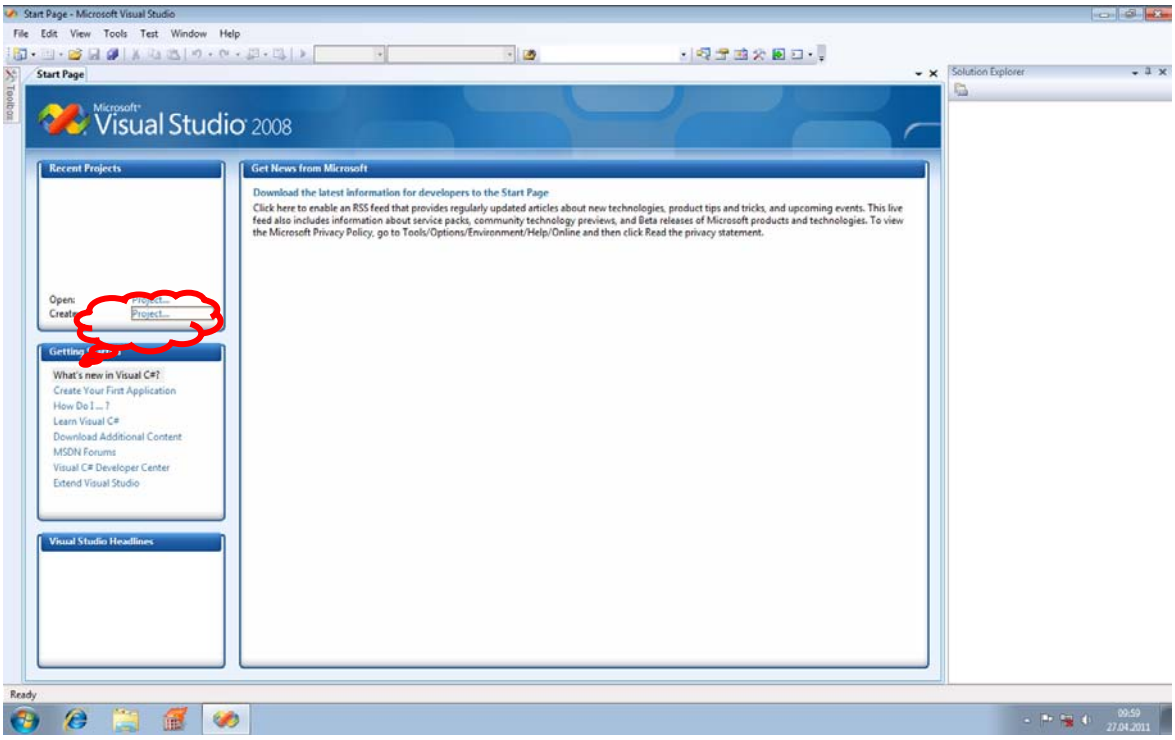
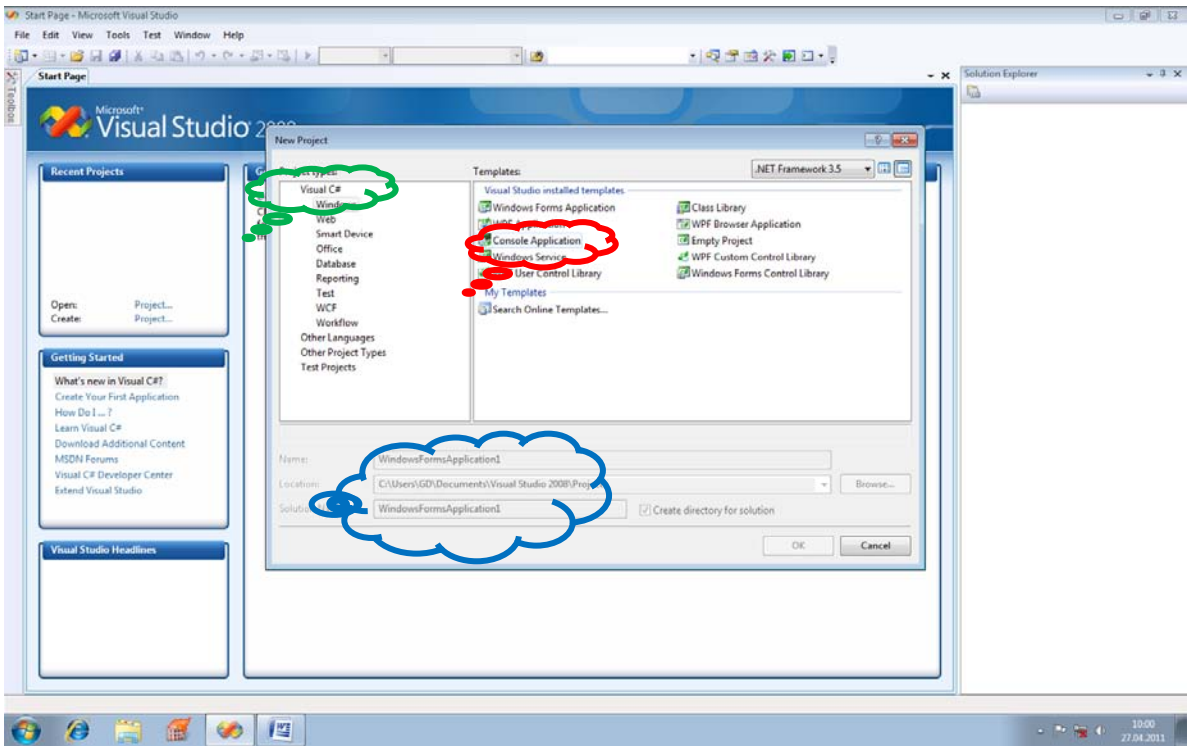
C# Programının çalıştırılması, Program yazma ve Çalıştırma

Program çalıştırıldığında karşımıza Şekil 1.deki arayüz çıkacaktır. Burada daha önce yapılan bir program üzerinde çalışılacak ise kırmızı bulut içinde "Open-Project" kutucuğuna tıklanır. Yeni bir program yazılacak ise "Create-Project" kutucuğuna tıklanır. Yeni program yazacağımız için "Create-Project" e tıklamalıyız.



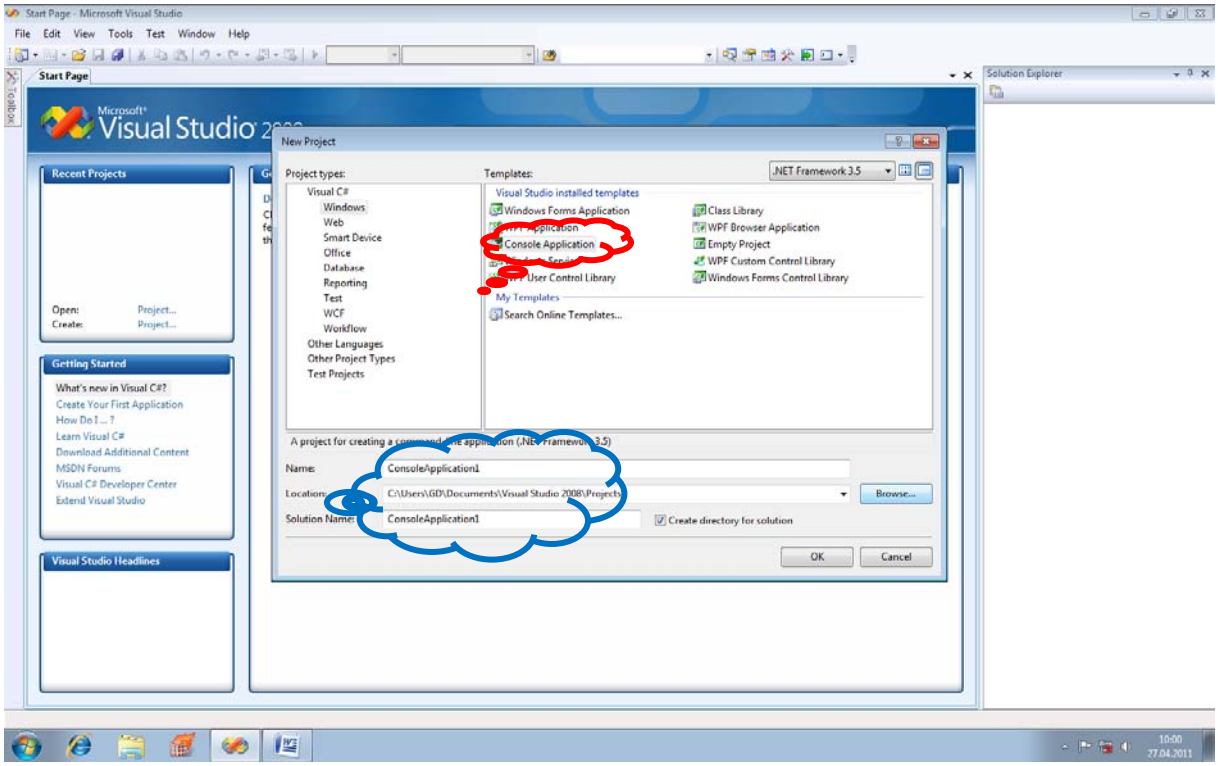
Şekil 1.

"Create-Project" tıklandığında Şekil 2.deki ekran karşımıza çıkacaktır. Açılan pencerede "Visual C#" görünmelidir (Yeşil bulut). Şayet pencerede "Visual C#" görünmüyorsa Şekil 2.A'dan devam ederek görüntülenmesini sağlayabilirsiniz. "Burada konsol uygulaması çalışacağımız için kırmızı bulut içindeki "Console Application" e tıklanır. Dikkat edilmesi gereken nokta "Console Application" seçildiğinde mavi bulut içindeki kısmın değişmesi gerekir.



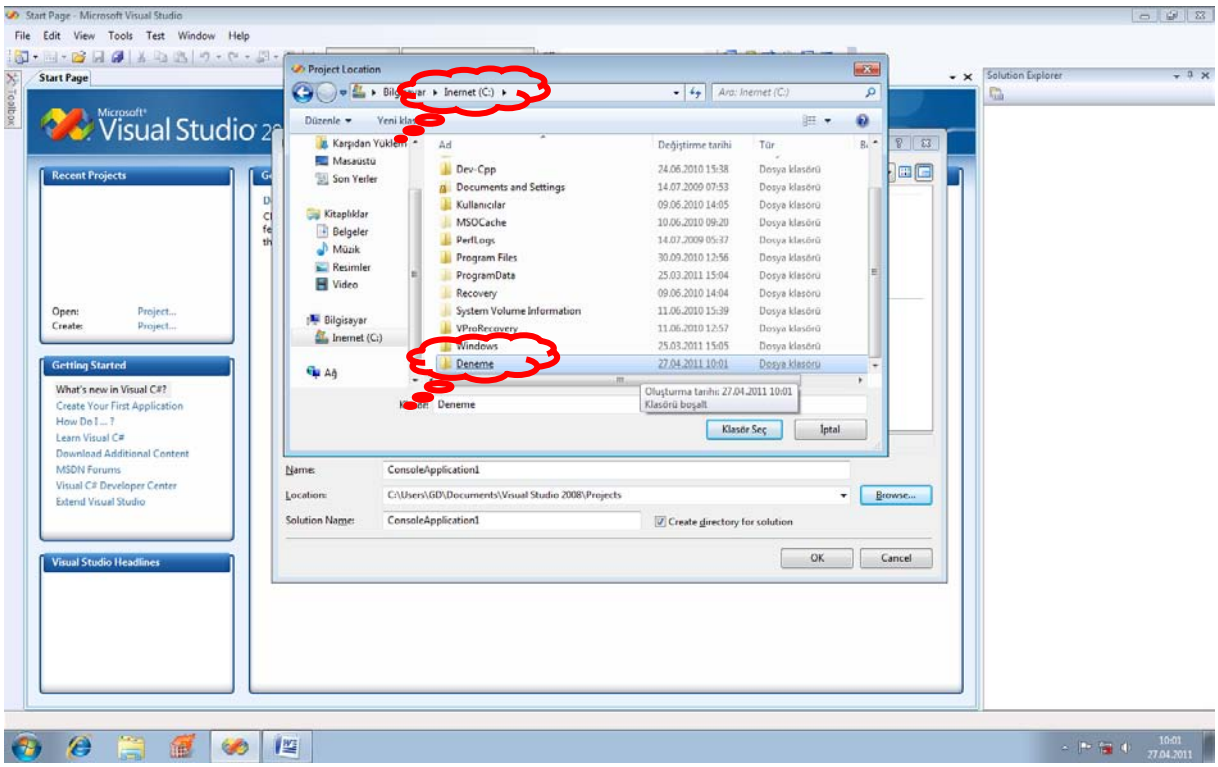
Şekil 2.

"Console Application" seçildiğinde mavi bulut içinde nasıl bir değişim olduğu Şekil 3.te görülmektedir. Burada kolay erişim sağlamak için dosyaların kaydedildiği "Location" kısmını değiştirmek gerekir. Yer değiştirmek için "Location" yanındaki "Browse" butonuna tıklanır.



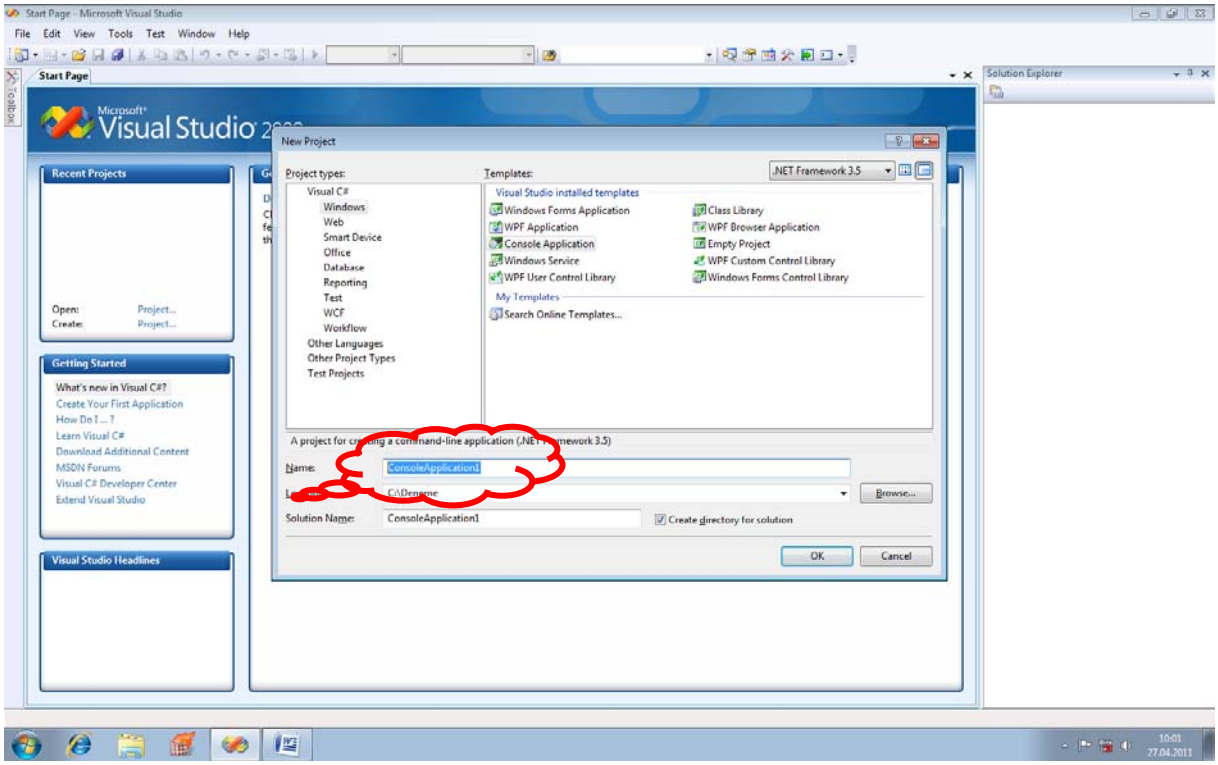
Şekil 3.

"Browse" butonuna tıklandığında Şekil 4.deki pencere açılır. Bu çalışmamızda C: 'de "deneme" isminde bir klasör açıp çalışmalarımızı burada yapmak istiyoruz (kırmızı bulutlar).



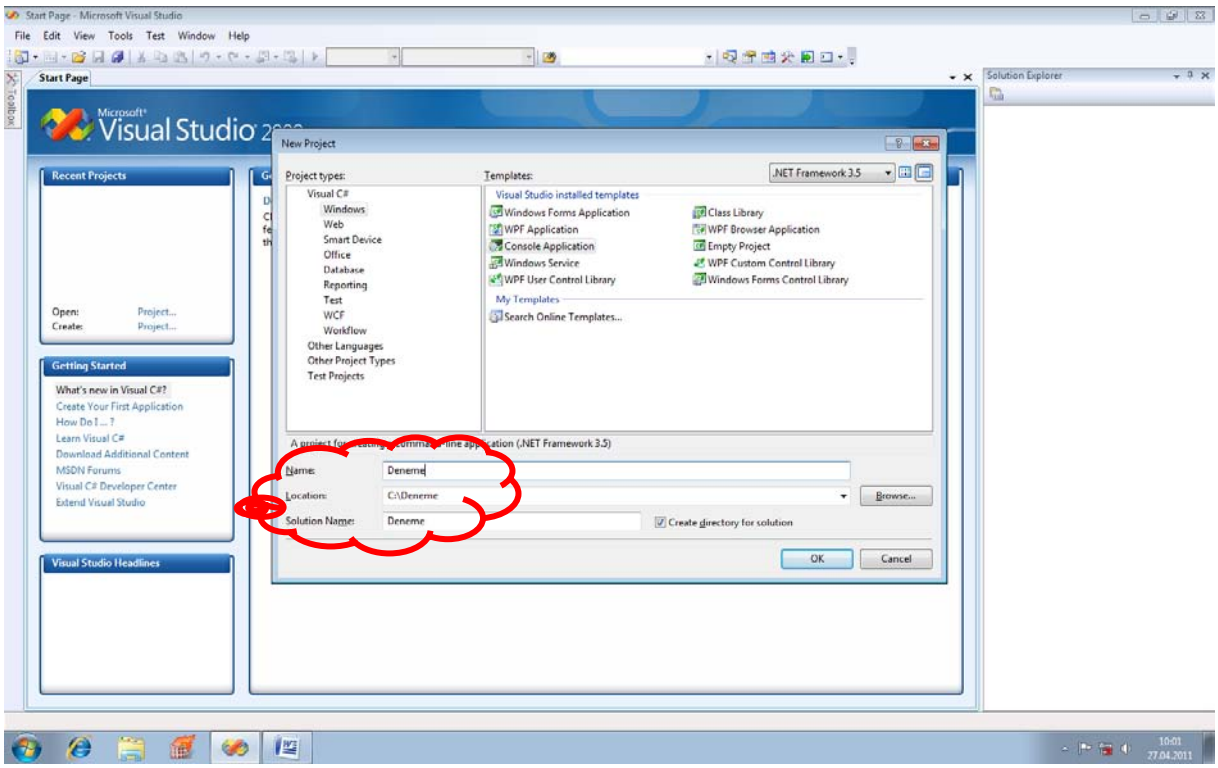
Şekil 4.

