^ x
    'DEF FnF (x, y) = x * x + y * y

    'Xa = -4: Xb = 4
    'DEF FnYa (x) = -SQR(ABS(16 - x * x))
    'DEF FnYb (x) = SQR(ABS(16 - x * x))
    'DEF FnF (x, y) = SQR(ABS(16 - x * x - y * y))

    'Xa = 0: Xb = 2
    'DEF FnYa (x) = x * x: DEF FnYb (x) = SQR(8 * x)
    'DEF FnF (x, y) = EXP(-x * SQR(y))

    CLS

    Hassasiyet = .01' Değiştirilebilir, 1E-02 -1E-4 arası normal değer
    CALL Yamuk2(Xa, Xb, Hassasiyet, Dintegral, iHata)

    IF iHata <> 0 THEN
     PRINT "İntegral yakınsamadı(Yamuk2)"
     PRINT "Hesaplanan yaklaşık değer="; Dintegral
    ELSE
     PRINT "İntegral="; Dintegral
    END IF

END ' Yamuk2 ana sonu
```

İki katlı integral:
Yamuk kuralı ile integrasyon
(alt program)

```
SUB Yamuk2 (Xa, Xb, Hassasiyet, Dintegral, iHata)
'------------------------------------------------------------------------
' F(x,y) fonksiyonunun [Xa,Xb] ve [Ya(x),Yb(x)]  aralığında iki katlı
' integralı yamuk kuralı ile hesaplanır.

'Veri:
' Xa, Xb : x ekseni boyunca integralin alt ve üst sınırı
' Ya(x), Yb(x): y ekseni boyunca alt ve üst sınırın fonksiyonu
' F(x,y) : integralı alınacak fonksiyon
' Hassasiyet: Hassasiyet

' Aralıklar ve f(x,y,z) çağıran programda aşağıdaki gibi tanımlanmalıdır:
' Xa=...            (sabit sayı)
' Xb=...            (sabit sayı)
' DEF FnYa(x)=...    (sabit sayı veya fonksiyon)
' DEF FnYb(x)=...    (sabit sayı veya fonksiyon)
' Def fnF(x,y)=...   (integralı hesaplanacak fonksiyon)

'Çıktı:
' Dintegral: hesaplanan integral
' iHata=0 : integral yakınsadı
' iHata<>0: integral yakınsamadı

'Çağrılan alt program: Yintegral
'------------------------------------------------------------------------
   iHata = 0
   MaxAralikSayisi = 2000 ' değiştirilebilir(100-3000 arası normal)
   Eskiintegral = 0

   FOR iAralikSayisi = 100 TO MaxAralikSayisi STEP 100
    Dintegral = 0
' yamuk kurali ile integral hesabi
    DeltaX = (Xb - Xa) / iAralikSayisi

    x = Xa
    CALL Yintegral(x, iAralikSayisi, Yint)
    Dintegral = Yint

    x = Xb
    CALL Yintegral(x, iAralikSayisi, Yint)
    Dintegral = .5 * (Dintegral + Yint)

    x = Xa
    FOR k = 1 TO iAralikSayisi - 1
     x = x + DeltaX
     CALL Yintegral(x, iAralikSayisi, Yint)
     Dintegral = Dintegral + Yint
    NEXT k

    Dintegral = DeltaX * Dintegral

    hata = ABS(Dintegral - Eskiintegral)
    IF hata <= Hassasiyet THEN EXIT SUB
    Eskiintegral = Dintegral
   NEXT iAralikSayisi

   iHata = 1 ' Yakınsamadı

END SUB ' Yamuk2 sonu

SUB Yintegral (x, iAralikSayisi, Yint)
'-------------------------------------
' X=sabit icin y boyunca integral
   Ya = FnYa(x)
   Yb = FnYb(x)

   deltay = (Yb - Ya) / iAralikSayisi

   Yint = .5 * (FnF(x, Ya) + FnF(x, Yb))
   y = Ya
   FOR j = 1 TO iAralikSayisi - 1
    y = y + deltay
    Yint = Yint + FnF(x, y)
   NEXT j
   Yint = deltay * Yint

END SUB  'Yintegral sonu
```

İki katlı integral:
Simpson metodu ile integrasyon
(ana program)

```
'--------------------Ana program Simpson2-----------------------------
' Ahmet TOPÇU, Osmangazi Üniversitesi, Eskişehir, 1998
' F(x,y) fonksiyonunun [Xa,Xb] ve [Ya(x),Yb(x)]  aralığında iki katlı
' integralı Simpson metodu ile hesaplanır.

