

**Bilgisayar Müh. için Diferansiyel Denklemler**

**BLM2642**

**Prof. Dr. Sırma YAVUZ**

**Ödev 2: 4. Derece Runge-Kutta Metodu ile Diferansiyel Denklemleri Çözme**

**Mehmet Ali Duran**

**21011090**

**ali.duran@std.yildiz.edu.tr**

**Ödevin Konusu**:

N terimli sabit katsayılı doğrusal adi diferansiyel denkleminin Runge Kutta-4 yöntemiyle sayısal çözümünü C dilinde kodlayınız.

**Girdiler:**

Denklemin terim sayısını ifade eden n değeri ve bu n değerine göre diferansiyel denklemin katsayıları kullanıcıdan alınmalıdır. Ayrıca, sayısal çözüm için denklem çözümü sonrasında elde edilecek sayısal değer için bağımsız değişken değeri (t veya x) kullanıcıdan alınmalıdır. **Denklemi modellerken mutlaka struct kullanınız.**

**Çıktı:**

Runge Kutta -4 ile elde edilen çözümünün t veya x bağımsız değişkenine göre yaklaşık çözümü ekranda gösterilmelidir. Ayrıca iterasyonlarda elde edilen yaklaşım sonuçları aşamalı olarak ekrana yazdırılmalıdır. Raporunuzda olacak test örneklerinde çözümü bilinen denklemleri kullanınız, bu sayede üretilen nümerik çözümle gerçek çözüm arasındaki mutlak hata değerini ekranda gösteriniz.

**Açıklamalar**

Bu ödevde Runge-Kutta4 metodu ile diferansiyel denklemlerin çözümünü iteratif bir şekilde yapan C programının açıklaması vardır. Programda denklemler bir struct yapısı ile tutulmuştur ve bu yapı denkleme ait

N🡪terim sayısını,

coefficient🡪terimlerin katsayılarını,

order🡪terimlerin derecelerini,

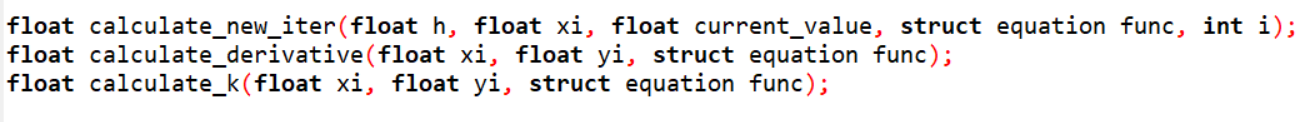
var🡪o anki değişkenin x mi y mi olduğunu

tutmaktadır. Ayrıca malloc() fonksiyonu için stdlib.h ve pow() fonksiyonu için math.h kütüphaneleri kullanılmıştır.

metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

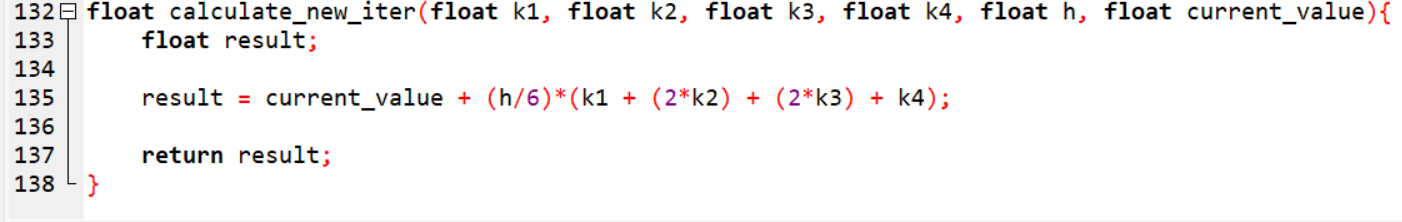
Açıklama otomatik olarak oluşturuldu

Burada şöyle bir tasarım yapılmıştır. y’ ifadesini alınca karşıda f(x, y) yani x ve y’lerden oluşan terimler kalıyor, sabit katsayılı dediği için x.y gibi ifadeler de bulunmaması gerekir. Bu durumda örnek olarak x^2 + y gibi bir ifade olabilir bu ifadede x ve y’yi yerine koyup işlem yapmak istediğimizde o an işlem yaptığımız ifadenin x mi y mi olduğunu bilmemiz gerekiyor. Bunun için de ayrıca bir var dizisi tutularak bu dizi içinde ifade x ise 1, y ise 0 olarak tutuluyor. İlerde değer hesaplama kısmında bu bilgiden yararlanılarak hangi değişken değerinin kullanılacağı seçiliyor. Bunu da fonksiyonun içinde basit bir if kontrolü ile yapıyoruz.