Şekil 5.te yazacağımız programın ismini değiştirmek için kullanacağımız gösterilmektedir (kırmızı bulut). Burada ilk programımız için “deneme” ismini verelim.



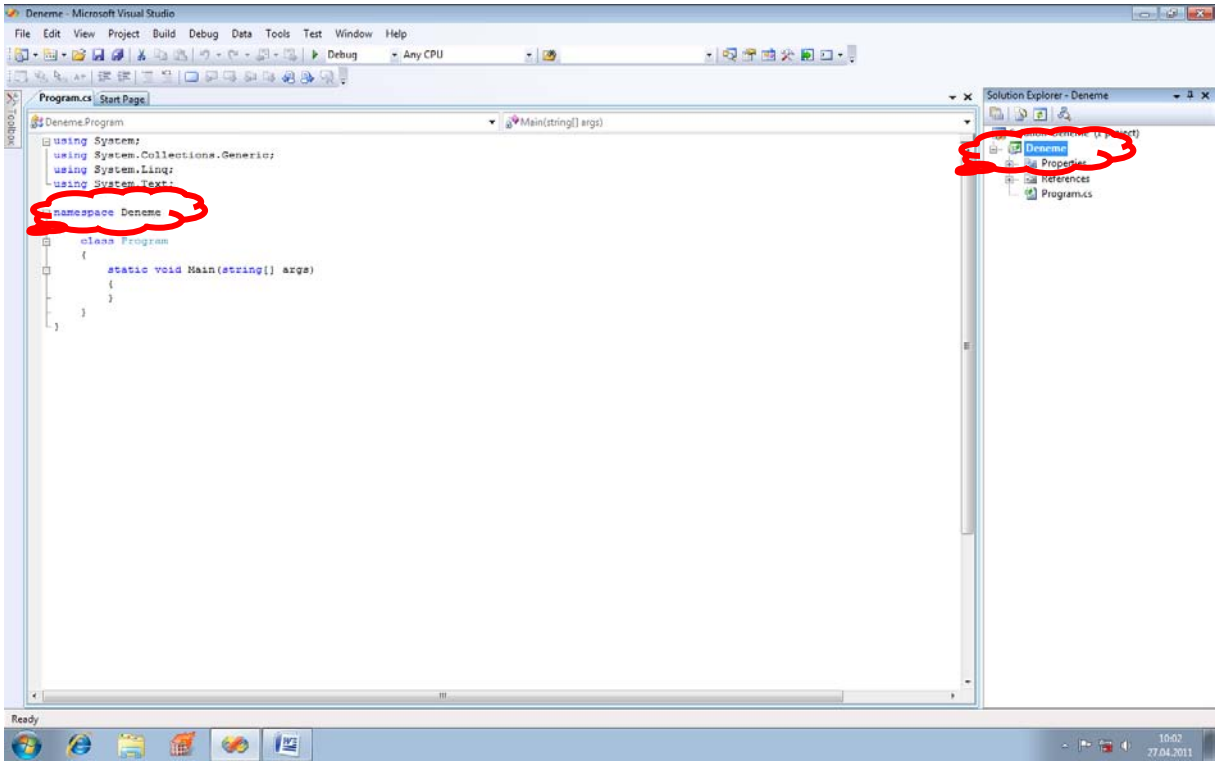
Şekil 5.

Şekil 6.da konsol uygulaması seçildikten sonra dosya ismi ve dosyanın bilgisayardaki yeri değiştirildikten sonraki durumu görülmektedir (kırmızı bulut). Gerekli değişiklikler yapıldıktan sonra “OK” butonuna basılır.



Şekil 6.

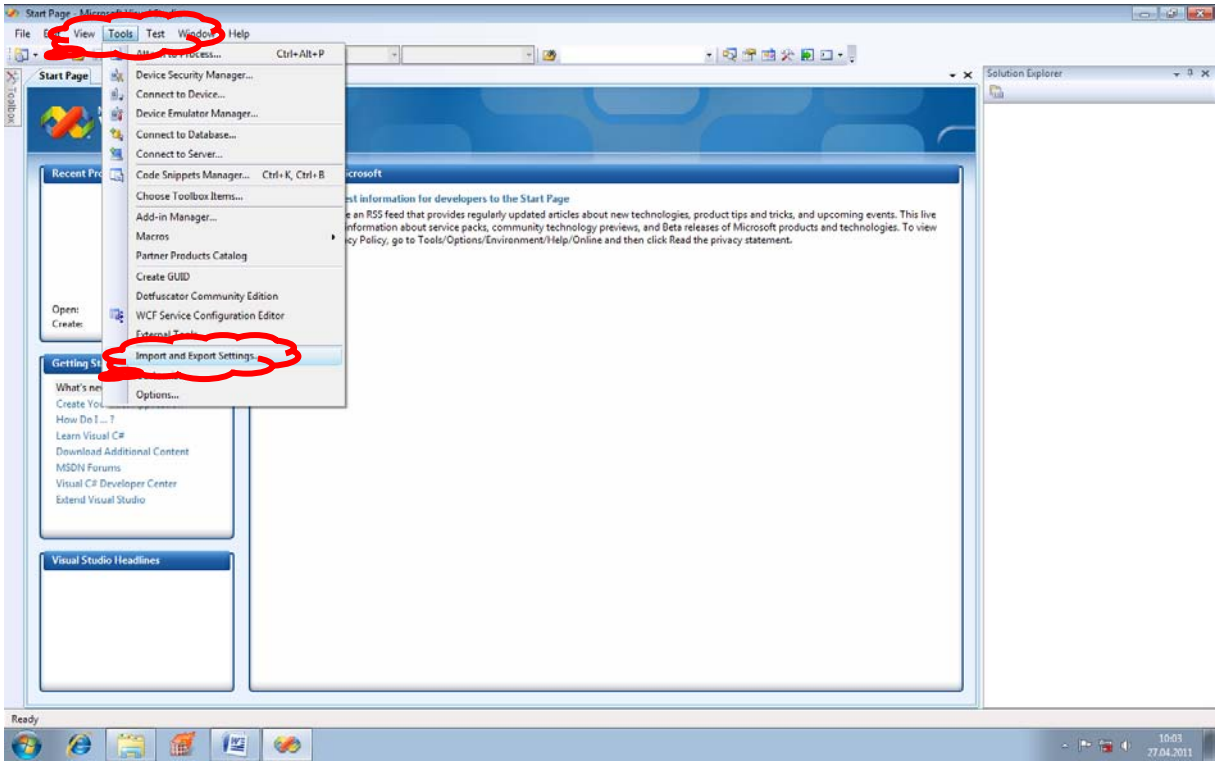
"OK" butonuna basıldığında program yazmak için gerekli ortam açılacaktır (Şekil 7). Burada kırmızı bulutlar içindeki kısımlarda verdiğimiz isimler olmalıdır (deneme diye isimlendirmiştik).



Şekil 7.

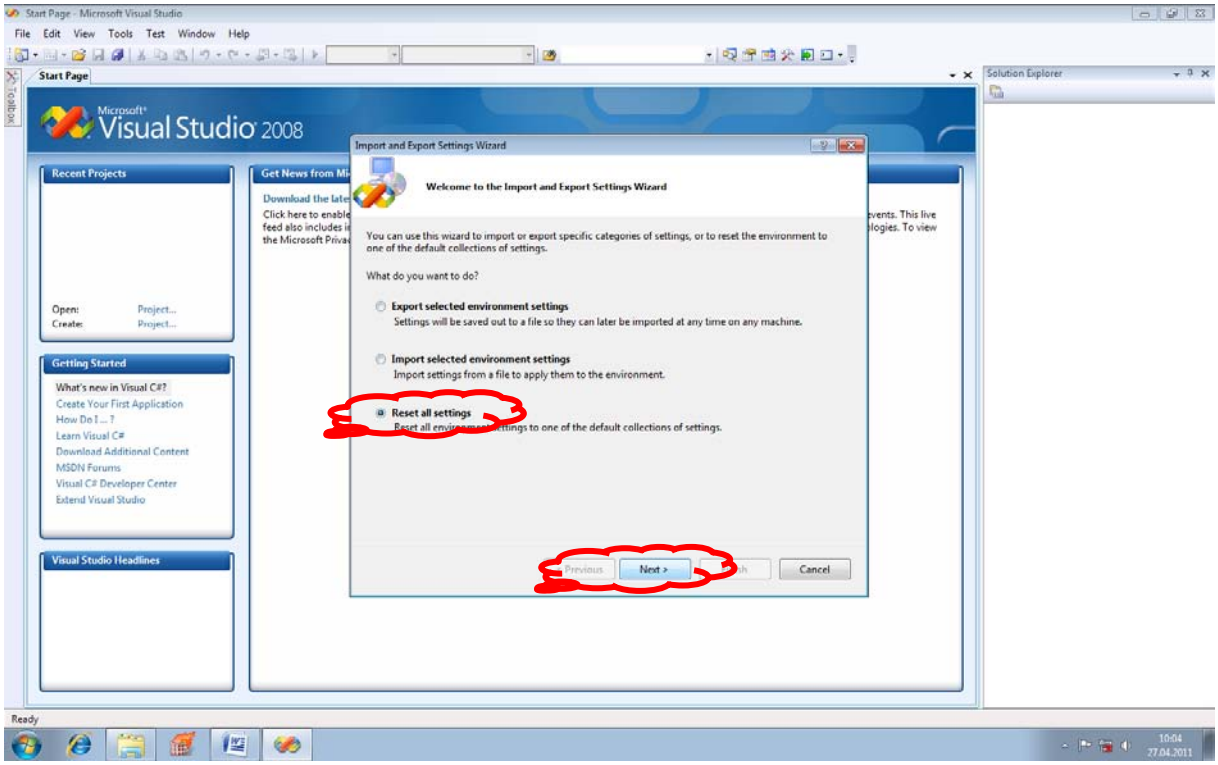
Şekil 2'de yeşil bulut içinde gösterilen yerde "Visual C#" görülmüyorsa aşağıdaki işlemler yapılarak görüntülenebilir.

Şekil 2A.1'deki gibi "tools" seçilir, açılan menüden "Import and Export Settings" e tıklanır (kırmızı bulutlar).



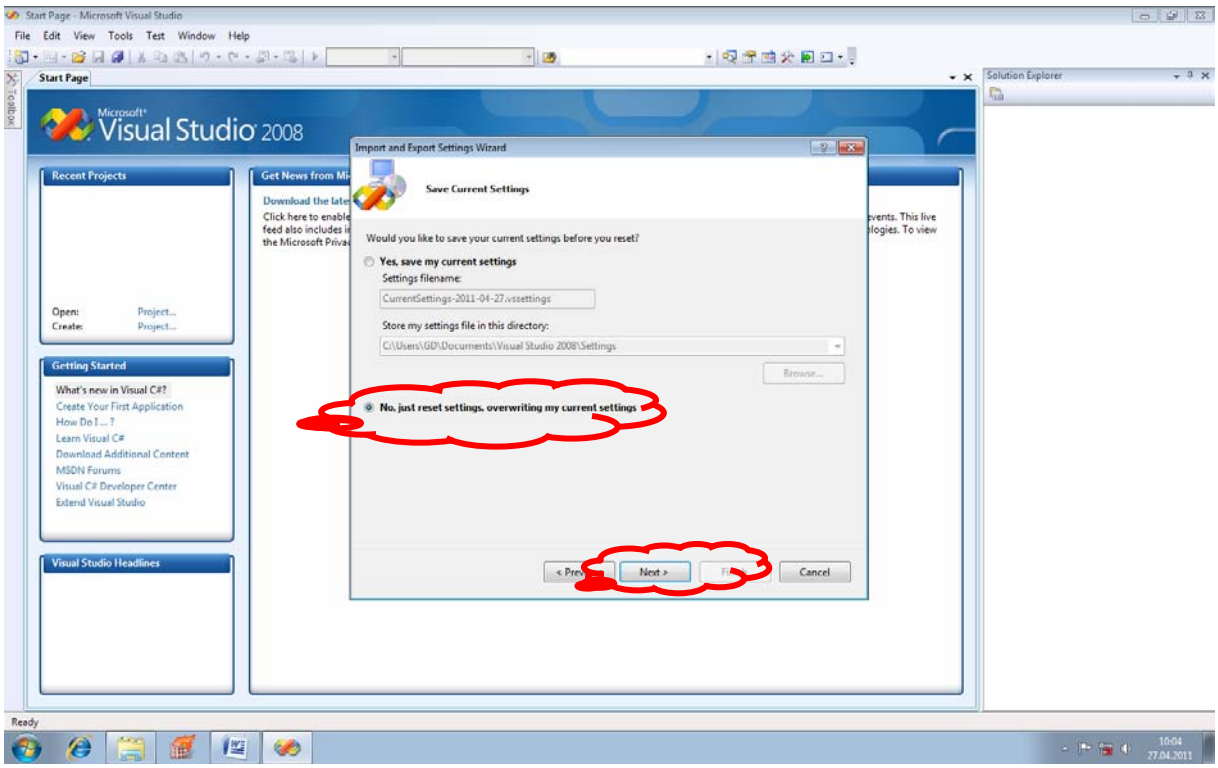
Şekil 2A.1

“Import and Export Settings” e tıklandığında Şekil 2A.2’deki pencere açılır. Burada “Reset all settings” seçilir ve “Next” butonuna basılır (kırmızı bulutlar).



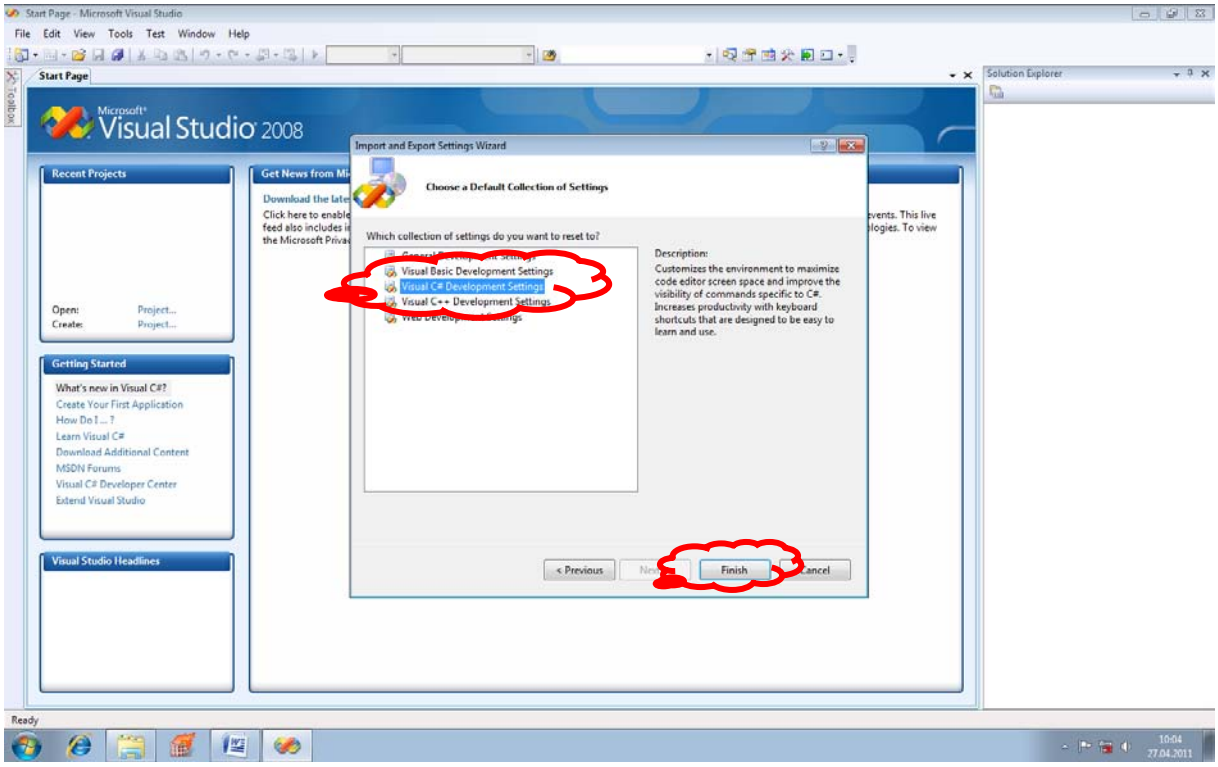
Şekil 2A.2

Açılan yeni pencerede (Şekil 2A.3) “No, just reset settings, …” seçilir ve “Next” butonuna basılır (kırmızı bulutlar).



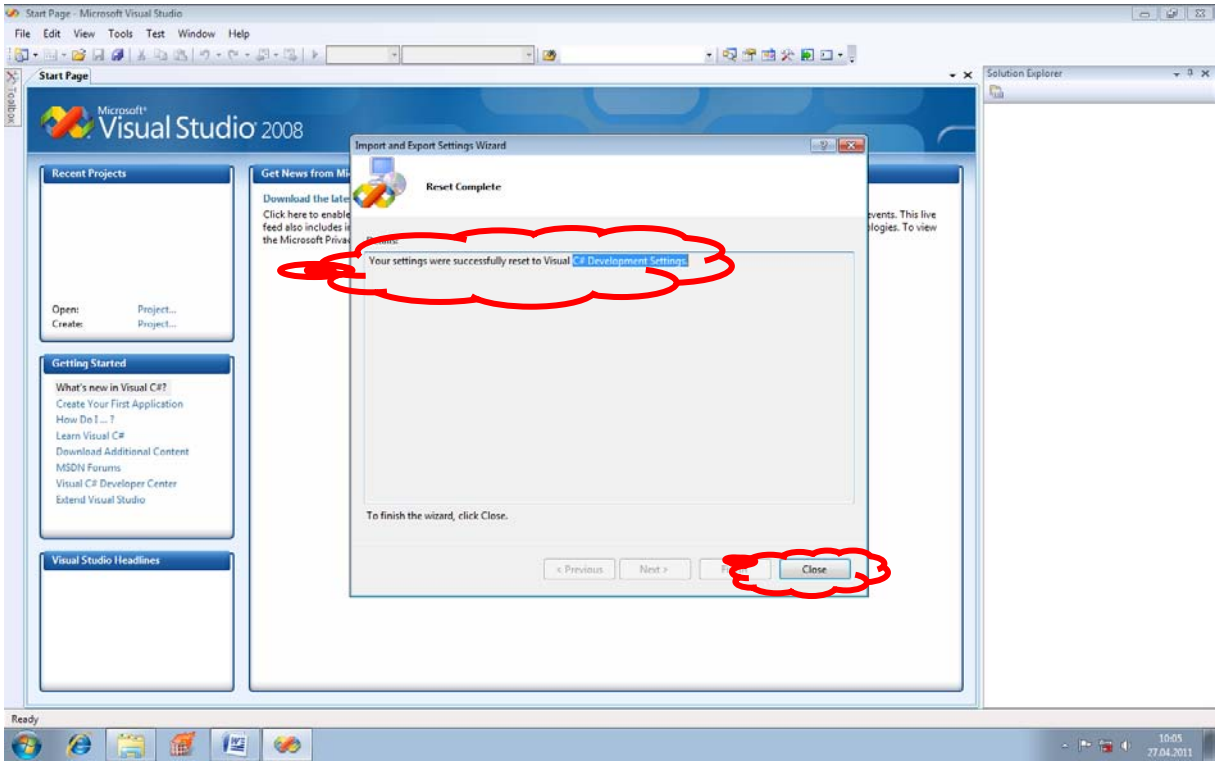
Şekil 2A.3

Açılan yeni pencerede (Şekil 2A.4) “Visual C# Development Settings” tıklanır ve “Finish” butonuna basılır (kırmızı bulutlar).