'Veri:
' Xa, Xb : x ekseni boyunca integralin alt ve üst sınırı
' Ya(x), Yb(x): y ekseni boyunca alt ve üst sınırın fonksiyonu
' F(x,y) : integralı alınacak fonksiyon
' Hassasiyet: Hassasiyet

' Aralıklar ve f(x,y,z) ana programda aşağıdaki gibi tanımlanmalıdır:
' Xa=...           (sabit sayı)
' Xb=...           (sabit sayı)
' DEF FnYa(x)=...    (sabit sayı veya fonksiyon)
' DEF FnYb(x)=...    (sabit sayı veya fonksiyon)
' Def fnF(x,y)=...   (integralı hesaplanacak fonksiyon)

'Çıktı:
' Dintegral: hesaplanan integral
' iHata=0 : integral yakınsadı
' iHata<>0: integral yakınsamadı

'Çağrılan alt program: Simpson2
'-----------------------------------------------------------------------

DEFINT I-N
DEFDBL A-H, O-Z
DECLARE SUB Simpson2 (Xa, Xb, Hassasiyet, Dintegral, iHata)

' aralıkların ve F(x,y) fonksiyonunun tanmlanması

    Pi = 4 * ATN(1)   ' Pi=3.14... sayısı

    Xa = -2: Xb = 2
    DEF FnYa (x) = -1: DEF FnYb (x) = 1
    DEF FnF (x, y) = EXP(-x * x * y * y)

    'Xa = 0: Xb = Pi / 2
    'DEF FnYa (x) = 0: DEF FnYb (x) = 1
    'DEF FnF (x, y) = (1 - y * y) * COS(x) ^ 2

    'Xa = 0: Xb = 2
    'DEF FnYa (x) = 0
    'DEF FnYb (x) = SQR((4 - x * x) / 2)
    'DEF FnF (x, y) = 4 - x * x - 2 * y * y

    'Xa = 0: Xb = 2
    'DEF FnYa (x) = 1 - x
    'DEF FnYb (x) = EXP(-x)
    'DEF FnF (x, y) = SQR(x + y)

    'Xa = 0: Xb = Pi / 2
    'DEF FnYa (x) = TAN(x / 2)
    'DEF FnYb (x) = SIN(x)
    'DEF FnF (x, y) = x + y

    'Xa = 0: Xb = 2
    'DEF FnYa (x) = -x ^ x: DEF FnYb (x) = x ^ x
    'DEF FnF (x, y) = x * x + y * y

    'Xa = -4: Xb = 4
    'DEF FnYa (x) = -SQR(ABS(16 - x * x))
    'DEF FnYb (x) = SQR(ABS(16 - x * x))
    'DEF FnF (x, y) = SQR(ABS(16 - x * x - y * y))

    'Xa = 0: Xb = 2
    'DEF FnYa (x) = x * x: DEF FnYb (x) = SQR(8 * x)
    'DEF FnF (x, y) = EXP(-x * SQR(y))

    CLS
    Hassasiyet = .01  'değiştirilebilir, 1E-2 - 1E-4 arası normal değer

    CALL Simpson2(Xa, Xb, Hassasiyet, Dintegral, iHata)

    IF iHata <> 0 THEN
     PRINT "İntegral yakınsamadı(Simpson2)"
     PRINT "Hesaplanan yaklaşık değer:"; Dintegral
    ELSE
     PRINT "İntegral(Simpson2)="; Dintegral
    END IF

END  ' Simpson2 ana sonu
```

İki katlı integral:
Simpson metodu ile integrasyon
(alt program)

```
SUB Simpson2 (Xa, Xb, Hassasiyet, Dintegral, iHata)
'------------------------------------------------------------------------
' F(x,y) fonksiyonunun [Xa,Xb] ve [Ya(x),Yb(x)]  aralığında iki katlı
' integralı Simpson metodu ile hesaplanır.

'Veri:
' Xa, Xb : x ekseni boyunca integralin alt ve üst sınırı
' Ya(x), Yb(x): y ekseni boyunca alt ve üst sınırın fonksiyonu
' F(x,y) : integralı alınacak fonksiyon
' Hassasiyet: Hassasiyet

' Aralıklar ve f(x,y,z) çağıran programda aşağıdaki gibi tanımlanmalıdır:
' Xa=...           (sabit sayı)
' Xb=...           (sabit sayı)
' DEF FnYa(x)=...    (sabit sayı veya fonksiyon)
' DEF FnYb(x)=...    (sabit sayı veya fonksiyon)
' Def fnF(x,y)=...   (integralı hesaplanacak fonksiyon)

'Çıktı:
' Dintegral: hesaplanan integral
' iHata=0 : integral yakınsadı
' iHata<>0: integral yakınsamadı

'Çağrılan alt program: Yok
'------------------------------------------------------------------------
    Maxiterasyon = 20 'değiştirilebilir (10-100 arası normal)
    iHata = 0
    Dintegral = 0
    Eskiintegral = 0