Kodda kullanılan fonksiyonlar:

Kısaca işlevleri ve içerikleri şöyle:

calculate\_new\_iter: Bu fonksiyon parametre olarak aldığı xi ve yi(current\_value) değerleri ve h ile o anki değer üzerinden yeni değeri hesaplar. current\_value başlangıçtaki değerdir(yani y0), y(x0) = y0. Bu formül Runge-Kutta 4’ün kendisinden gelmektedir.

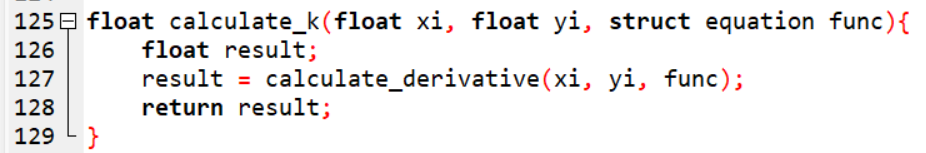


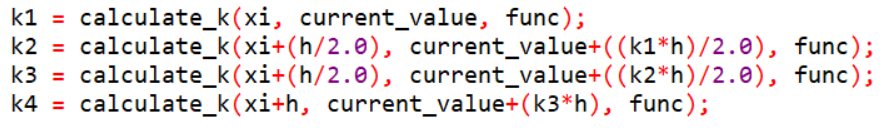
calculate\_derivative: Bu fonksiyon y’ fonksiyonunda x ve y’ yi yerine koyarak sonucu döndürür.

metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

calculate\_k: Bu fonksiyon calculate\_derivative fonksiyonu üzerinden k değerlerini hesaplar. Burada k değerleri için gelen parametreler farklıdır.





Kodumuz gerekli girdileri aldıktan sonra her iterasyonun sonucunu ve o iterasyon için hata değerini ekrana basar.

Şimdi de 4 örnek üzerinden kodun çıktıları gösterilecektir.

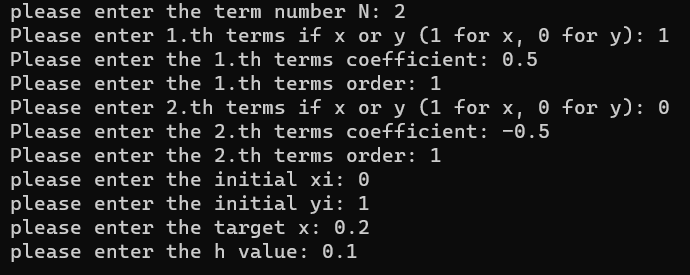
**Örnek-1**

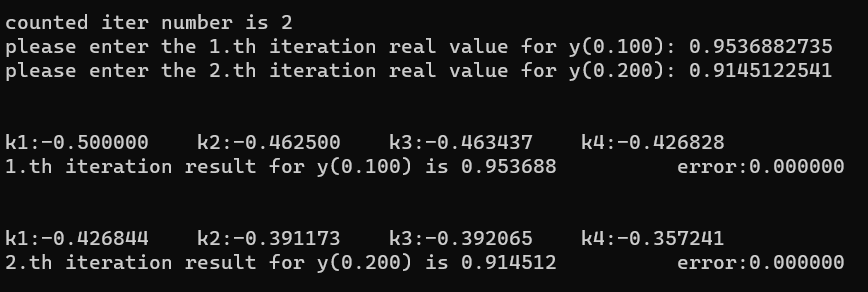
**Find y(0.2) for *y*′=(*x*-*y/)*2, *x*0=0,*y*0=1, with step length 0.1 using Runge-Kutta 4 method (1st order derivative)**

y’=0.5x–0.5y X0=0, Y0=1, h=0.1,

Real Y(0.1) = 0.9536882735

Real Y(0.2) = 0.9145122541





**Örnek-2**

Y’=-2x-y, x0=0, y0=-1, h=0.1, y(0.5) = ?