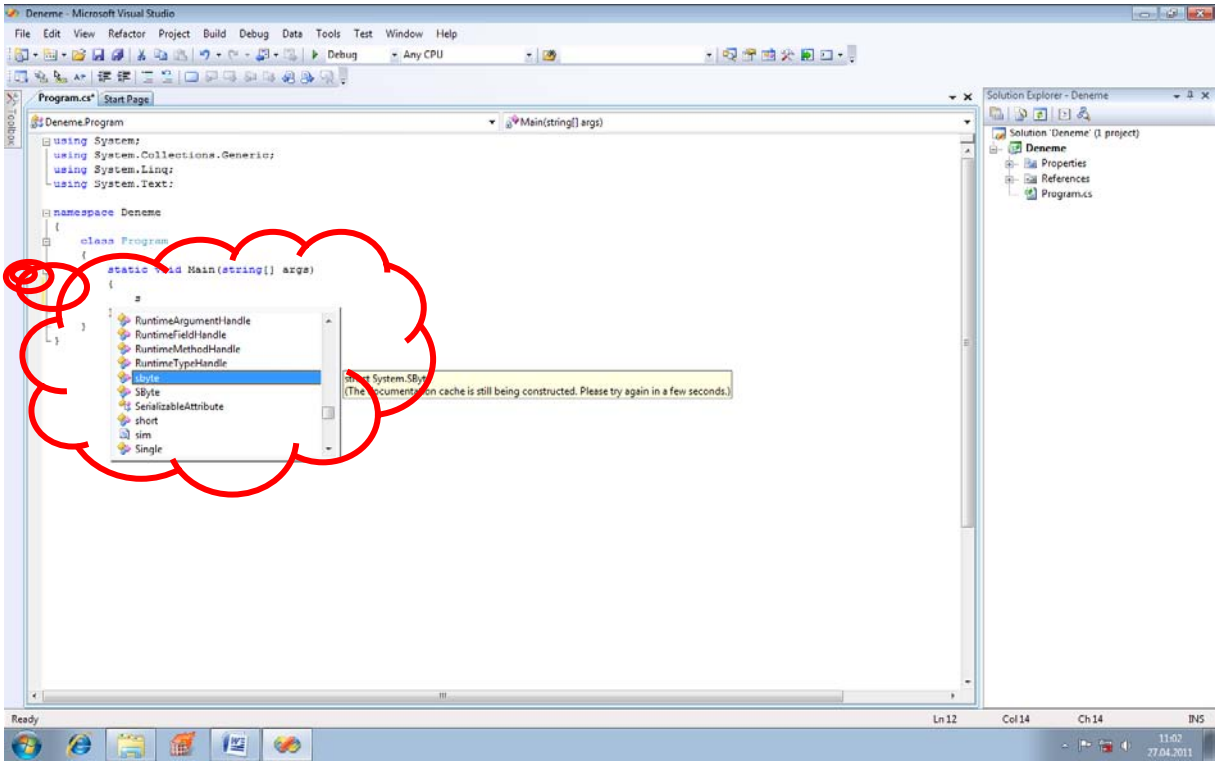
Şekil 2A.4

“Finish” butonuna tıklandığında açılan yeni pencerede (Şekil 2A.5) kırmızı bulut içindeki yazı görünüyorsa işlem başarıyla gerçekleştirilmiştir. Artık Şekil 2’de “Visual C#” yazısı görünecektir. “Close” butonuna basılarak pencere kapatılır. Bundan sonra Şekil 2.’den devam ederek ayarları yapmak gerekir.



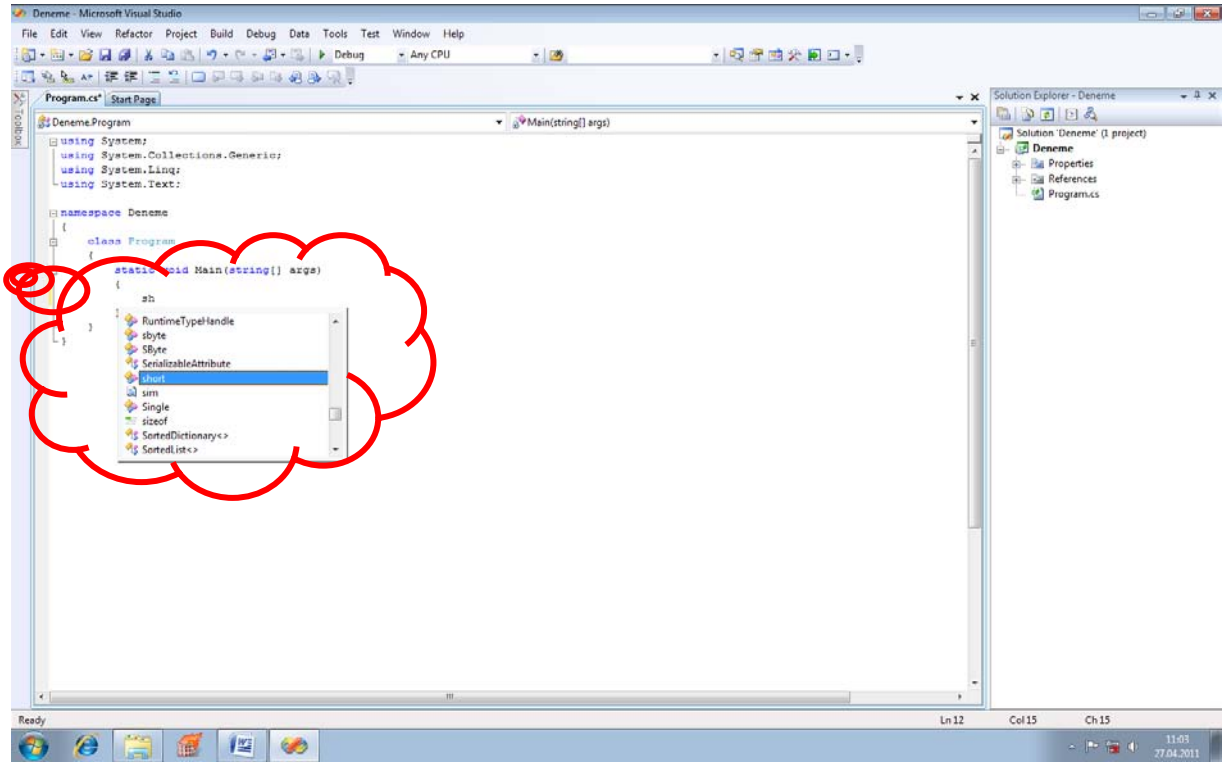
Şekil 2A.5

Program yazmak üzere açılan ortamda (Şekil 8.) komut yamaya başlarken program bize yazabileceğimiz komut listesini açar. Mesela “short” yazmak için “s” yazdığımızda ekrana “s” ile başlayan komutlar belirmektedir (kırmızı bulut).



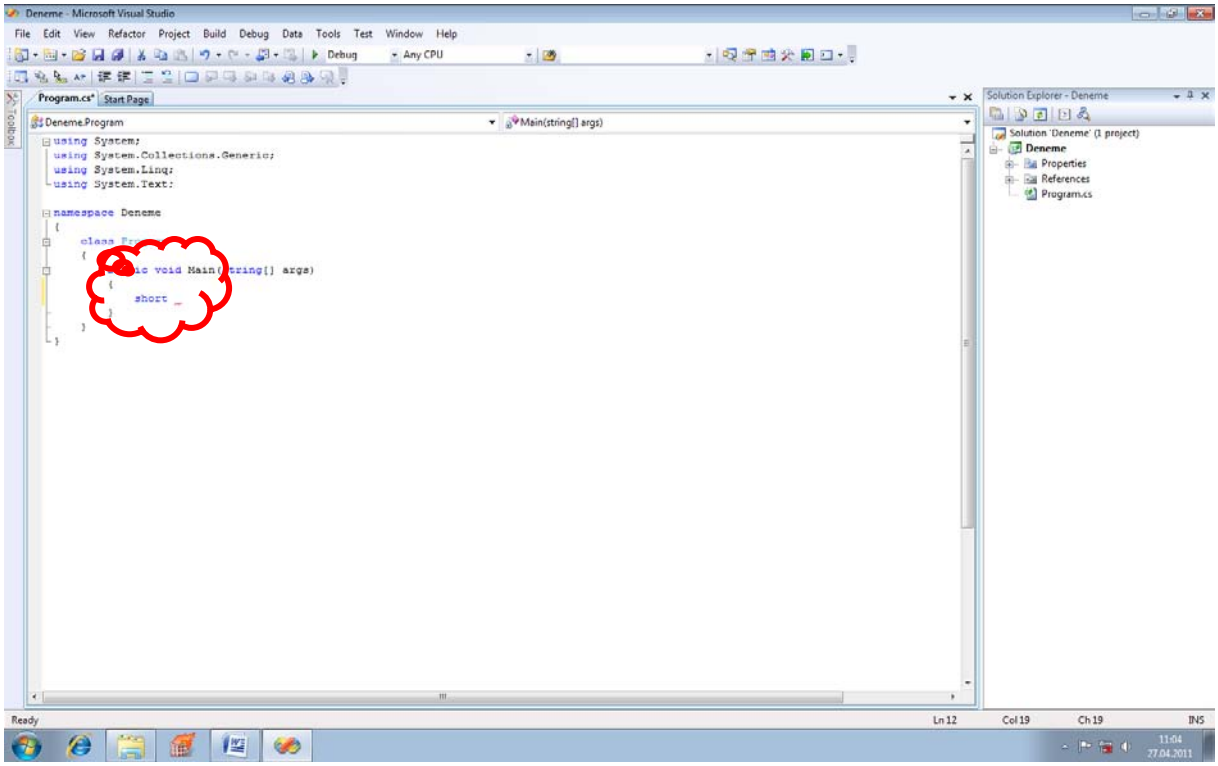
Şekil 8.

Şekil 9.’da komutu yazmaya devam ettiğimizde yani “sh” yazdığımızda istediğimiz “short” komutu gelir (kırmızı bulut). İstediğimiz komut belirince komuttan sonra hangi karakter kullanılacaksa ona basmak yeterlidir. Mesela “sh” yazdığımızda “short” belirdi, “short” tan sonra “boşluk” bırakmak gerekir. O halde boşluk tuşuna bastığımızda program kendisi komutu tamamlayacaktır. Burada istenilen komut fare ile de seçilebilir.



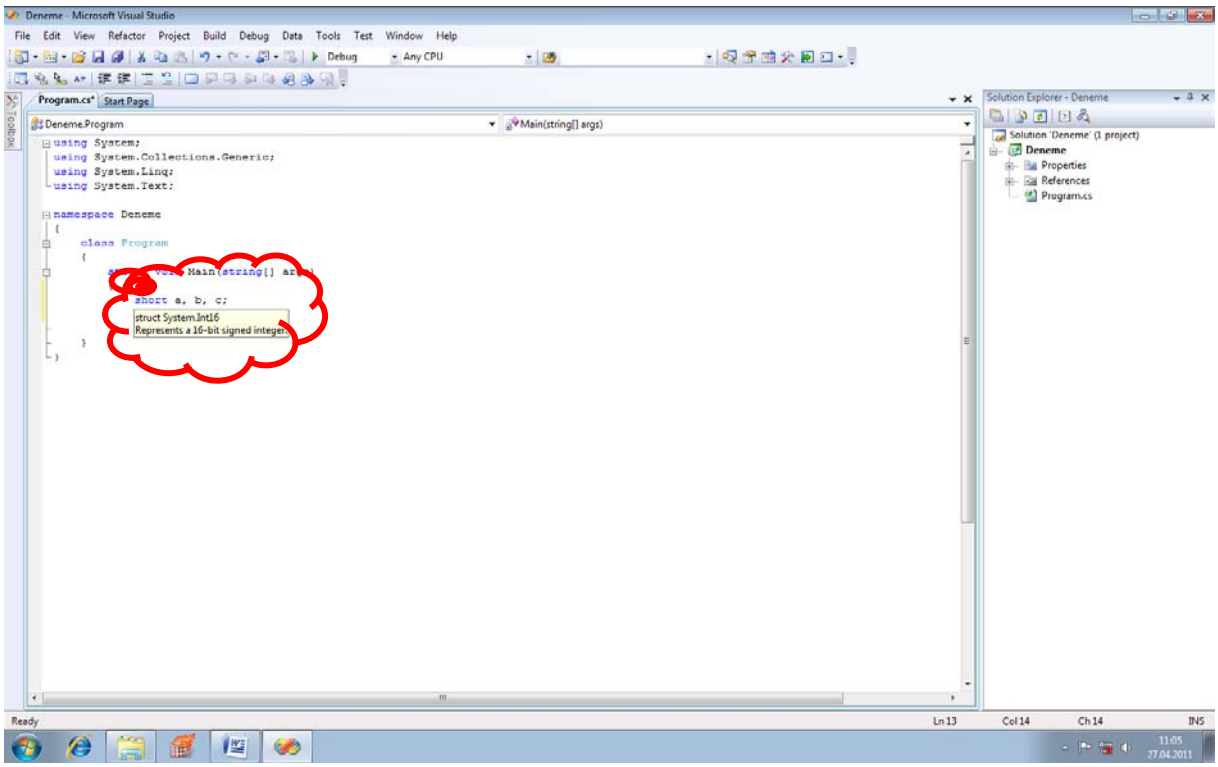
Şekil 9.

Şekil 10.da “sh” yazıp boşluk tuşuna bastığımızda “short” komutunu program kendisi tamamladı. Burada “short” komutunun yanındaki kırmızı dalga şekline dikkat etmeliyiz. Bu dalga bize komutta hata olduğunu ifade etmektedir. Yani komutta eksiklik olduğu bildirmektedir.



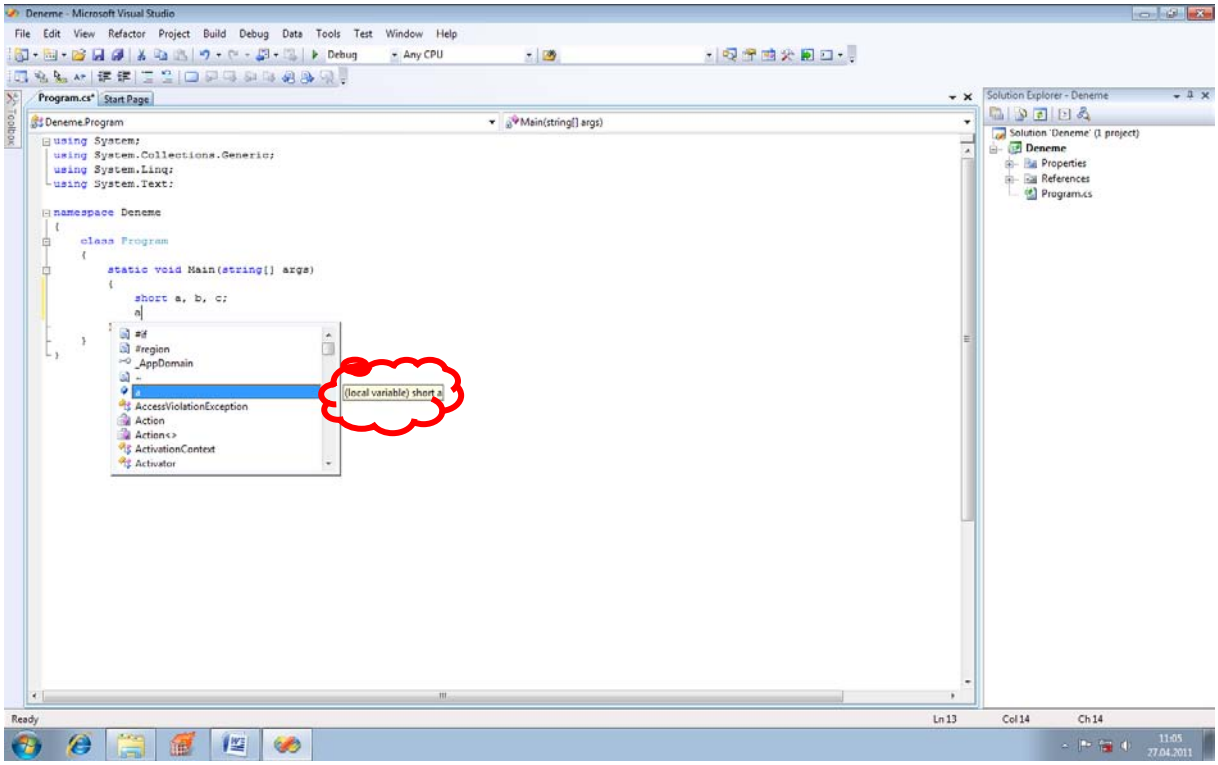
Şekil 10.

Şekil 11.de “short a, b, c;” şeklinde komutu tamamladığımızda kırmızı dalganın gittiğini görürüz. Burada ayrıca fare ile “short” komutunun üstüne geldiğimizde short’ un Int16 olması gerektiği gösterilmektedir.



