Algoritmalar ve Programlama I – BLM 103

Hafta 8: Test ve Hata Ayıklama

Fenerbahçe Üniversitesi

8. Hafta İçeriği

- Hata Türleri
  - Sözdizimi (Syntax)
  - Anlamsal (Semantic)
  - Algoritmik
- Hata Ayıklama
- Visual Studio Hata Ayıklama Araçları

Hata Türleri

- Sözdizimsel (Syntactic) Hatalar
  - Derlenecek olan kodda programlama dilinde tanımlı olmayan ifadeler veya beklenen bazı şeyler eksiktir.
  - Derleyici tarafından yakalanır.
  - Derlenme işlemi gerçekleştirilmez.

- Örn.

```
printf("abc);
```

printf'in sonundaki " işareti eksiktir.

Hata Türleri

- Anlamsal (Semantic) Hatalar
  - Kodda sözdizim hatası yoktur.
  - Programlayan kişinin, yazmak istediği kod değildir.

  Örn.

  ```
  for(int i=0;i<=100;)
  ```

  Programcı i'i bir arttırmayı unutmuştur. Aslında yazmak istemektedir.

Hata Türleri

- Algoritmik Hatalar
  - Programın çözüm mantığındaki hatalardır.
  - Programcı yazmak istediği kodu doğru yazmıştır ancak
  - Yazılan kod çözülmesi gereken problemi çözmemektedir.

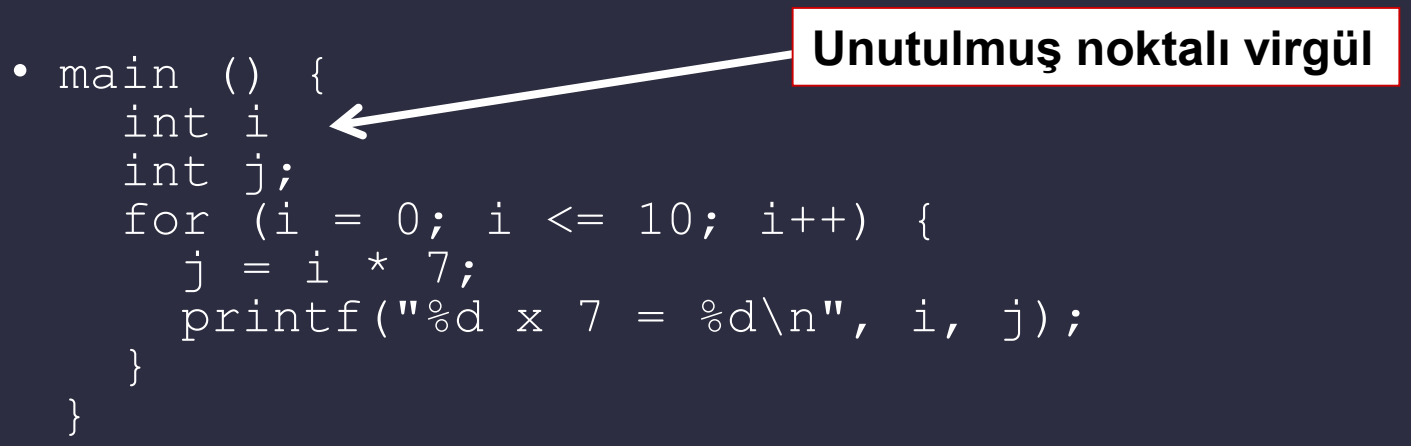
  Örn.

  - Üç sayının en büyüğünü alan kodu yazması beklenirken,

  en küçüğünü hesaplayan kodun yazılması gibi…

Sözdizim (Syntactic) Hataları

- En sık karşılaşılan hata türüdür.
  - Unutulmuş noktalı virgül veya parantez hataları
  - Yanlış yazılmış veri türleri, flot gibi
- Sadece bir yanlış, derleyicinin birden çok hata vermesine neden olabilir.