    FOR iterasyonNo = 1 TO Maxiterasyon
     iAralikSayisiX = 10 * iterasyonNo
     iAralikSayisiY = 10 * iterasyonNo
     iTamSayi = iAralikSayisiX / 2
     iAralikSayisiX = 2 * iTamSayi
     iTamSayi = iAralikSayisiY / 2
     iAralikSayisiY = 2 * iTamSayi
     IF Xb < Xa THEN
      iHata = 1
      EXIT SUB
     END IF
     Hx = (Xb - Xa) / iAralikSayisiX
     Toplam1 = 0
     Toplam2 = 0
     Toplam4 = 0
     NoktaSayisiX = iAralikSayisiX + 1
     NoktaSayisiY = iAralikSayisiY - 1
' y boyunca integral
     FOR iNoktaX = 1 TO NoktaSayisiX
      x = Xa + (iNoktaX - 1) * Hx
      IF iNoktaX = NoktaSayisiX THEN x = Xb
      Ya = FnYa(x)
      Yb = FnYb(x)
      Hy = (Yb - Ya) / iAralikSayisiY
      F1 = FnF(x, Ya)
      F1 = F1 + FnF(x, Yb)
      F2 = 0
      F4 = 0
      FOR iNoktaY = 1 TO NoktaSayisiY
       y = Ya + iNoktaY * Hy
       Fxy = FnF(x, y)
       iTamSayi = iNoktaY / 2
       IF iNoktaY = 2 * iTamSayi THEN
        F2 = F2 + Fxy
       ELSE
        F4 = F4 + Fxy
       END IF
      NEXT iNoktaY
' x boyunca integral
      ToplamY = Hy / 3 * (F1 + 2 * F2 + 4 * F4)
      IF iNoktaX = 1 OR iNoktaX = NoktaSayisiX THEN
       Toplam1 = Toplam1 + ToplamY
      ELSE
       iTamSayi = iNoktaX / 2
       IF 2 * iTamSayi = iNoktaX THEN
        Toplam4 = Toplam4 + ToplamY
       ELSE
        Toplam2 = Toplam2 + ToplamY
       END IF
      END IF
     NEXT iNoktaX
     Dintegral = Hx / 3 * (Toplam1 + 2 * Toplam2 + 4 * Toplam4)
      Hata = ABS(Dintegral - Eskiintegral)
     IF Hata <= Hassasiyet THEN EXIT SUB
     Eskiintegral = Dintegral
    NEXT iterasyonNo
    iHata = 2  'yakınsamıyor

END SUB   ' Simpson2 sonu
```

İki katlı integral:
Romberg metodu ile integrasyon
(ana program)

```
'----------------------Ana program Romberg2-----------------------
' Ahmet TOPÇU, Osmangazi Üniversitesi, Eskişehir, 1998
' F(x,y) fonksiyonunun [Xa,Xb] ve [Ya(x),Yb(x)]  aralığında iki katlı
' integralı Romberg metodu ile hesaplanır.

'Veri:
' Xa, Xb : x ekseni boyunca integralin alt ve üst sınırı
' Ya(x), Yb(x): y ekseni boyunca alt ve üst sınırın fonksiyonu
' F(x,y) : integralı alınacak fonksiyon
' Hassasiyet: Hassasiyet

' Aralıklar ve f(x,y,z) ana programda aşağıdaki gibi tanımlanmalıdır:
' Xa=...            (sabit sayı)
' Xb=...            (sabit sayı)
' DEF FnYa(x)=...    (sabit sayı veya fonksiyon)
' DEF FnYb(x)=...    (sabit sayı veya fonksiyon)
' Def fnF(x,y)=...   (integralı hesaplanacak fonksiyon)

'Çıktı:
' Dintegral: hesaplanan integral
' iHata=0 : integral yakınsadı
' iHata<>0: integral yakınsamadı

'Çağrılan alt program: Romberg2
'-----------------------------------------------------------------------

DEFINT I-N
DEFDBL A-H, O-Z
DECLARE SUB Romberg2 (Xa, Xb, Hassasiyet, Dintegral, iHata)
DECLARE SUB YamukKurali (Xa, Xb, iAralikSayisi, Dintegral)
DECLARE SUB Yintegral (x, iAralikSayisi, Yint)

' aralıkların ve F(x,y) fonksiyonunun tanmlanması

    Pi = 4 * ATN(1)   ' Pi=3.14... sayısı

    'Xa = -2: Xb = 2
    'DEF FnYa (x) = -1: DEF FnYb (x) = 1
    'DEF FnF (x, y) = EXP(-x * x * y * y)

    'Xa = 0: Xb = Pi / 2
    'DEF FnYa (x) = 0: DEF FnYb (x) = 1
    'DEF FnF (x, y) = (1 - y * y) * COS(x) ^ 2

    'Xa = 0: Xb = 2
    'DEF FnYa (x) = 0
    'DEF FnYb (x) = SQR((4 - x * x) / 2)
    'DEF FnF (x, y) = 4 - x * x - 2 * y * y

    'Xa = 0: Xb = 2
    'DEF FnYa (x) = 1 - x
    'DEF FnYb (x) = EXP(-x)
    'DEF FnF (x, y) = SQR(x + y)