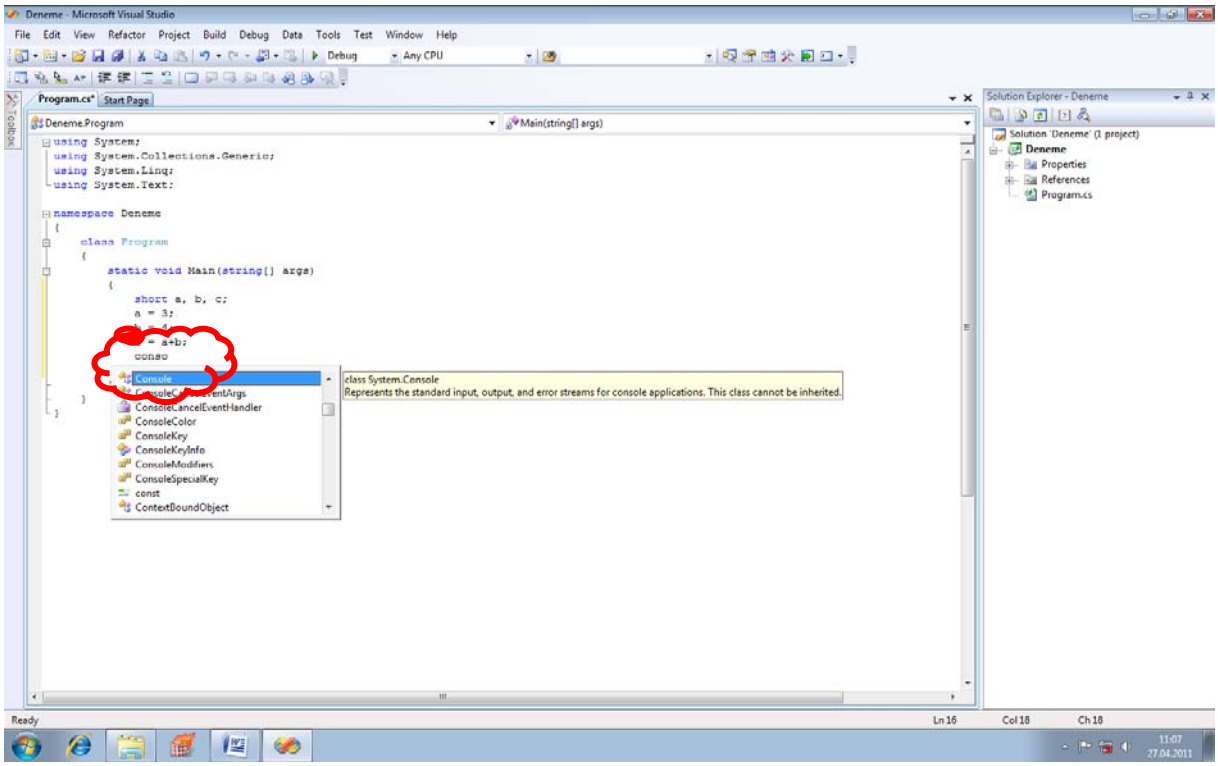
Şekil 11.

Şekil 12.de program yazmaya devam ettiğimizde mesela “a” yazdığımızda bize hemen seçenekler gelir. Burada dikkate edilmesi gereken husus “a”nın yanında “(local variable) short a” diye yazmaktadır. Bunun anlamı a lokal bir değişkendir ve hangi iki parantez içinde tanımlanmışsa sadece o parantezler içinde geçerlidir. Mesela bir “while” döngüsü içinde o tanım yapılmışsa sadece “while” içinde geçerli olur, “while” dışında o değişken hata verir.



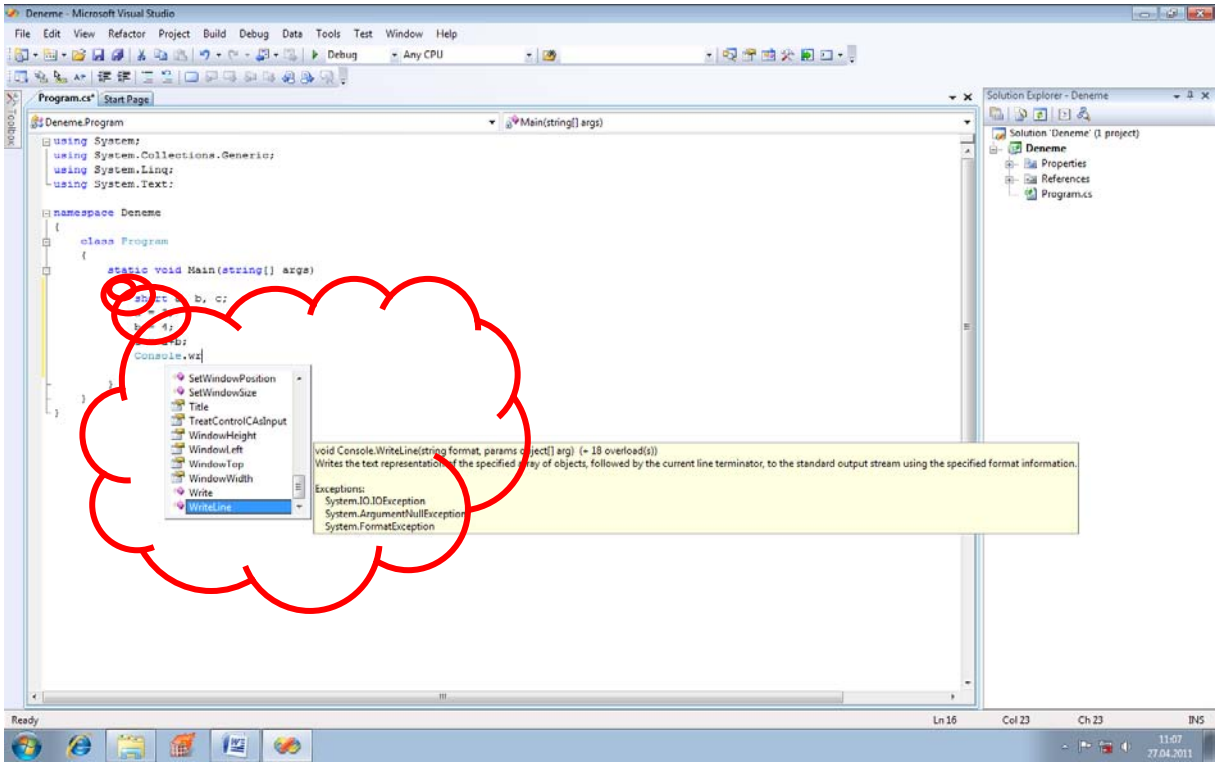
Şekil 12.

Şekil 13.te “conso” yazıldığında “console” yazılacağı için sonrasında “.” geleceğinden “.” ya bastığımızda “console.”ile yeniden bir menü açılır. Burada istenilen komut fare ile de seçilebilir.



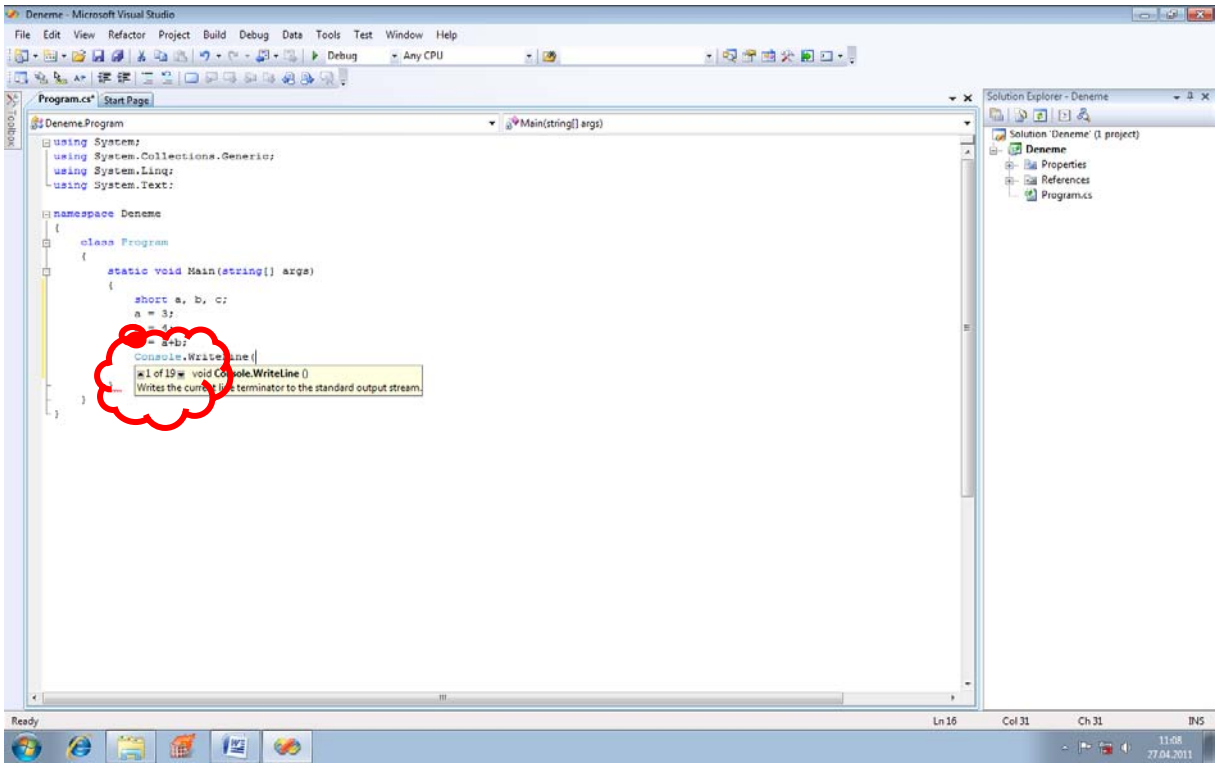
Şekil 13.

Şekil 14.te görüldüğü gibi yazılacak komutun üzerine fare gelindiğinde komutun yazılışı ve kullanımı hakkında bilgi ekrana çıkmaktadır. “Console.wr” yazıldığında dikkat edilirse istediğimiz komut belirdi, burada “(”e basılınca komutun tamamı yazılacak.



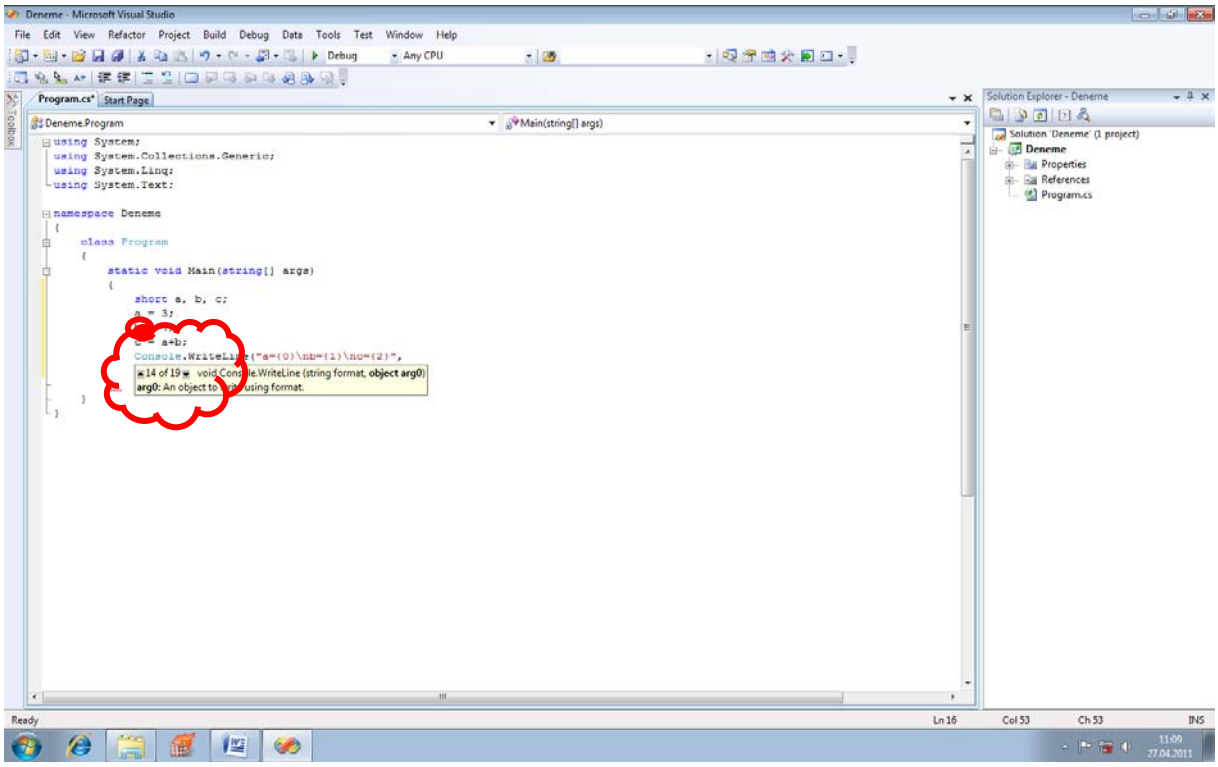
Şekil 14.

Şekil 15.te görüldüğü gibi “Console.WriteLine(.” yazıldığında bununla ilgili 19 kullanımının olduğu belirtilmektedir. Yani “Console.WriteLine(.” nın 19 çeşit kullanımı var demektir.



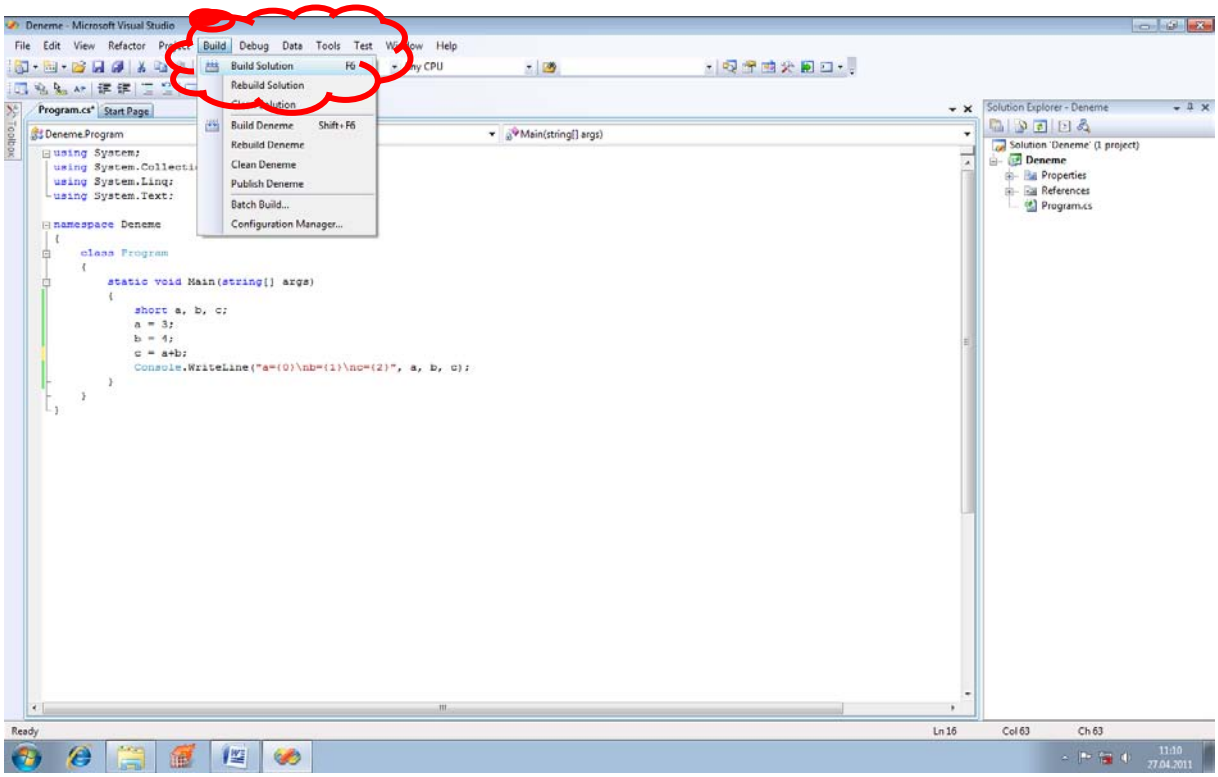
Şekil 15.

Şekil 16.da “Console.WriteLine(“a={0}\nb={1}\nc={2}”,” yazıldığında WriteLine komutunun 14-19 arasındaki kullanımlarının kullanılabileceğini anlamaktayız. Yani ekrana yazılacak kısım bittikten ve “,” konduktan sonra 5 farklı kullanım olduğu ifade edilmektedir.



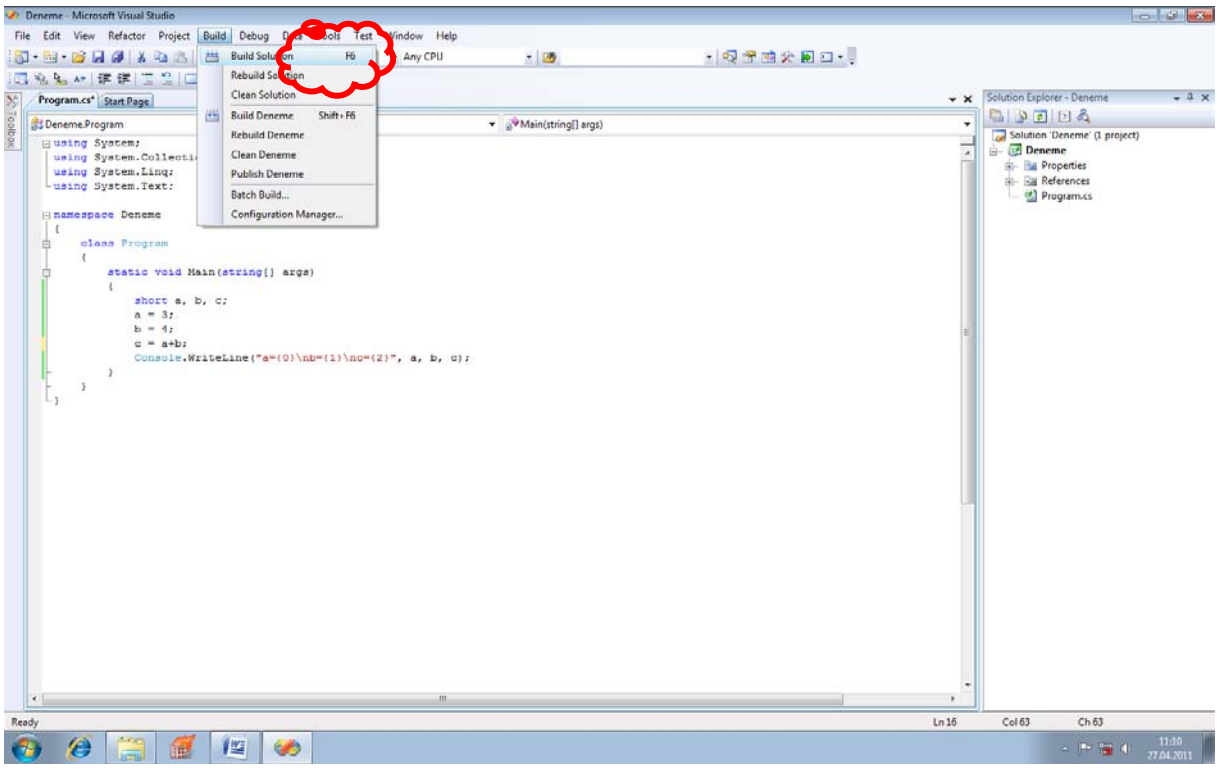
Şekil 16.