Anlamsal (Semantic) Hatalar

- Genel Hatalar
  - Süslü parantez yazmanın unutulması
  - Eşitlik kontrolü yerine (==) atama işareti konması (=)
  - Operatör önceliklerinin yanlış düşünülmesi
  - For döngüsündeki yanlış konan sınırlar
  - İlk ataması yapılmamış değişkenler

Unutulmuş Süslü parantezler

```
main () {
   int i
   int j;
   for (i = 0; i <= 10; i++)
     j = i * 7;
     printf("%d x 7 = %d\n", i, j);
}
```

Algoritmik Hatalar

- Tasarım yanlıştır.

Algoritma doğru problemi çözmez

- Tespit edilmesi zordur
  - Program, programcının istediği işi yapar.
  - Ancak programcı, çözülmesi gereken problemi anlamamıştır.

- Çözmesi zor olabilir.
  - Tekrar tasarım ihtiyacı doğrumaktadır.

Hata Ayıklama Teknikleri

- İlkel Yaklaşım
  - Programın akışı esnasında çeşitli yerlere printf yerleştirerek, akışı izlemek
  - Avantajı:
    - Hata ayıklama uygulamasına ihtiyaç yoktur
  - Dezavantaj:
    - Kod ve beklenen değerler hakkında oldukça bilgili olmak gerekir.
    - Kodu yeniden derleme ve çalıştırmak gerekir.
    - Eklenen printf'lerde hata yapılabilir.

Hata Ayıklama Teknikleri

- Hata Ayıklayıcı Kullanmak
  - Visual Studio'nun içindeki hata ayıklama araçları kullanılabilir.

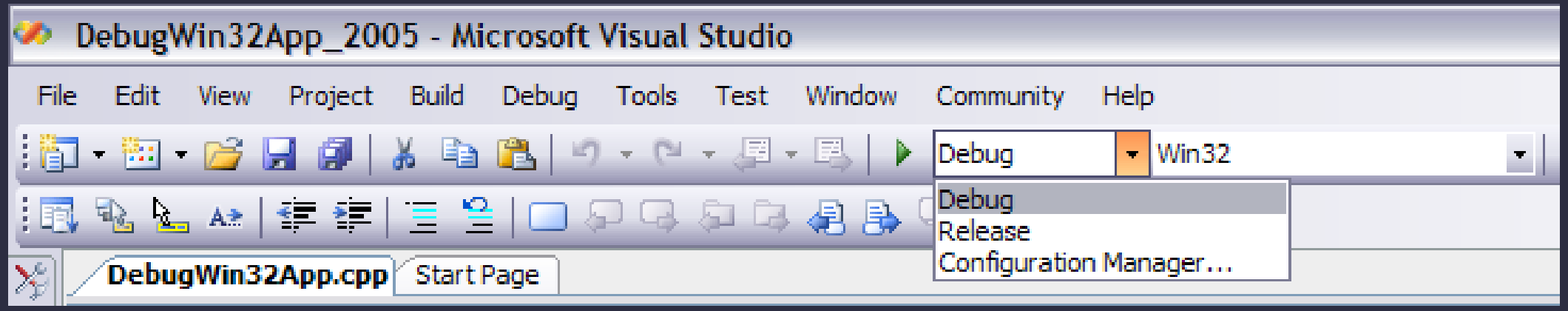
Hata Ayıklama Teknikleri

- Visual Studio iki adet çalışma zamanında hata giderme özelliğine sahiptir.
  - Programın akışını takip etmek
  - Program çalışırken değişkenlerin değerlerini izlemek

- Bu araçlar, program çalışırken belli bir yerde durdurup, değişkenlerin değerlerini okumak hatta onları değiştirmek için çok faydalıdır.