    'Xa = 0: Xb = Pi / 2
    'DEF FnYa (x) = TAN(x / 2)
    'DEF FnYb (x) = SIN(x)
    'DEF FnF (x, y) = x + y

    'Xa = 0: Xb = 2
    'DEF FnYa (x) = -x ^ x: DEF FnYb (x) = x ^ x
    'DEF FnF (x, y) = x * x + y * y

    'Xa = -4: Xb = 4
    'DEF FnYa (x) = -SQR(ABS(16 - x * x))
    'DEF FnYb (x) = SQR(ABS(16 - x * x))
    'DEF FnF (x, y) = SQR(ABS(16 - x * x - y * y))

    Xa = 0: Xb = 2
    DEF FnYa (x) = x * x: DEF FnYb (x) = SQR(8 * x)
    DEF FnF (x, y) = EXP(-x * SQR(y))

    CLS

    Hassasiyet = .01  '1E-2 - 1E-4 arası normal değer
    CALL Romberg2(Xa, Xb, Hassasiyet, Dintegral, iHata)

    IF iHata <> 0 THEN
     PRINT "İntegral yakınsamadı(Romberg2)"
     PRINT "Hesaplanan yaklaşık değer="; Dintegral
    ELSE
     PRINT "İntegral(Romberg2)="; Dintegral
    END IF

END ' Romberg2 ana sonu
```

İki katlı integral:
Romberg metodu ile integrasyon
(alt program)

```
SUB Romberg2 (Xa, Xb, Hassasiyet, Dintegral, iHata)
'-------------------------------------------------------------------------
' Ahmet TOPÇU, Osmangazi Üniversitesi, Eskişehir, 1998
' F(x,y) fonksiyonunun [Xa,Xb] ve [Ya(x),Yb(x)]  aralığında iki katlı
' integralı Romberg metodu ile hesaplanır.

'Veri:
' Xa, Xb : x ekseni boyunca integralin alt ve üst sınırı
' Ya(x), Yb(x): y ekseni boyunca alt ve üst sınırın fonksiyonu
' F(x,y) : integralı alınacak fonksiyon
' Hassasiyet: Hassasiyet

' Aralıklar ve f(x,y,z) çağıran programda aşağıdaki gibi tanımlanmalıdır:
' Xa=...            (sabit sayı)
' Xb=...            (sabit sayı)
' DEF FnYa(x)=...    (sabit sayı veya fonksiyon)
' DEF FnYb(x)=...    (sabit sayı veya fonksiyon)
' Def fnF(x,y)=...   (integralı hesaplanacak fonksiyon)

'Çıktı:
' Dintegral: hesaplanan integral
' iHata=0 : integral yakınsadı
' iHata<>0: integral yakınsamadı

'Çağrılan alt program: YamukKurali
'-------------------------------------------------------------------------

    iAralikSayisi = 10 ' değiştirilebilir(10-20 arası normal)
    MaxRomberg = 10 ' değiştirilebilir(5-10 arası normal)
    DIM t(MaxRomberg + 1, MaxRomberg + 1)

    iHata = 0
    IF Xb < Xa THEN
     iHata = 1   ' aralık hatalı
     EXIT SUB
    END IF

    CALL YamukKurali(Xa, Xb, iAralikSayisi, Dintegral)
    t(1, 1) = Dintegral

    FOR i = 1 TO MaxRomberg
     iAralikSayisi = 2 * iAralikSayisi
     CALL YamukKurali(Xa, Xb, iAralikSayisi, Dintegral)
     i1 = i + 1
     t(i1, 1) = Dintegral

' Romberg tablosu
     FOR j = 1 TO i
      t(i1, j + 1) = t(i1, j) + (t(i1, j) - t(i, j)) / (4 ^ j - 1)
     NEXT j
     Dintegral = t(i1, i1)
     Hata = ABS(t(i1, i1) - t(i, i))
     IF Hata <= Hassasiyet THEN EXIT SUB
    NEXT i

    iHata = 1 ' yakınsamadı

END SUB ' Romberg2 sonu
```

```
SUB YamukKurali (Xa, Xb, iAralikSayisi, Dintegral)
'-------------------------------------------------------
' sabit x için y boyunca yamuk kurali ileintegral hesabı
' çağrılan alt program: Yintegral
'-------------------------------------------------------
  Dintegral = 0
' yamuk kurali ile integral hesabi
  DeltaX = (Xb - Xa) / iAralikSayisi

  x = Xa
  CALL Yintegral(x, iAralikSayisi, Yint)
  Dintegral = Yint

  x = Xb
  CALL Yintegral(x, iAralikSayisi, Yint)
  Dintegral = .5 * (Dintegral + Yint)

  x = Xa
  FOR k = 1 TO iAralikSayisi - 1
   x = x + DeltaX
   CALL Yintegral(x, iAralikSayisi, Yint)
   Dintegral = Dintegral + Yint
  NEXT k