Şekil 17.de görüldüğü gibi program yazıldıktan sonra program hemen çalıştırılmaz. Hata olup olmadığı denenir. Bunun “Build” sekmesinde “Build solution” seçilir veya “F6” tuşuna basılır.



Şekil 17.

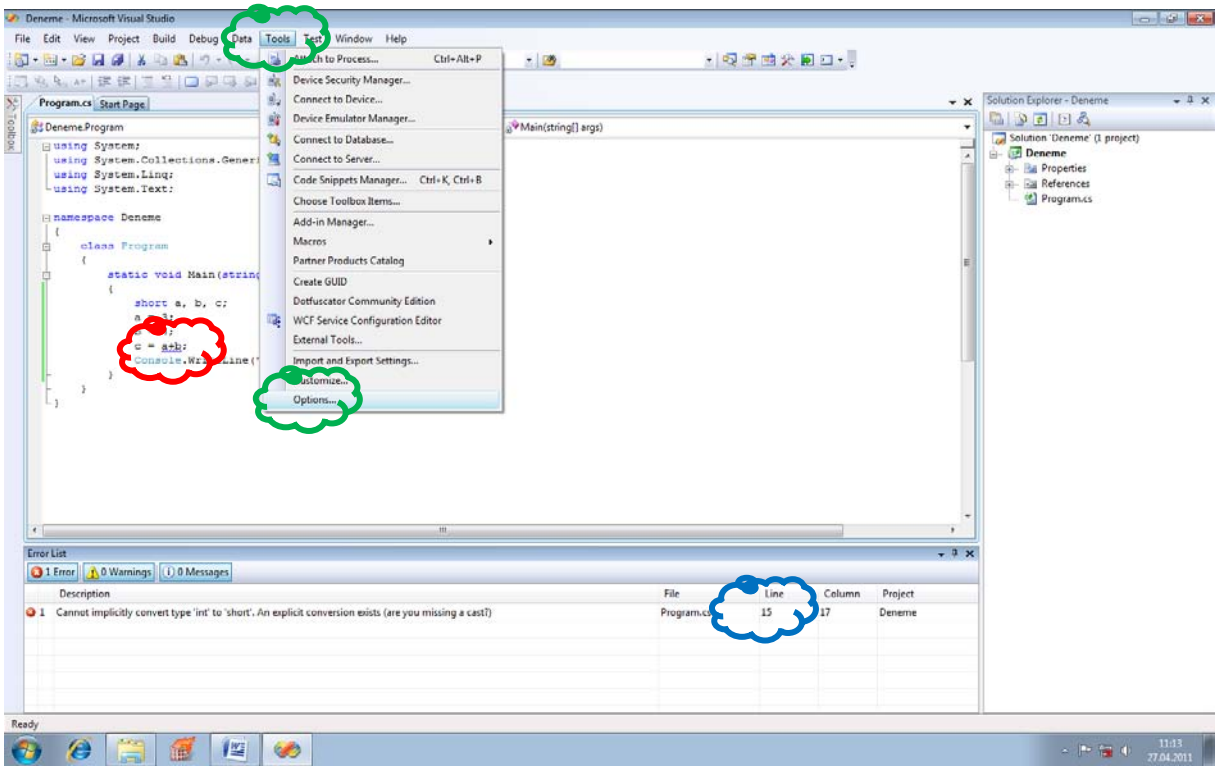
“F6” tuşu ile program derlenir.



Şekil 18.

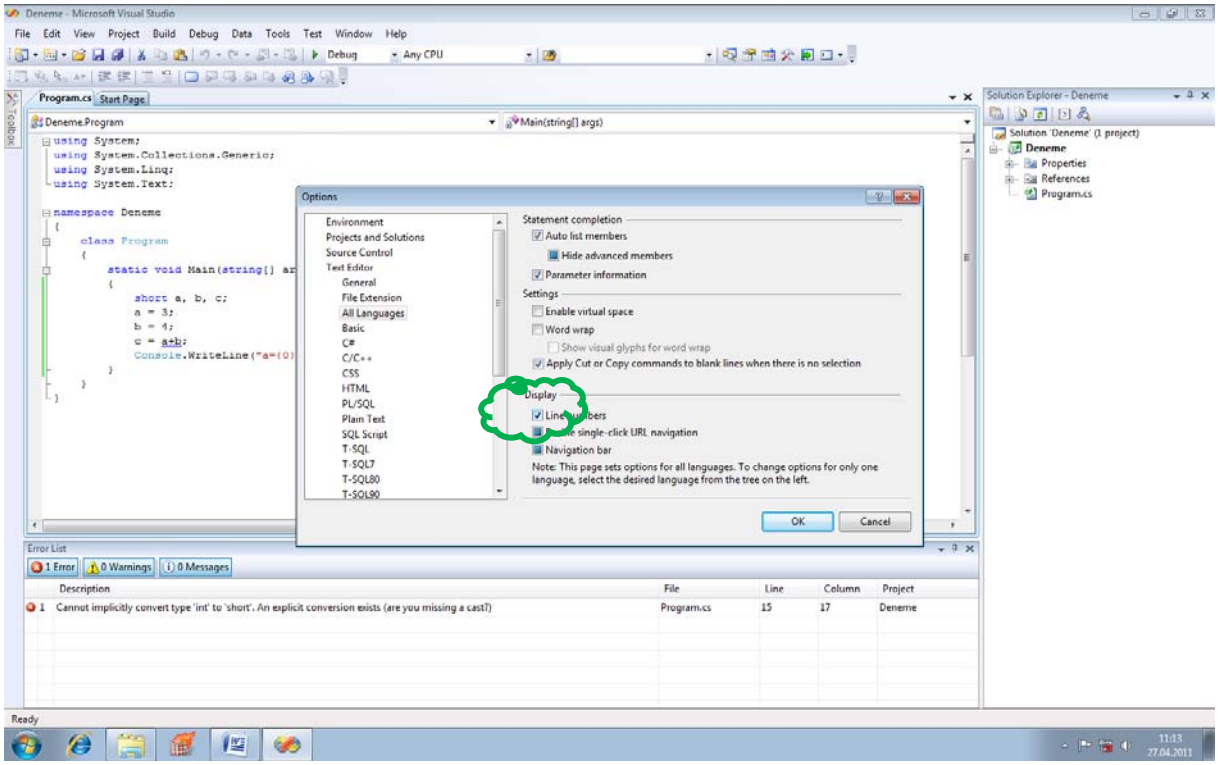
Program yazımında kırmızı dalga, sarı dalga ve mavi dalga ile üç çeşit hata bilgisi verilir. Şekil 19.da “c=a+b;” nin altı mavi dalga ile çizilidir. Burada bir hata olduğu anlaşılır. Programın alt kısmında hata mesajı görülmektedir (mavi bulut). Şayet hata satırının hangisi olduğunu kestiremiyorsak satır numarasını ortaya çıkarmak için:

Satır numaralandırma nasıl açılır: “Tools/Option” tıklanır (yeşil bulut).



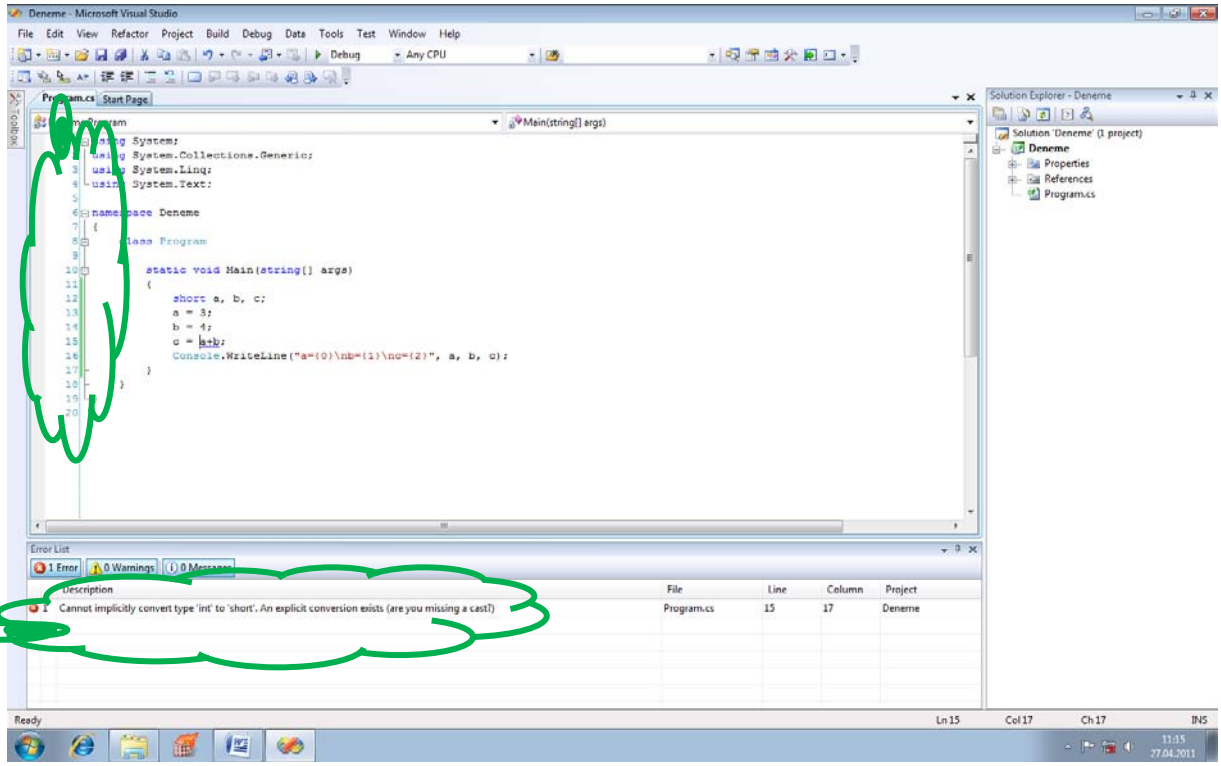
Şekil 19.

Şekil 20.de açılan yeni pencere de “Line numbers” yazısının önündeki kutucuğa tik işareti atılır. “OK” butonuna basılınca sıra numaraları görünecektir.



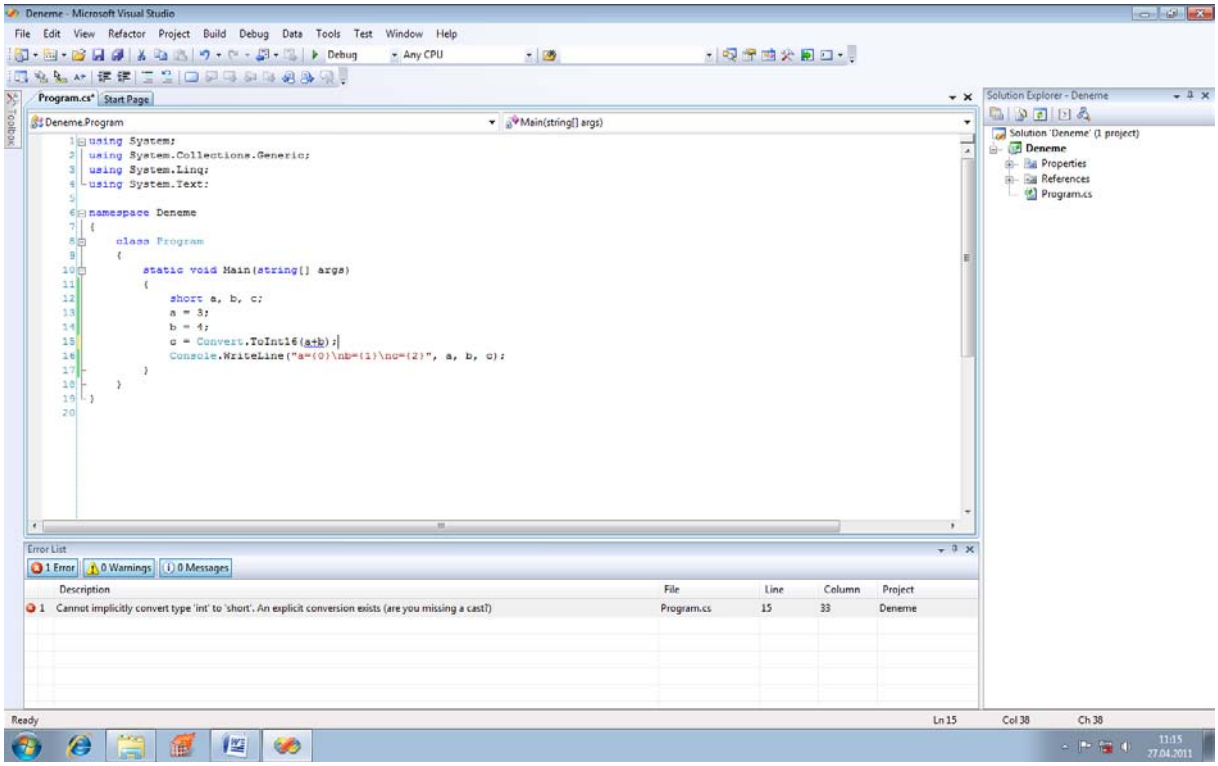
Şekil 20.

Şekil 21.de sıra numaraları görünmektedir. Hata satırına çift tıklanarak da hatanın olduğu satıra gidilebilir (yeşil bulut).



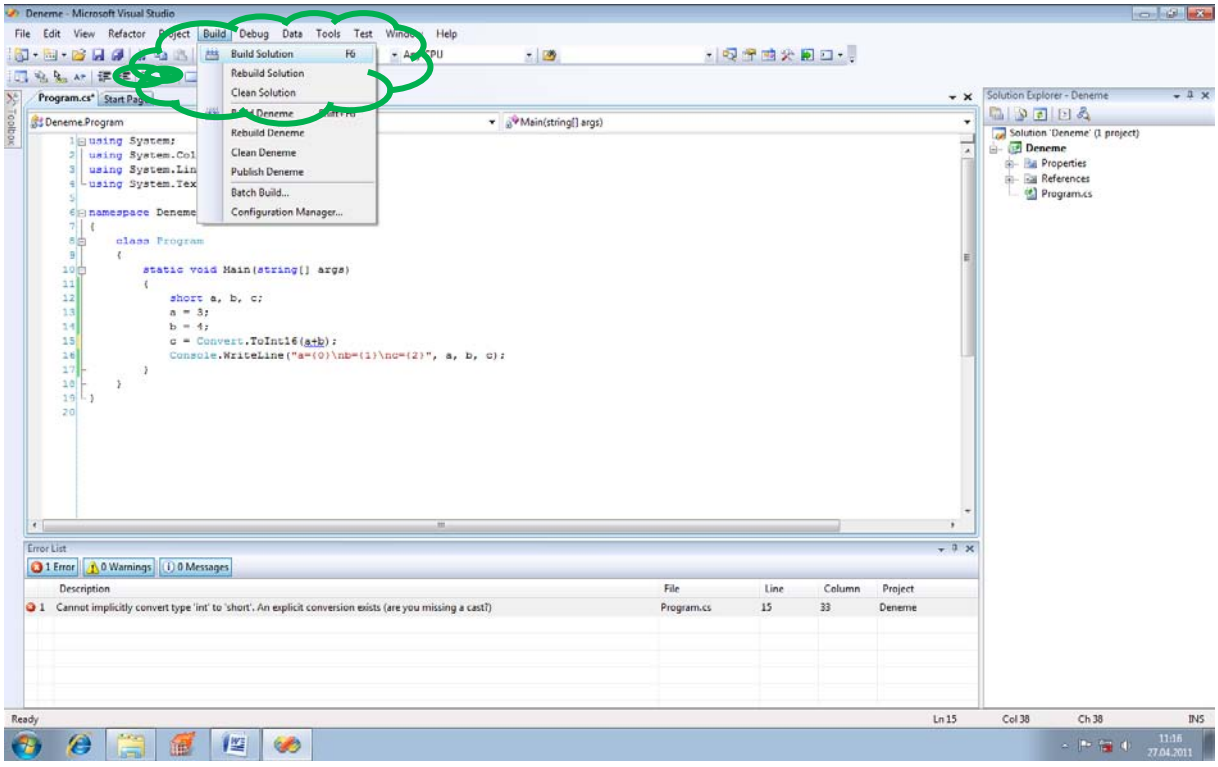
Şekil 21.

Şekil 22.de 15. satırdaki hatayı düzelttikten sonra Şekil 23. Olduğu gibi F6 ya bastığımızda



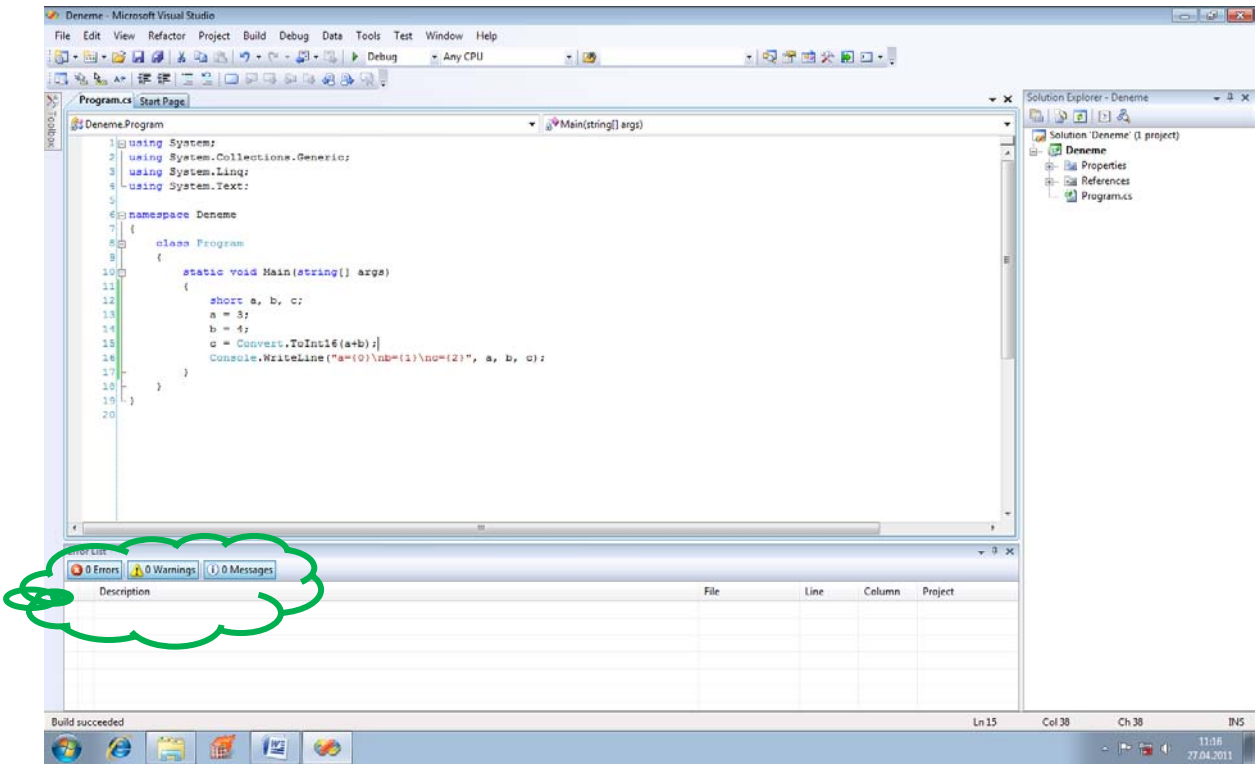
Şekil 22.