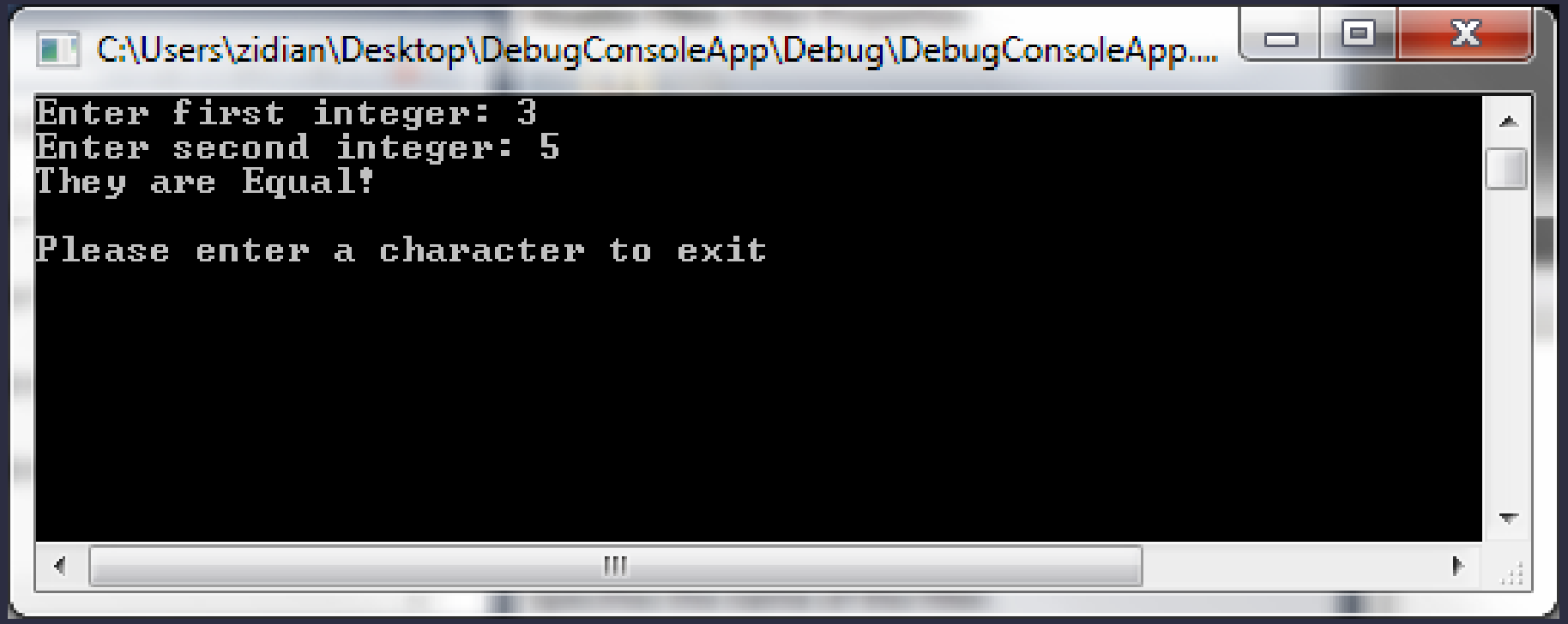
Hata Ayıklama Teknikleri

- Visual Studio'da program çalıştırılırken 2 mod bulunur.
  - Debug: Hata ayıklama için kullanılmaktadır. Program hata ayıklama modunda iken daha yavaş çalışmaktadır.
  - Relase: Programın hata ayıklama süreci tamamlandıktan sonra, diğer kullanıcılar ile paylaşmak istendiğinde kullanılmaktadır.