  Dintegral = DeltaX * Dintegral

END SUB ' YamukKuralı sonu



SUB Yintegral (x, iAralikSayisi, Yint)
'-------------------------------------
' X=sabit icin y boyunca integral
  Ya = FnYa(x)
  Yb = FnYb(x)

  deltay = (Yb - Ya) / iAralikSayisi

  Yint = .5 * (FnF(x, Ya) + FnF(x, Yb))
  y = Ya
  FOR j = 1 TO iAralikSayisi - 1
   y = y + deltay
   Yint = Yint + FnF(x, y)
  NEXT j
  Yint = deltay * Yint

END SUB  ' Yintegral sonu
```

İki katlı integral: Gauss-Legendre metodu ile integrasyon (ana program)

```
'----------------------Ana program GaussLegendre2----------------------
' Ahmet TOPÇU, Osmangazi Üniversitesi, Eskişehir, 1998
' F(x,y) fonksiyonunun [Xa,Xb] ve [Ya(x),Yb(x)]  aralığında iki katlı
' integralı Gauss-Legendre metodu ile hesaplanır.

'Veri:
' Xa, Xb : x ekseni boyunca integralin alt ve üst sınırı
' Ya(x), Yb(x): y ekseni boyunca alt ve üst sınırın fonksiyonu
' F(x,y) : integralı alınacak fonksiyon
' Hassasiyet: Hassasiyet

' Aralıklar ve f(x,y,z) ana programda aşağıdaki gibi tanımlanmalıdır:
' Xa=...           (sabit sayı)
' Xb=...           (sabit sayı)
' DEF FnYa(x)=...    (sabit sayı veya fonksiyon)
' DEF FnYb(x)=...    (sabit sayı veya fonksiyon)
' Def fnF(x,y)=...   (integralı hesaplanacak fonksiyon)

'Çıktı:
' Dintegral: hesaplanan integral
' iHata=0 : integral yakınsadı
' iHata<>0: integral yakınsamadı

'Çağrılan alt program: GaussLegendre2
'------------------------------------------------------------------------

DEFINT I-N
DEFDBL A-H, O-Z
DECLARE SUB GaussLegendre2 (Xa, Xb, Hassasiyet, Dintegral, iHata)
DECLARE SUB GauLeg (ai, bi, x(), w(), n)

' aralıkların ve F(x,y) fonksiyonunun tanmlanması

    Pi = 4 * ATN(1)   ' Pi=3.14... sayısı

    Xa = -2: Xb = 2
    DEF FnYa (x) = -1: DEF FnYb (x) = 1
    DEF FnF (x, y) = EXP(-x * x * y * y)

    'Xa = 0: Xb = Pi / 2
    'DEF FnYa (x) = 0: DEF FnYb (x) = 1
    'DEF FnF (x, y) = (1 - y * y) * COS(x) ^ 2

    'Xa = 0: Xb = 2
    'DEF FnYa (x) = 0
    'DEF FnYb (x) = SQR((4 - x * x) / 2)
    'DEF FnF (x, y) = 4 - x * x - 2 * y * y

    'Xa = 0: Xb = 2
    'DEF FnYa (x) = 1 - x
    'DEF FnYb (x) = EXP(-x)
    'DEF FnF (x, y) = SQR(x + y)

    'Xa = 0: Xb = Pi / 2
    'DEF FnYa (x) = TAN(x / 2)
    'DEF FnYb (x) = SIN(x)
    'DEF FnF (x, y) = x + y

    'Xa = 0: Xb = 2
    'DEF FnYa (x) = -x ^ x: DEF FnYb (x) = x ^ x
    'DEF FnF (x, y) = x * x + y * y

    'Xa = -4: Xb = 4
    'DEF FnYa (x) = -SQR(ABS(16 - x * x))
    'DEF FnYb (x) = SQR(ABS(16 - x * x))
    'DEF FnF (x, y) = SQR(ABS(16 - x * x - y * y))

    'Xa = 0: Xb = 2
    'DEF FnYa (x) = x * x: DEF FnYb (x) = SQR(8 * x)
    'DEF FnF (x, y) = EXP(-x * SQR(y))

    CLS

    Hassasiyet = .01 'Değiştirilebilir,1e-2 - 1E-4 arası normal değer
    CALL GaussLegendre2(Xa, Xb, Hassasiyet, Dintegral, iHata)
    IF iHata <> 0 THEN
     PRINT "Yakınsamadı, yaklaşık değer(GaussLegendre2)="; Dintegral
    ELSE
     PRINT "İntegral(GaussLegendre2)="; Dintegral
    END IF

END  ' GaussLegendre2 ana sonu
```

İki katlı integral: Gauss-Legendre metodu ile integrasyon (alt program)

```
SUB GaussLegendre2 (Xa, Xb, Hassasiyet, Dintegral, iHata)
'------------------------------------------------------------------------
' Ahmet TOPÇU, Osmangazi Üniversitesi, Eskişehir, 1998
' F(x,y) fonksiyonunun [Xa,Xb] ve [Ya(x),Yb(x)]  aralığında iki katlı
' integralı Gauss-Legendre metodu ile hesaplanır.