Şekil 23. F6 ile yine hata kontrolü yapılır.



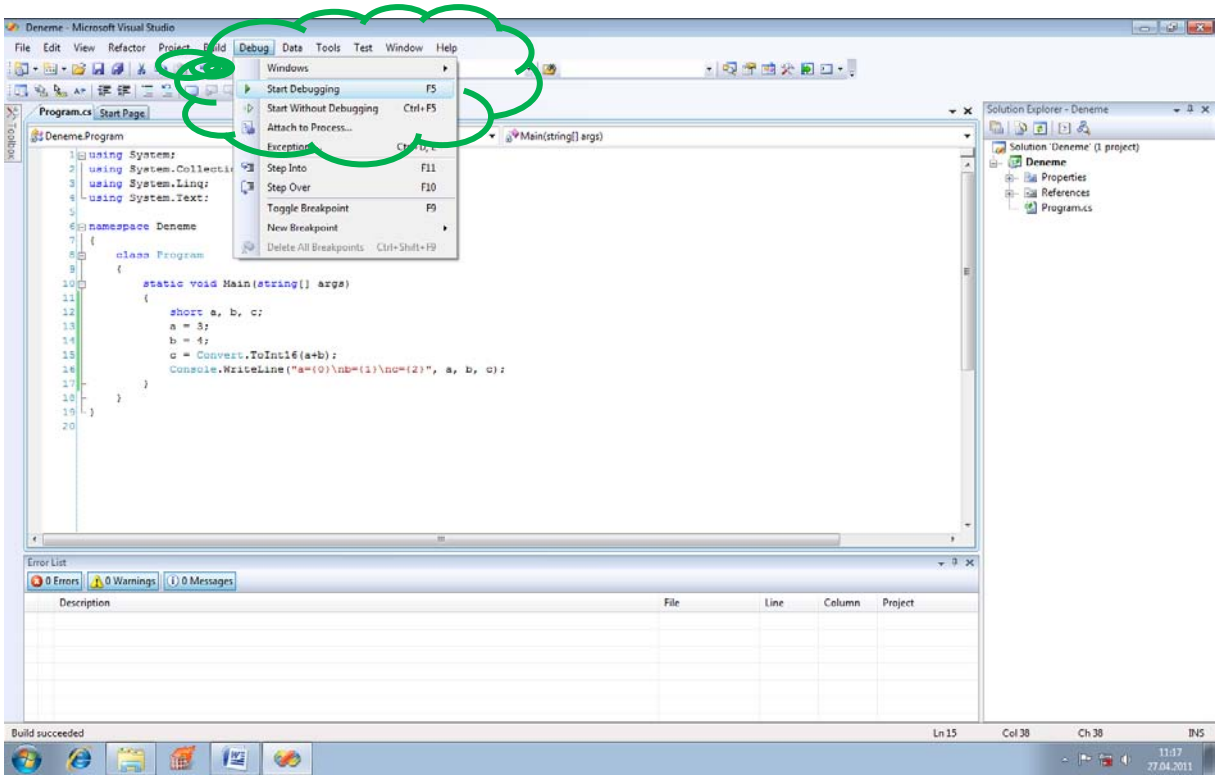
Şekil 23.

Şekil 24.te dikkat edilirse hata bulunmadı.



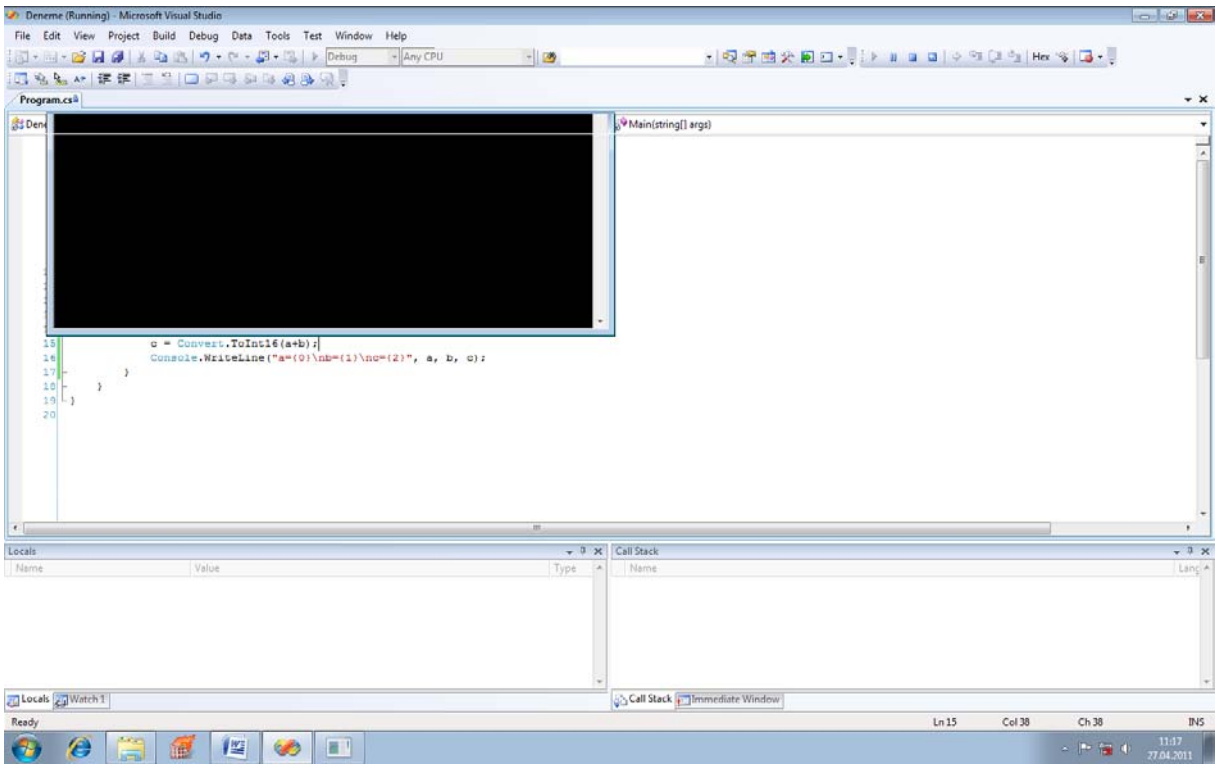
Şekil 24.

Şekil 25. Build ile hata kontrolünden sonra Debug / Start Debugging veya F5 ile program çalıştırılır.



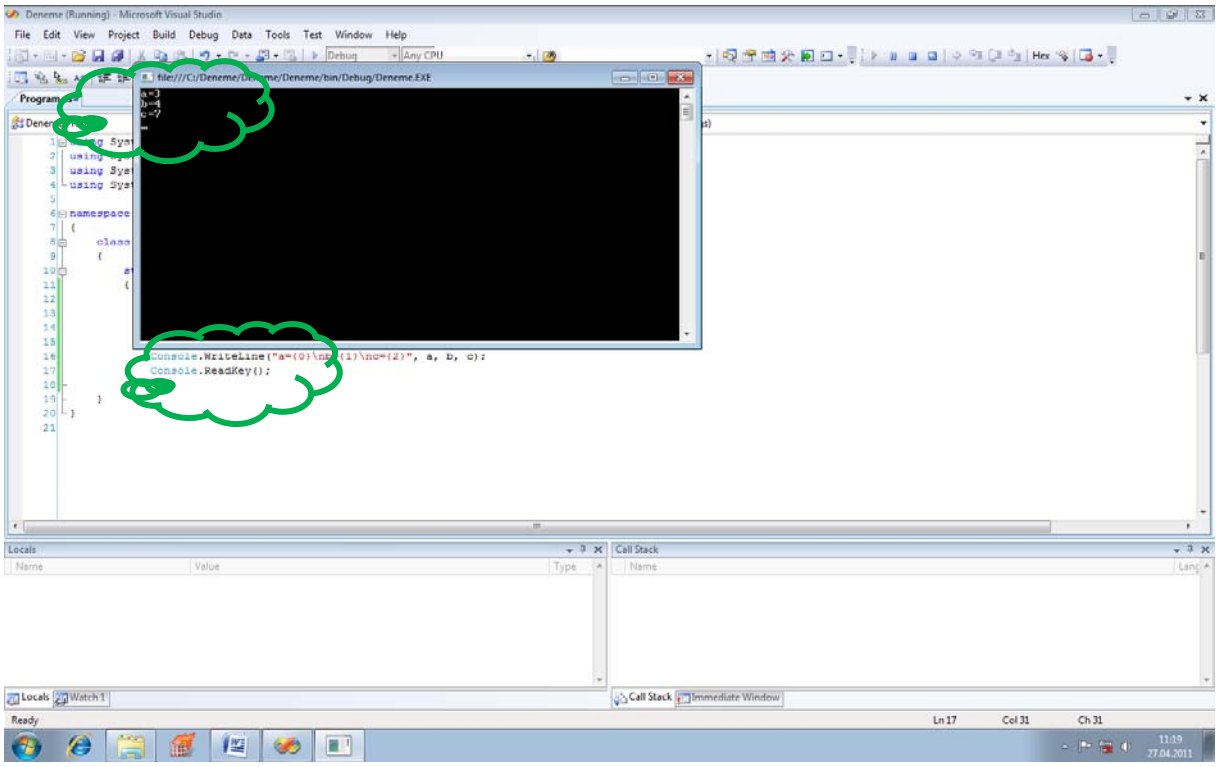
Şekil 25.

Şekil 26.da görüldüğü gibi sonuç bir anlık ekranda gözüküyor ve kayboluyor.



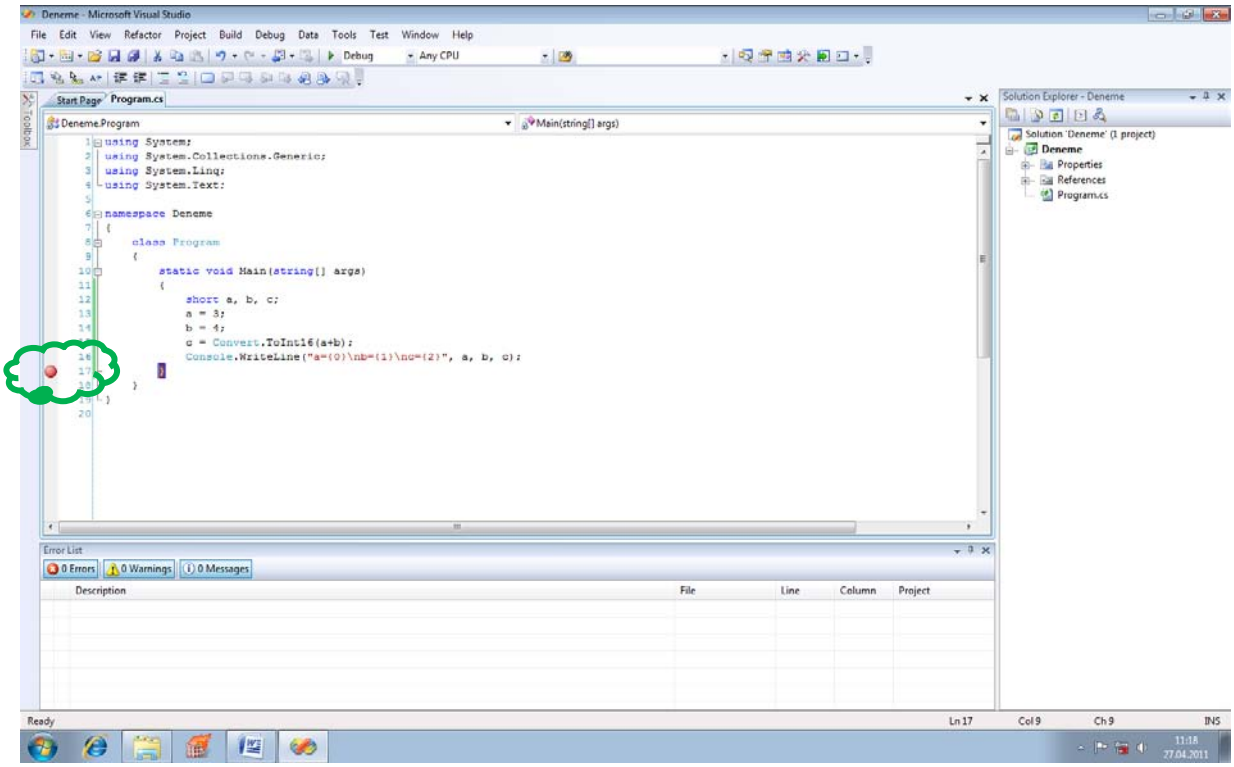
Şekil 26.

Şekil 27.de sonuçları ekranda tutmak için programın sonunda “Console.ReadKey();” yazıldığına dikkate ediniz. Bu durum bir tuşa basıncaya kadar siyah ekran kaybolmayacaktır.



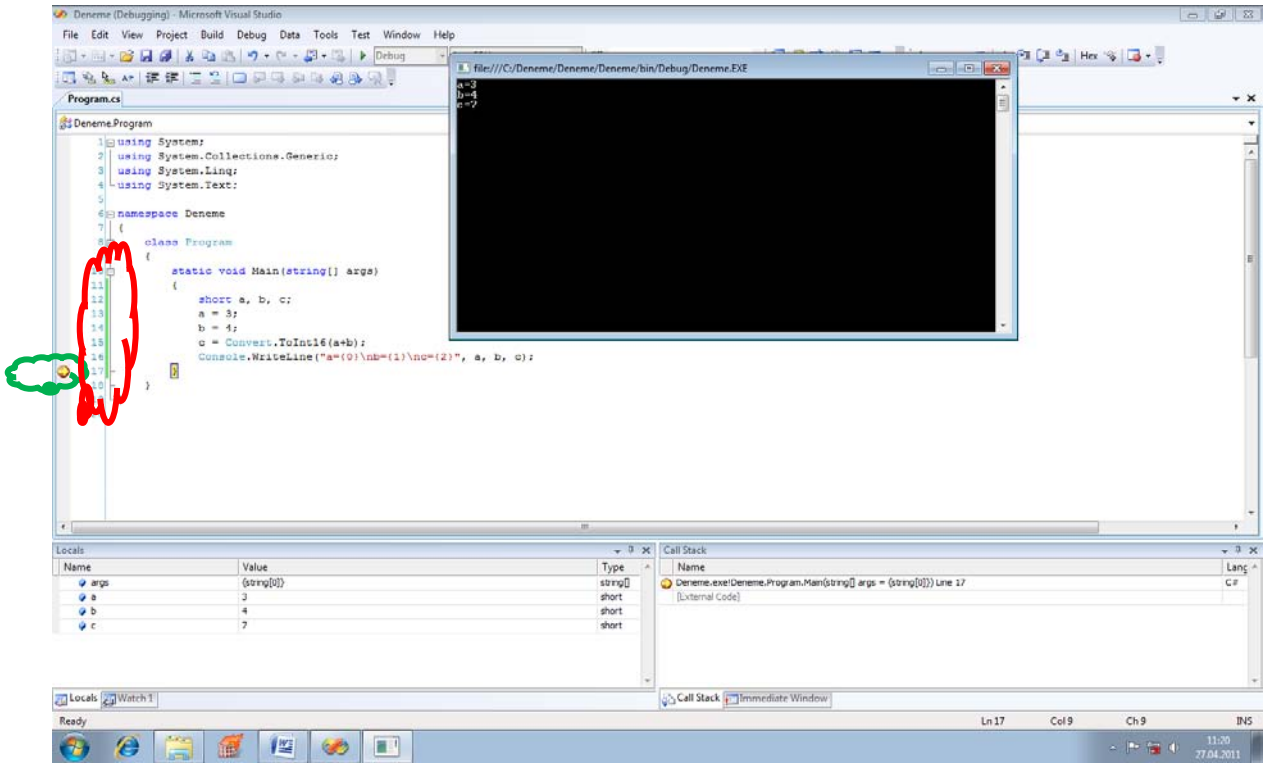
Şekil 27.

Şekil 28.de programda en sol boşluğa break point işareti konularak da program istenildiği yerde durdurulabilir. Break point konulacak yere çift tıklanınca işaret kendiliğinden gelir.



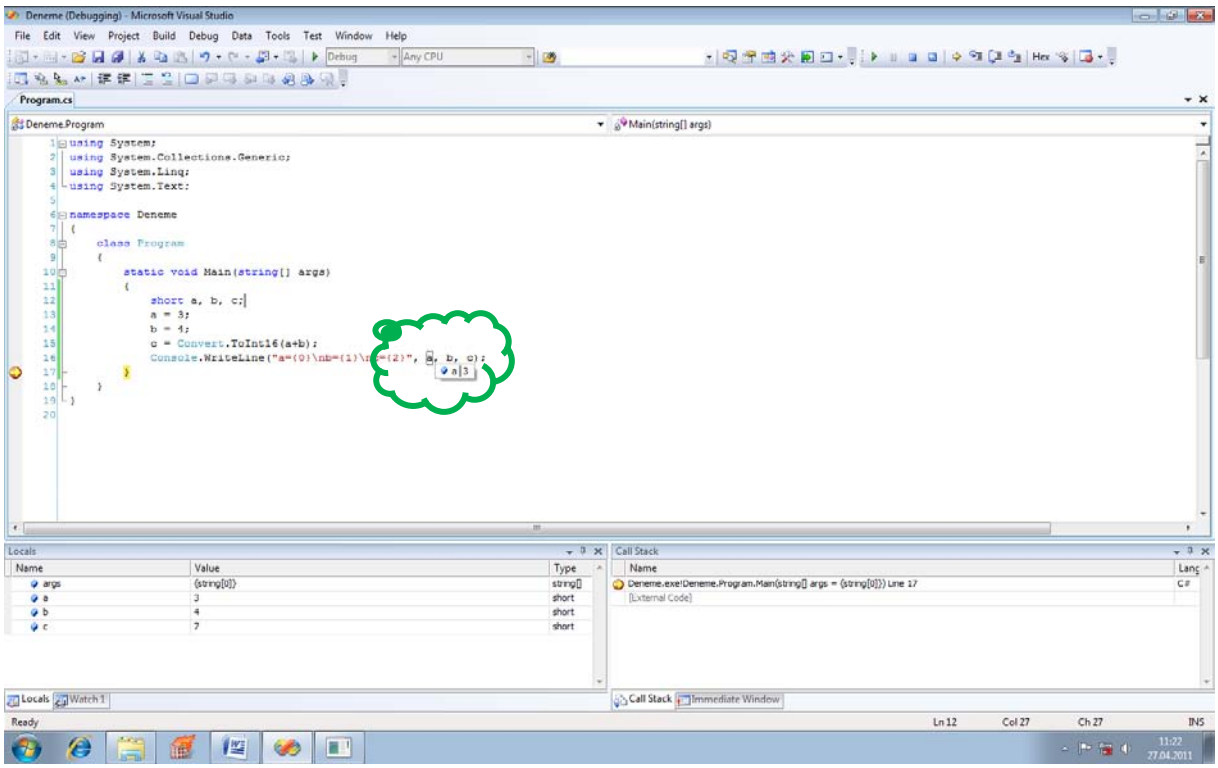
Şekil 28.