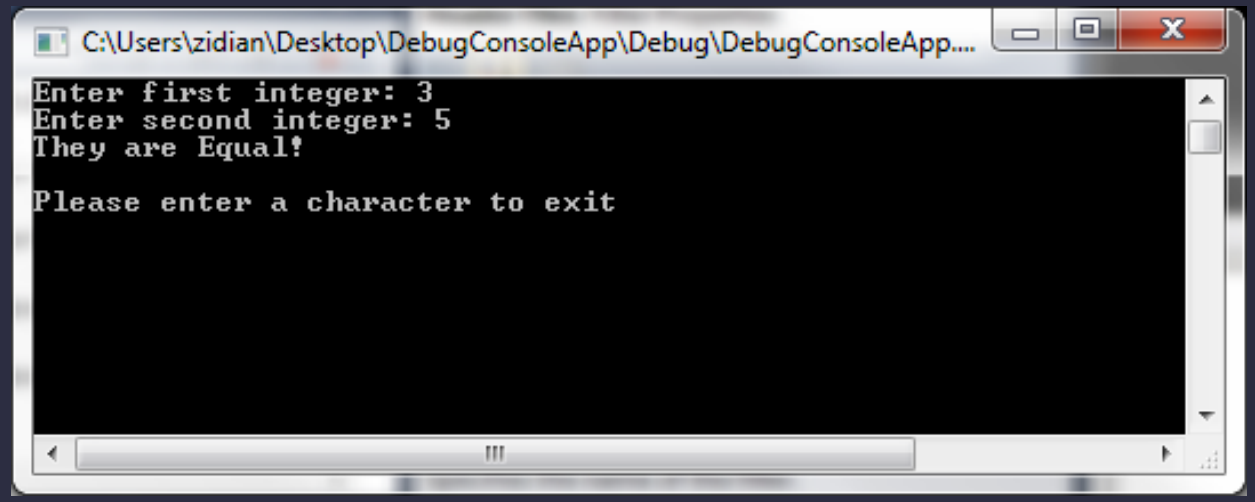
Hata Ayıklama Teknikleri

- Örn. Kullanıcıdan iki sayı alın ve eşit olup olmadığını kontrol eden bir program yazınız.



Hata Ayıklama Teknikleri

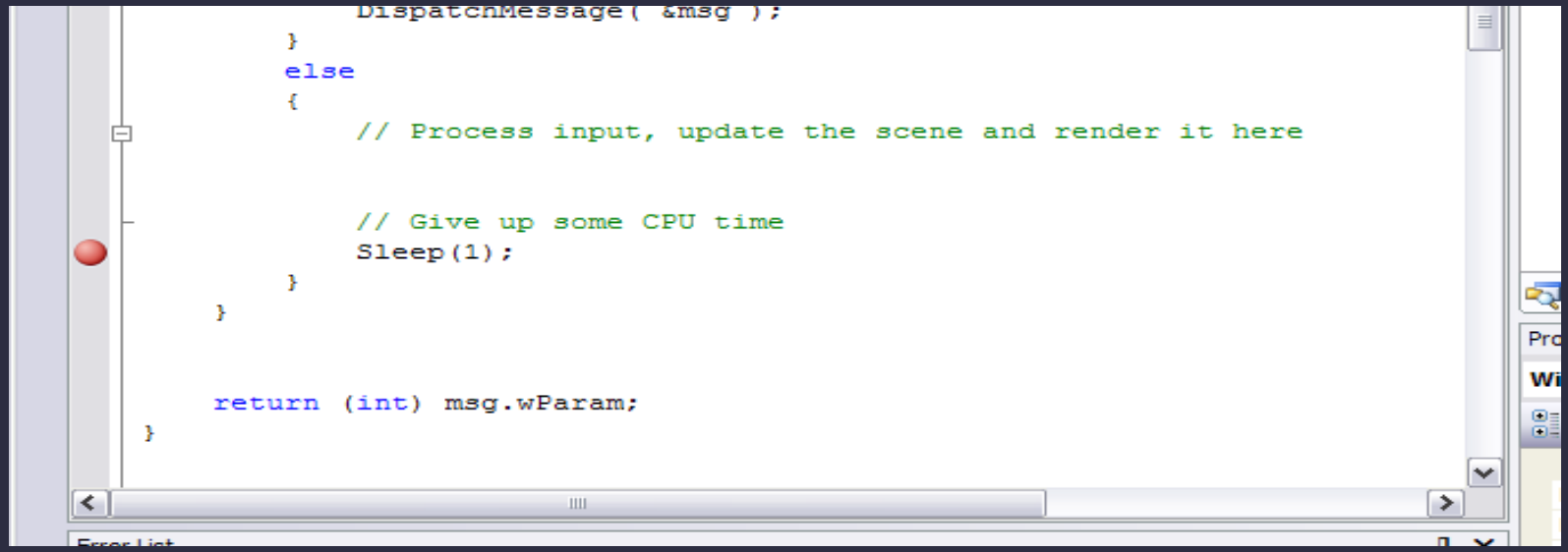
- Örn. Kullanıcıdan iki sayı alın ve eşit olup olmadığını kontrol eden bir program yazınız.