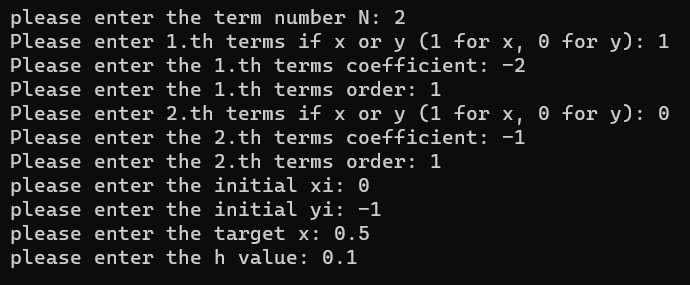
Real Y(0.1) = -0.9145122541

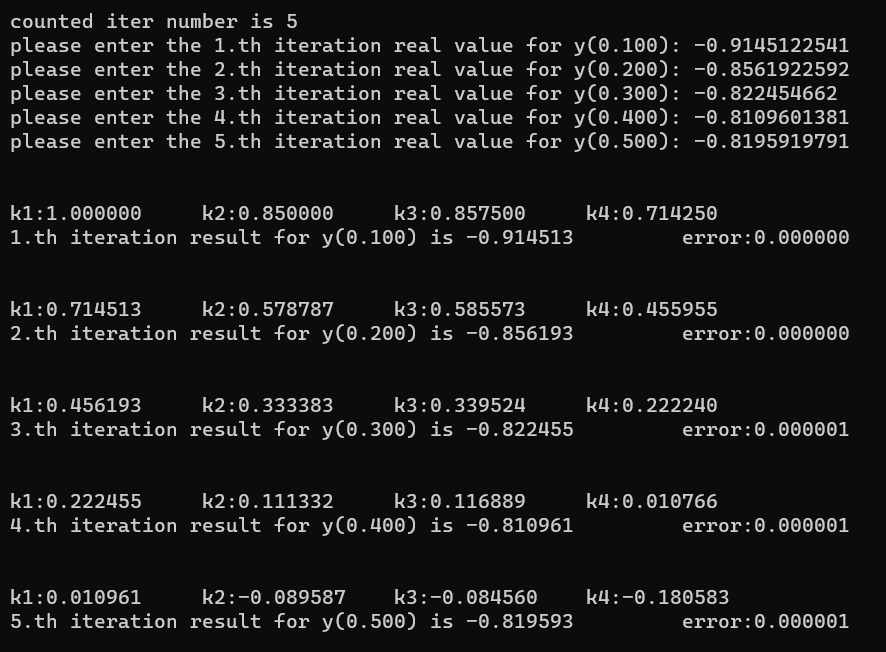
Real Y(0.2) = -0.8561922592

Real Y(0.3) = -0.822454662

Real Y(0.4) = -0.8109601381

Real Y(0.5) = -0.8195919791



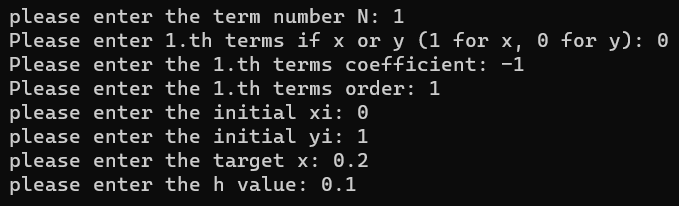


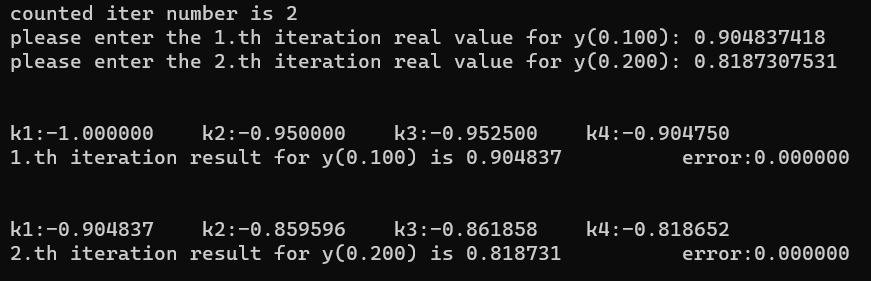
**Örnek-3**

Y’=-y, x0=0, y0=1, h=0.1, y(0.2) = ?

Real Y(0.1) = 0.904837418

Real Y(0.2) = 0.8187307531





**Örnek-4**

y'=x^2+y x0=0, y0=1, h=0.1, y(0.2) = ?

Real Y(0.1) = 1.105512754

Real Y(0.2) = 1.224208274