Şekil 29.da break point işareti varken F5 ile program çalıştırıldığında siyah ekran bir tuşa basıncaya kadar bekler. Kırmızı bulut ile gösterilen kısımda satır numaranın sağında dikey yeşil bir çizgi varsa build yapılmış, dikey sarı bir çizgi varda build yapılmamış olduğunu gösterir. Şuan yeşil çizgi olması build yaptığımızı gösterir .



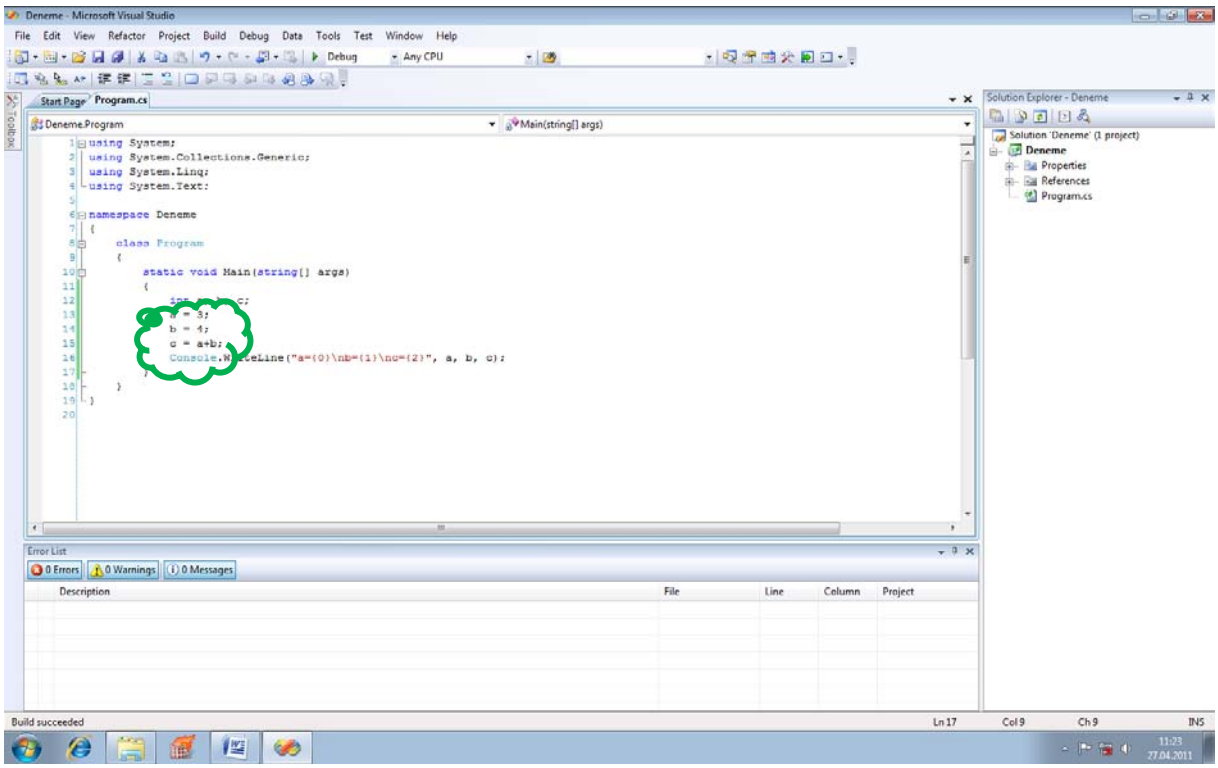
Şekil 29.

Şekil 30.da fare ile değişkenin üzerine gelindiğinde değişkenin içeriği görünür.



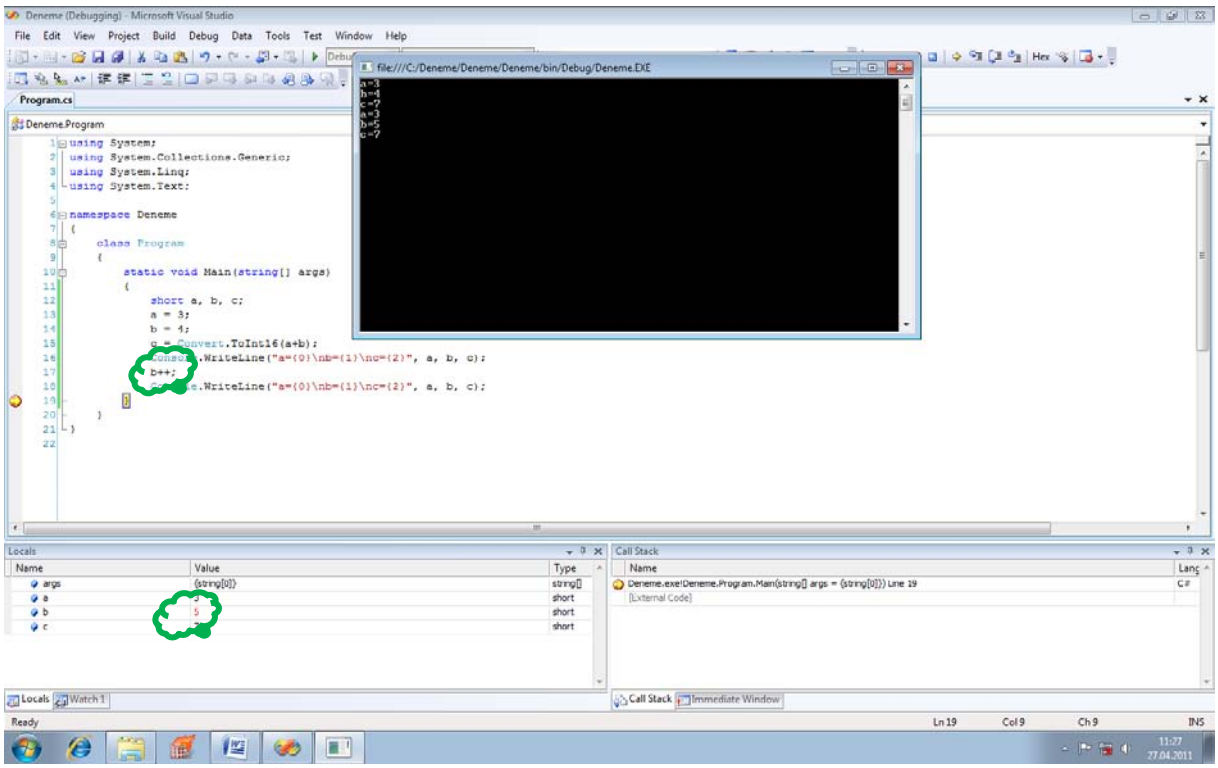
Şekil 30.

Şekil 31.de değişkenler short yerine int, long, decimal, float, double olsaydı, c=a+b; yazılabilirdi. Ancak farklı değişken olacaksa veya değişken dönüştürülecekse kullanılmak zorundadır. Mesela short=int.. veya long = short gibi.



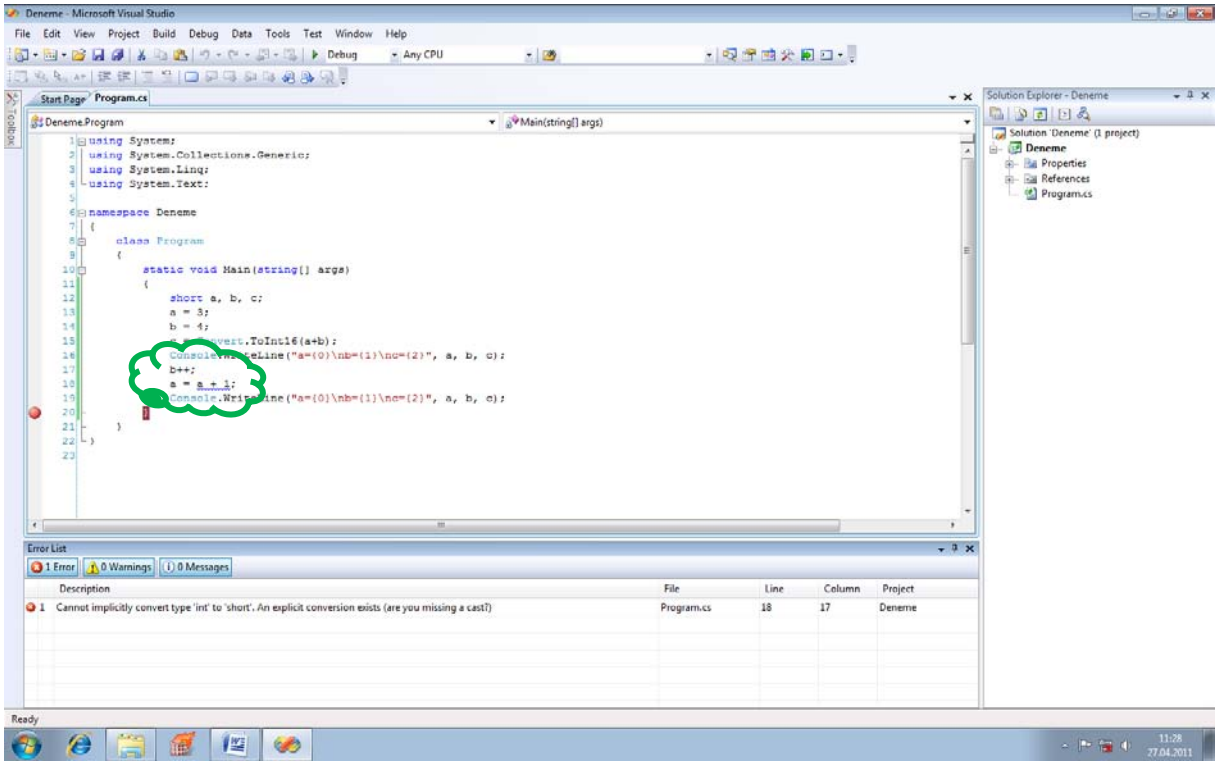
Şekil 31.

Şekil 32.de b++ artırma komutunun kullanımı görülmektedir.



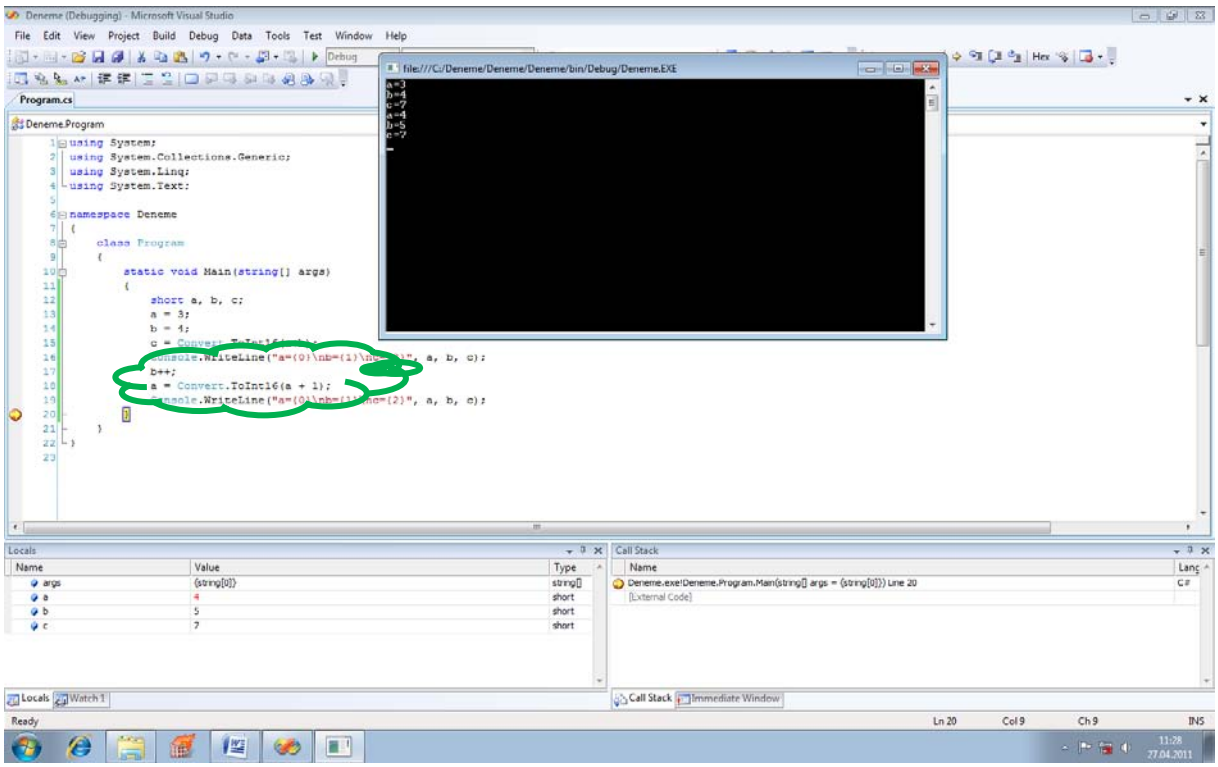
Şekil 32.

Şekil 33.te a=a+1 yazılmak istendiğinde hata verdiğine dikkat edelim. Burada a+1 convert içerisinde yazılmak istenmektedir.



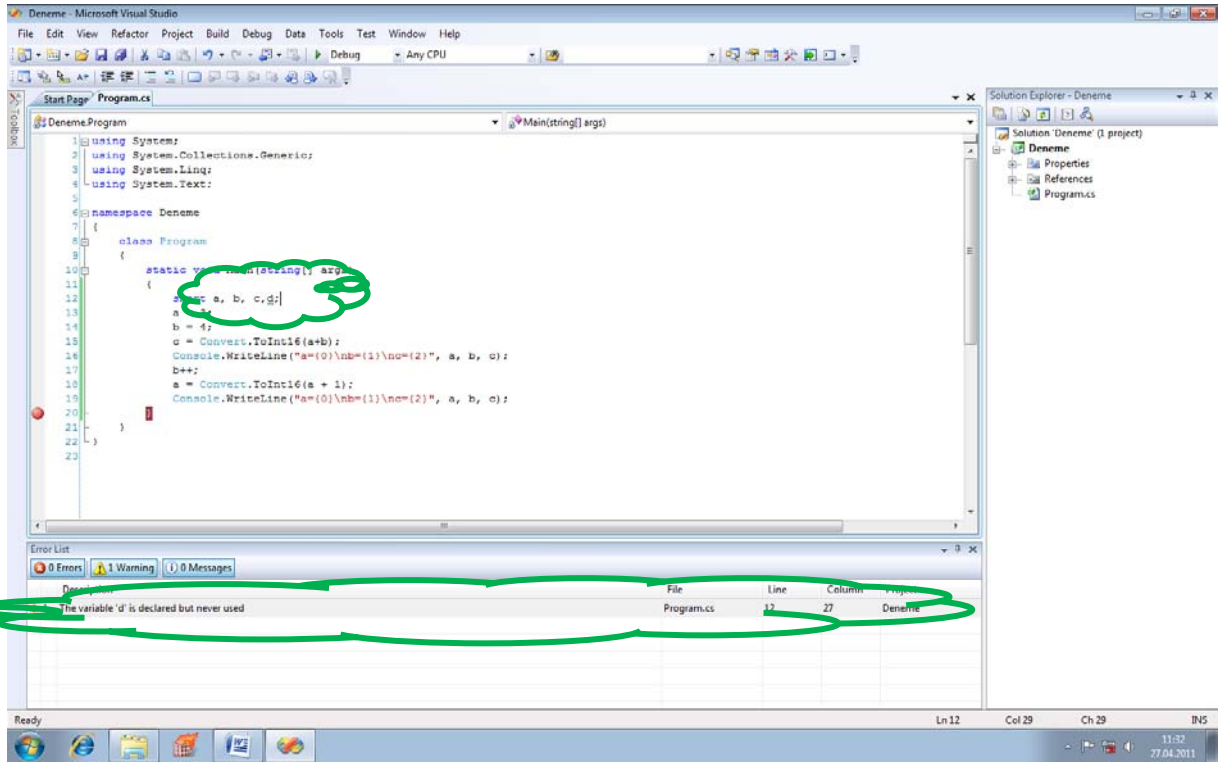
Şekil 33.

Şekil 34.te a+1 convert ile yazıldığında hata ortadan kalkmış olur. Fakat daha uzun bir komut yazmış olduk. Dolayısıyla b++ yazmak daha mantıklıdır.



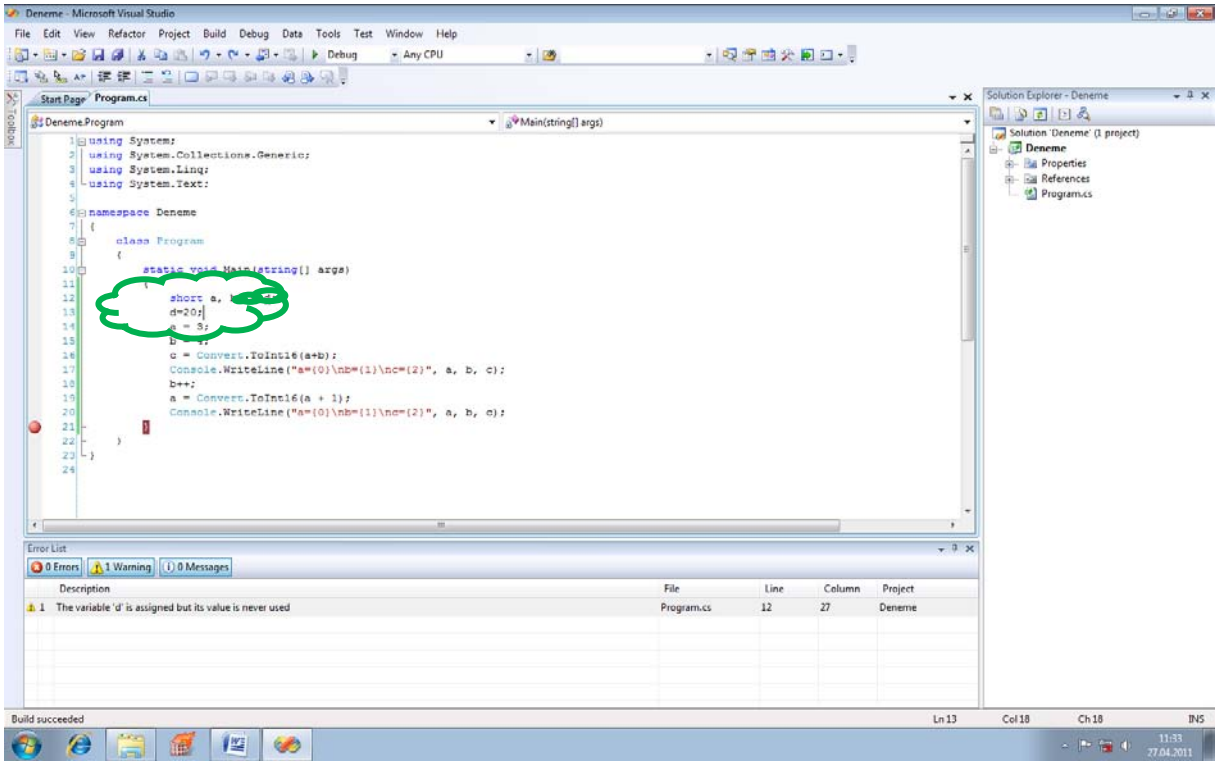
Şekil 34.