'Veri:
' Xa, Xb : x ekseni boyunca integralin alt ve üst sınırı
' Ya(x), Yb(x): y ekseni boyunca alt ve üst sınırın fonksiyonu
' F(x,y) : integralı alınacak fonksiyon
' Hassasiyet: Hassasiyet

' Aralıklar ve f(x,y,z) çağıran programda aŞağıdaki gibi tanımlanmalıdır:
' Xa=...            (sabit sayı)
' Xb=...            (sabit sayı)
' DEF FnYa(x)=...    (sabit sayı veya fonksiyon)
' DEF FnYb(x)=...    (sabit sayı veya fonksiyon)
' Def fnF(x,y)=...   (integralı hesaplanacak fonksiyon)

'Çıktı:
' Dintegral: hesaplanan integral
' iHata=0 : integral yakınsadı
' iHata<>0: integral yakınsamadı

'Çağrılan alt program: GauLeg
'------------------------------------------------------------------------

    MaxNokta = 1024  ' öngörülen maksimum Gauss-Legendre nokta sayısı
    DIM x(MaxNokta), w(MaxNokta)

    iMax = LOG(MaxNokta) / LOG(2)

    iHata = 0
    Eskiintegral = 0

    FOR i = 4 TO iMax
     NoktaSayisi = 2 ^ i
' Gauss-Lgendre koordinatları:
     CALL GauLeg(-1, 1, x(), w(), NoktaSayisi)

     Hx = .5 * (Xb - Xa)
     HxOrta = .5 * (Xb + Xa)

     Dintegral = 0

     FOR NoktaX = 1 TO NoktaSayisi
      x = Hx * x(NoktaX) + HxOrta
      Xint = 0
      Ya = FnYa(x)
      Yb = FnYb(x)
      Hy = .5 * (Yb - Ya)
      HyOrta = .5 * (Yb + Ya)

      FOR NoktaY = 1 TO NoktaSayisi
       y = Hy * x(NoktaY) + HyOrta
       Fxy = FnF(x, y)
       Xint = Xint + w(NoktaY) * Fxy
      NEXT NoktaY

      Dintegral = Dintegral + w(NoktaX) * Hy * Xint
     NEXT NoktaX

     Dintegral = Dintegral * Hx

     hata = ABS(Dintegral - Eskiintegral)
     IF hata <= Hassasiyet THEN EXIT SUB
     Eskiintegral = Dintegral

    NEXT i

    iHata = 1' Yakınsamadı

END SUB ' GaussLegendre2 sonu
```

GauLeg: Bu alt program GaussLegendre2 tarafından çağrılır

```
SUB GauLeg (ai, bi, x(), w(), n)
'------------------------------------------------------------------------
' Gauss-Legendre integrasyon ordinatları ve ağırlıkları hesaplanır
' ai: integralin alt sınırı
' bi: integralin üst sınırı
' x(n): Gauss-Legendre ordinatlarının depolandığı vektör
' w(n): Gauss-Legendere ağırlıklarının depolandığı vektör
' n: Gaus-Legendre integral nokta sayısı

' çağrılan alt program: yok

' Fortran kodu http://www.haoli.org/nr/bookf/f4-5.ps den alınmıştır
'------------------------------------------------------------------------
   x1 = ai
   x2 = bi
   Pi = 4 * ATN(1)' Pi=3.14...  sayısı
   Hassasiyet = 3E-14: ' Gauleg için hassasiyet(değiştirmeyiniz!)

   m = (n + 1) / 2
   xm = .5 * (x2 + x1)
   x1 = .5 * (x2 - x1)

   FOR i = 1 TO m
    z = COS(Pi * (i - .25) / (n + .5))
' # işareti DOUBLE anlamındadır
    z1 = 0
    WHILE ABS(z - z1) > Hassasiyet
     p1 = 1
     p2 = 0
     FOR j = 1 TO n
      p3 = p2
      p2 = p1
      p1 = ((2 * j - 1) * z * p2 - (j - 1) * p3) / j
     NEXT j
     pp = n * (z * p1 - p2) / (z * z - 1)
     z1 = z
     z = z1 - p1 / pp
    WEND

    x(i) = xm - x1 * z
    x(n + 1 - i) = xm + x1 * z
    w(i) = 2 * x1 / ((1 - z * z) * pp * pp)
    w(n + 1 - i) = w(i)
   NEXT i

END SUB ' Gauleg sonu
```

Üç katlı integral: GaussLegendre metodu ile integrasyon (ana program)

```
'----------------------Ana program GaussLegendre 3----------------------
' Ahmet TOPÇU, Osmangazi Üniversitesi, Eskişehir, 2010
' F(x,y,z) fonksiyonunun [Xa,Xb], [Ya(x),Yb(x)] ve [Za(x,y),Zb(x,y)]
' aralığında üç katlı integralini  Gauss-Legendre metodu ile hesaplar.

'Veri:
' Xa, Xb : x ekseni boyunca integralin alt ve üst sınırı
' Ya(x), Yb(x): y ekseni boyunca alt ve üst sınırın fonksiyonu
' Za(x,y), Zb(x,y): z ekseni boyunca alt ve üst sınırın fonksiyonu
' F(x,y,z) : integralı hesaplanacak fonksiyon
' Hassasiyet: Hassasiyet

' Aralıklar ve f(x,y,z) ana programda Xaşağıdaki gibi tanımlanmalıdır:
' Xa=...            (sabit sayı)
' Xb=...            (sabit sayı)
' DEF FnYa(x)=...    (sabit sayı veya fonksiyon)
' DEF FnYb(x)=...    (sabit sayı veya fonksiyon)
' DEF FnZa(x,y)=...  (sabit sayı veya fonksiyon)
' DEF FnZb(x,y)=...  (sabit sayı veya fonksiyon)
' Def fnF(x,y,z)=... (integralı hesaplanacak fonksiyon)

'Çıktı:
' Dintegral: hesaplanan integral
' iHata=0 : integral yakınsadı
' iHata<>0: integral yakınsamadı

'Çağrılan alt program: GaussLegendre3
'-------------------------------------------------------------------------
DEFINT I-N
DEFDBL A-H, O-Z
DECLARE SUB GaussLegendre3 (Xa, Xb, Hassasiyet, Dintegral, iHata)
DECLARE SUB GauLeg (Ai, Bi, x(), w(), n)
     Pi = 4 * ATN(1)  ' Pi=3.14... sayısı

     Xa = 0: Xb = 1
     DEF fnYa (x) = 0: DEF fnYb (x) = 2
     DEF fnZa (x, y) = 0: DEF fnZb (x, y) = 3
     DEF fnF (x, y, z) = x * x * y * y * z * z

     'Xa = 0: Xb = 2
     'DEF fnYa (x) = 0: DEF fnYb (x) = 3
     'DEF fnZa (x, y) = 0: DEF fnZb (x, y) = SQR(9 - y * y)
     'DEF fnF (x, y, z) = z * z

     'Xa = 0: Xb = 4
     'DEF fnYa (x) = 0: DEF fnYb (x) = SQR(4 * x - x * x)
     'DEF fnZa (x, y) = 0: DEF fnZb (x, y) = 2 * x * x * y
     'DEF fnF (x, y, z) = 1

     'Xa = -2: Xb = 2
     'DEF fnYa (x) = -SQR(4 - x * x): DEF fnYb (x) = SQR(4 - x * x)
     'DEF fnZa (x, y) = 0: DEF fnZb (x, y) = SQR(4 - x * x - y * y)
     'DEF fnF (x, y, z) = z

     'Xa = 0: Xb = 6
     'DEF FnYa (x) = 0: DEF FnYb (x) = (12 - 2 * x) / 3
     'DEF fnZa (x, y) = 0: DEF fnZb (x, y) = (12 - 2 * x - 3 * y) / 6
     'DEF FnF (x, y, z) = 1

     'Xa = -Pi / 2: Xb = Pi
     'DEF FnYa (x) = -SIN(x) ^ 2: DEF FnYb (x) = SIN(x) ^ 2
     'DEF fnZa (x, y) = -x - y: DEF fnZb (x, y) = x + y
     'DEF FnF (x, y, z) = SIN(x) + COS(y) + SIN(z)


     CLS

     Hassasiyet = .01 ' Değiştirilebilir, 0.01-0.0001 arası normal değer

     CALL GaussLegendre3(Xa, Xb, Hassasiyet, Dintegral, iHata)

     IF iHata <> 0 THEN
      PRINT "Yakınsamadı, yaklaşık değer(GaussLegendre3)="; Dintegral
     ELSE
      PRINT "İntegral(GaussLegendre3)="; Dintegral
     END IF

END  ' GaussLegendre3 ana sonu
```

Üç katlı integral: Gauss-Legendre metodu ile integrasyon (alt program)

```
SUB GaussLegendre3 (Xa, Xb, Hassasiyet, Dintegral, iHata)
'------------------------------------------------------------------------
' Ahmet TOPÇU, Osmangazi Üniversitesi, Eskişehir, 2010
' F(x,y,z) fonksiyonunun [Xa,Xb], [Ya(x),Yb(x)] ve [Za(x,y),Zb(x,y)]
' aralığında üç katlı integralini Gauss-Legendre metodu ile hesaplar.