Şekil 35.te Programa short türünde d değişkeni ekleyip F6 ile build yaptığımızda hata ile karşılaşırız. d değişkenin altı yeşil dalga ile işaretli olduğuna dikkat edelim. Burada hata d yi herhangi bir yerde kullanmadığımızdan kaynaklandı.



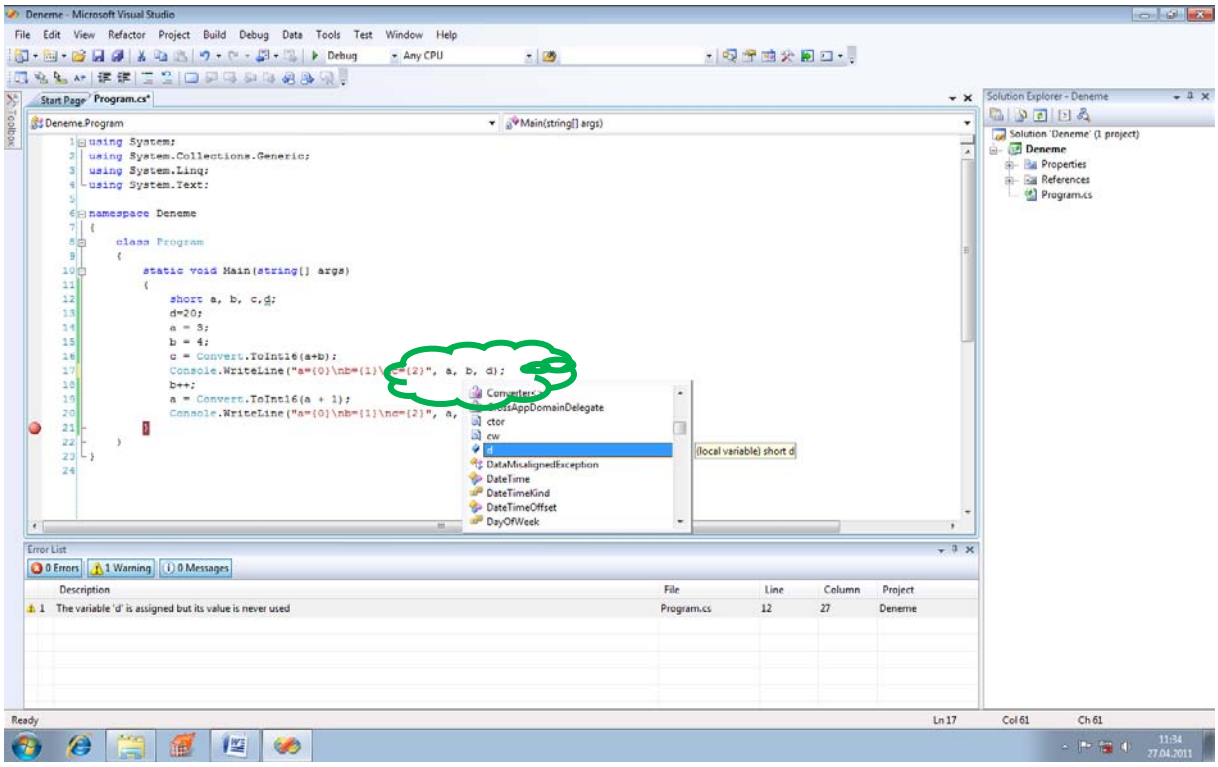
Şekil 35.

Şekil 36.da “d =20;” satırı programa eklenip ve F6 ile tekrar build yapıldığında yine aynı hata ile karşılaşırız. Çünkü d yi henüz kullanmadık. Sadece tanımladık ve içine 20 atadık.



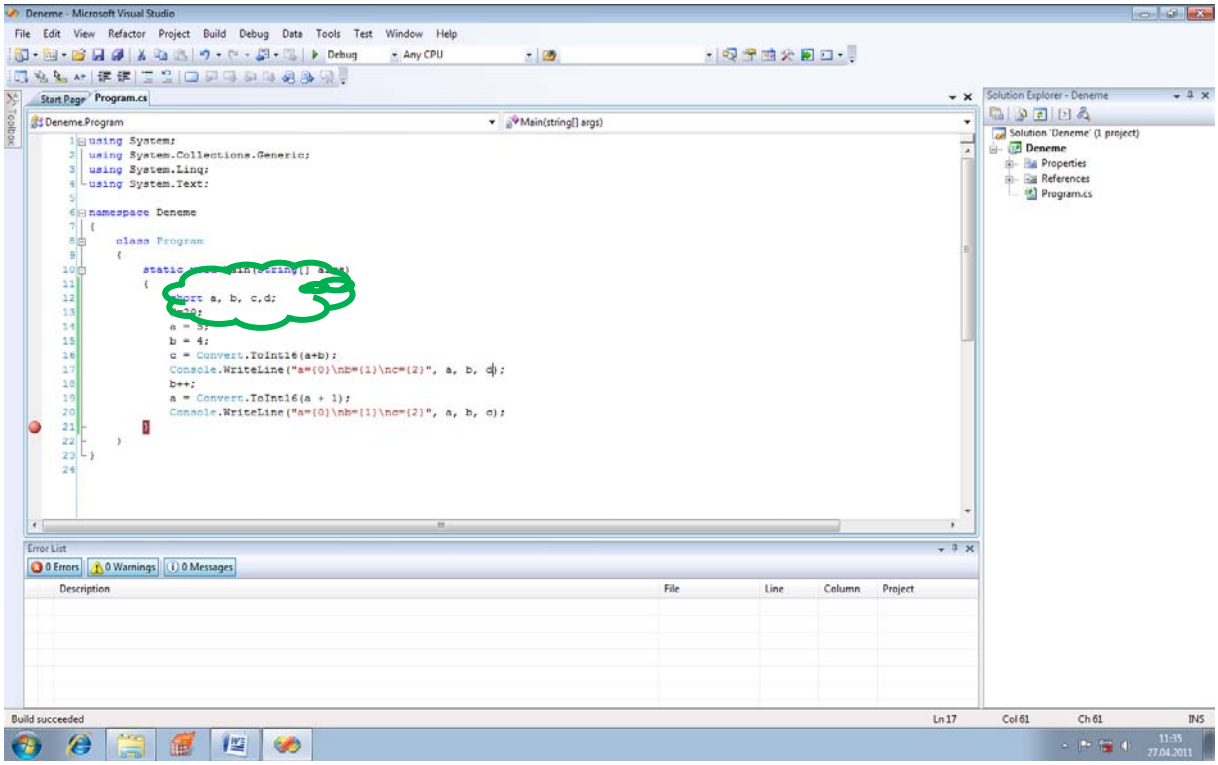
Şekil 36.

Şekil 37.de görüldüğü gibi Console.WriteLine komutunun içine c yerine d yazılınca artık d değişkeni kullanılmış oldu.



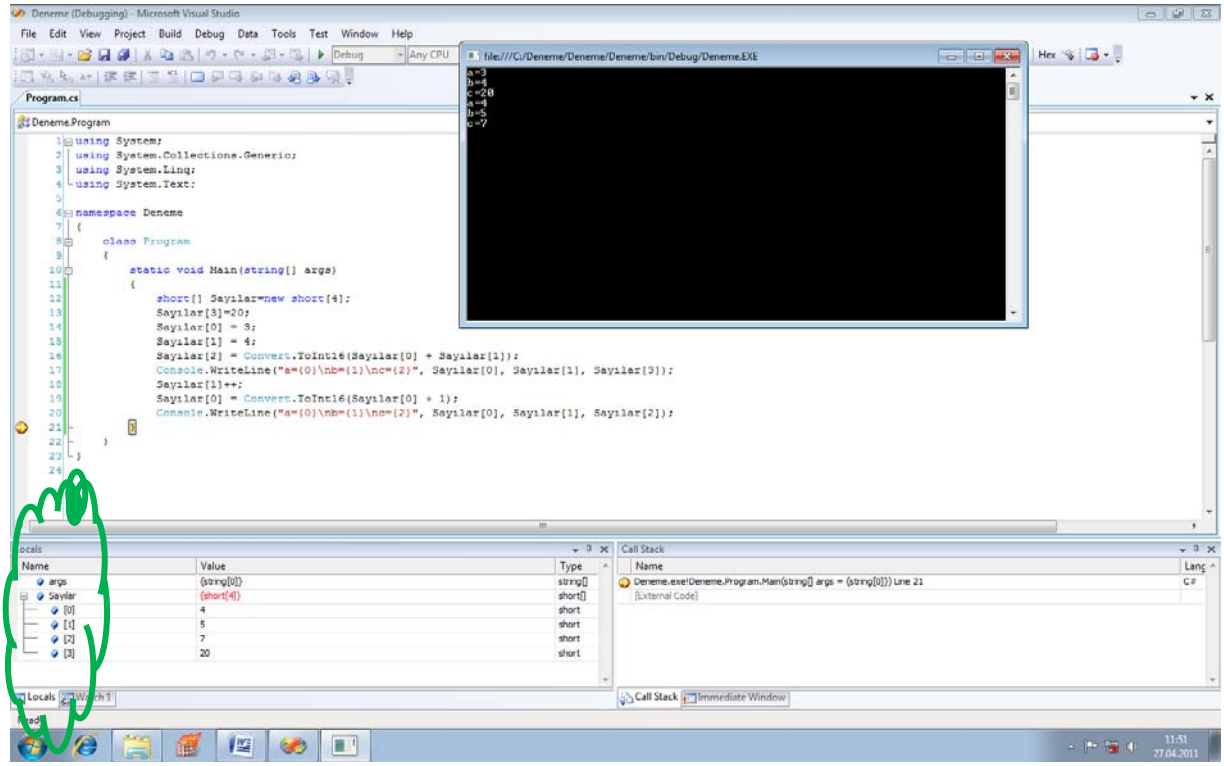
Şekil 37.

Dolayısıyla Şekil 38.de görüldüğü gibi program F6 ile build yapıldığında d nin atındaki yeşil dalga kaybolacaktır. Yani hata düzeltilmiş olacaktır.



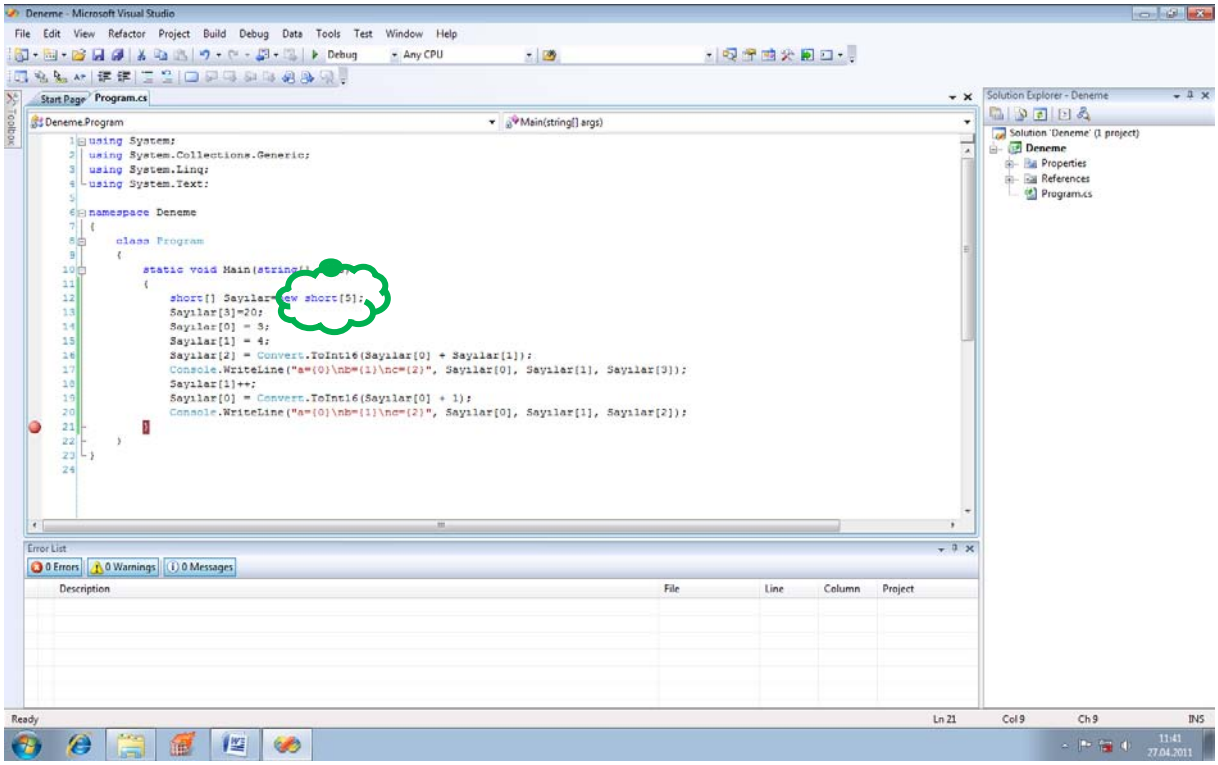
Şekil 38.

Değişken dizi tanımlandığında dikkat edilirse değişken adı altında



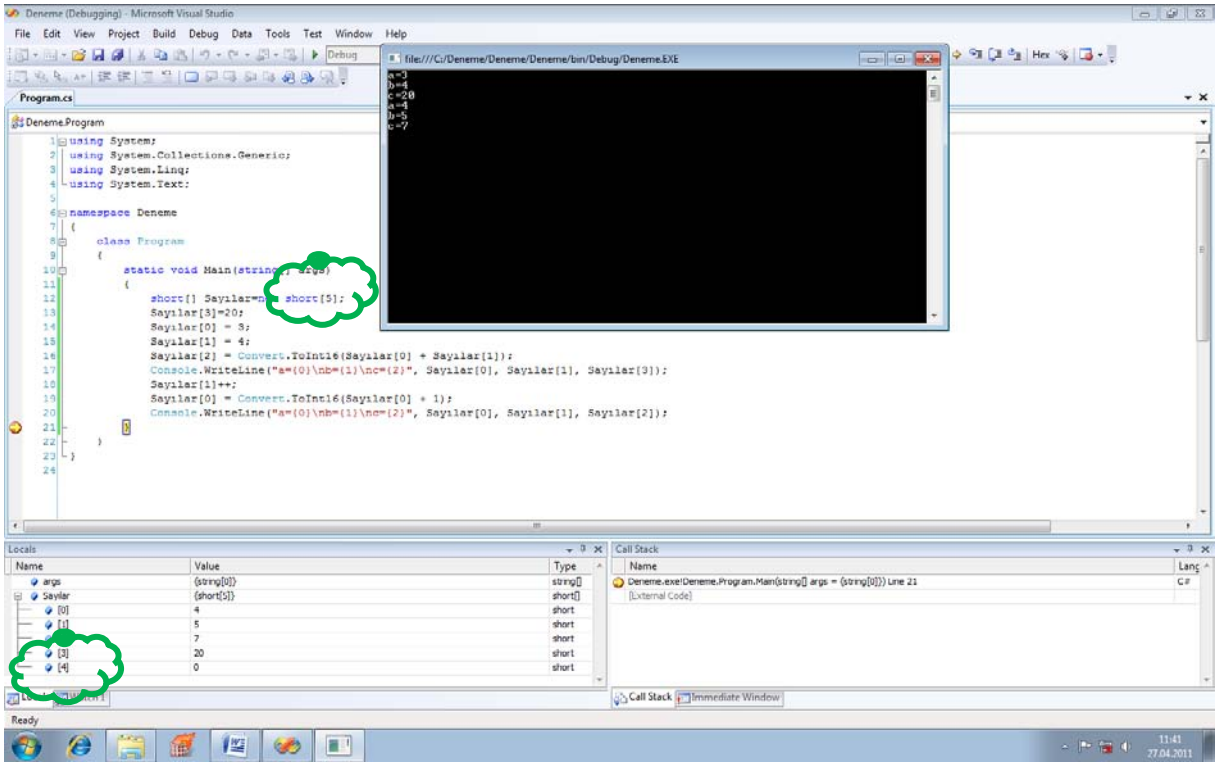
Şekil 39.

Şekil 40.ta değişken dizinin boyutu değiştirildiğinde değişkenlerden biri boş olsa bile değişken kullanıldığı için program hata vermeyecektir.



Şekil 40.

Şekil 41.de görüldüğü gibi değişkenin son değeri sıfır kabul edilmiştir.



Şekil 41.

Program nasıl yazılır:

Adım 1: Ekrana neler geleceği belirlenir.

Adım 2: Adım 1'e uygun olarak akış diyagramı çizilir.

Adım 3: Adım 2'ye uygun olarak akış diyagramı birkaç parçaya bölünerek C# programı yazılır.

Adım 4: Adım 2'ye uygun olarak Adım 3'de yazılmış programın tüm çalışma olasılığı kontrol edilir. Kullanıcının yapabileceği hatalar göz önüne alınarak C# programın çalışması kontrol edilir.

Önemli not: Hiçbir zaman C# programı tamamen yazılıp denenmez. Akış diyagramı uygun parçalara ayrılarak parça parça yazılarak/denenerek yazılır.

Önemli not: Hiçbir zaman C# programı sorunun cevabını bilemez! Sadece bizim yazdığımız programın sonucunu verir. C#'ın verdiği sonuç bizim önceden belirlediğimiz sonuçla aynı ise programı doğru yazdığımız anlamındadır. Fakat önceden bir sonuç belirlemediğimizde C#'bın verdiği cevabı doğrulayamayız ve kesinlikle doğru kabul edemeyiz.