'Veri:
' Xa, Xb : x ekseni boyunca integralin alt ve üst sınırı
' Ya(x), Yb(x): y ekseni boyunca alt ve üst sınırın fonksiyonu
' Za(x,y), Zb(x,y): z ekseni boyunca alt ve üst sınırın fonksiyonu
' F(x,y,z) : integralı hesaplanacak fonksiyon
' Hassasiyet: Hassasiyet

' Aralıklar ve f(x,y,z) çağıran programda Xaşağıdaki gibi tanımlanmalıdır:
' Xa=...            (sabit sayı)
' Xb=...            (sabit sayı)
' DEF FnYa(x)=...   (sabit sayı veya fonksiyon)
' DEF FnYb(x)=...   (sabit sayı veya fonksiyon)
' DEF FnZa(x,y)=... (sabit sayı veya fonksiyon)
' DEF FnZb(x,y)=... (sabit sayı veya fonksiyon)
' Def fnF(x,y,z)=... (integralı hesaplanacak fonksiyon)

'Çıktı:
' Dintegral: hesaplanan integral
' iHata=0 : integral yakınsadı
' iHata<>0: integral yakınsamadı

'Çağrılan alt program: GaussLegendre3
'------------------------------------------------------------------------
    MaxNokta = 1024  ' =2^10 Öngörülen maksimum Gauss-Legendre nokta sayısı
    DIM x(MaxNokta), w(MaxNokta)

    iHata = 0
    Eskiintegral = 0
    iMax = LOG(MaxNokta) / LOG(2)

    FOR i = 4 TO iMax
     NoktaSayisi = 2 ^ i
     NoktaSayisiX = NoktaSayisi  ' x yönü nokta sayısı
     NoktaSayisiY = NoktaSayisi  ' y yönü nokta sayısı
     NoktaSayisiZ = NoktaSayisi  ' z yönü nokta sayısı

' Gauss-Lgendre koordinatları:
     CALL GauLeg(-1, 1, x(), w(), NoktaSayisi)

     Hx = .5 * (Xb - Xa)
     HxOrta = .5 * (Xb + Xa)
     Dintegral = 0
     FOR iX = 1 TO NoktaSayisiX
      x = Hx * x(iX) + HxOrta
      AintX = 0
      Ya = fnYa(x)
      Yb = fnYb(x)
      Hy = .5 * (Yb - Ya)
      HyOrta = .5 * (Yb + Ya)

      FOR iY = 1 TO NoktaSayisiY
       y = Hy * x(iY) + HyOrta
       aintY = 0
       Za = fnZa(x, y)
       Zb = fnZb(x, y)
       Hz = .5 * (Zb - Za)
       HzOrta = .5 * (Zb + Za)

       FOR iZ = 1 TO NoktaSayisiZ
        z = Hz * x(iZ) + HzOrta
        Fxyz = fnF(x, y, z)
        aintY = aintY + w(iZ) * Fxyz
       NEXT iZ
       AintX = AintX + w(iY) * Hz * aintY

      NEXT iY
      Dintegral = Dintegral + w(iX) * Hy * AintX
     NEXT iX

     Dintegral = Dintegral * Hx

     hata = ABS(Dintegral - Eskiintegral)
     IF hata <= Hassasiyet THEN EXIT SUB
     Eskiintegral = Dintegral

    NEXT i

    iHata = 1' Yakınsamadı

END SUB  ' GaussLegendre3 sonu
```

GauLeg: Bu alt program GaussLegendre3 tarafından çağrılır

```
SUB GauLeg (Ai, Bi, x(), w(), n)
'------------------------------------------------------------------------
' Gauss-Legendre integrasyon ordinatları ve ağırlıkları hesaplanır
' Ai: integralin alt sınırı
' Bi: integralin üst sınırı
' x(n): Gauss-Legendre ordinatlarının depolandığı vektör
' w(n): Gauss-Legendere ağırlıklarının depolandığı vektör
' n: Gaus-Legendre integral nokta sayısı

' çağrılan alt program: yok

' Fortran kodu http://www.haoli.org/nr/bookf/f4-5.ps den alınmıştır
'------------------------------------------------------------------------
   x1 = Ai
   x2 = Bi
   Pi = 4 * ATN(1)' Pi=3.14... sayısı
   Hassasiyet = 3E-14: ' Gauleg için hassasiyet(değiştirmeyiniz!)

   m = (n + 1) / 2
   xm = .5 * (x2 + x1)
   x1 = .5 * (x2 - x1)

   FOR i = 1 TO m
    z = COS(Pi * (i - .25) / (n + .5))
' # işareti DOUBLE anlamındadır
    z1 = 0
    WHILE ABS(z - z1) > Hassasiyet
     p1 = 1
     p2 = 0
     FOR j = 1 TO n
      p3 = p2
      p2 = p1
      p1 = ((2 * j - 1) * z * p2 - (j - 1) * p3) / j
     NEXT j
     pp = n * (z * p1 - p2) / (z * z - 1)
     z1 = z
     z = z1 - p1 / pp
    WEND

    x(i) = xm - x1 * z
    x(n + 1 - i) = xm + x1 * z
    w(i) = 2 * x1 / ((1 - z * z) * pp * pp)
    w(n + 1 - i) = w(i)
   NEXT i

END SUB ' Gauleg sonu
```