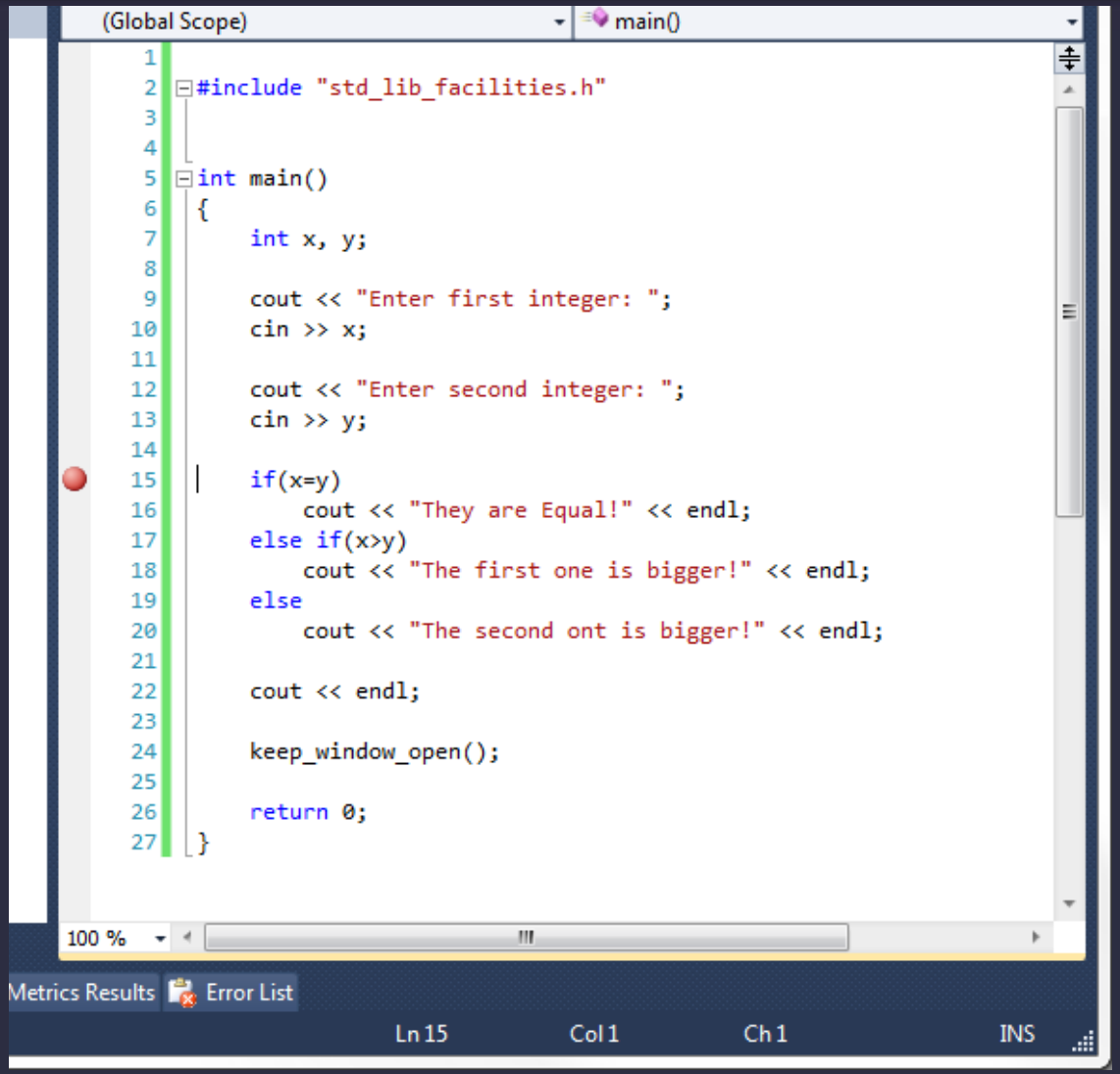
- Derleyici 0 hata, 0 uyarı verdi, ancak program yanlış çalışıyor.

Hata Ayıklama Teknikleri

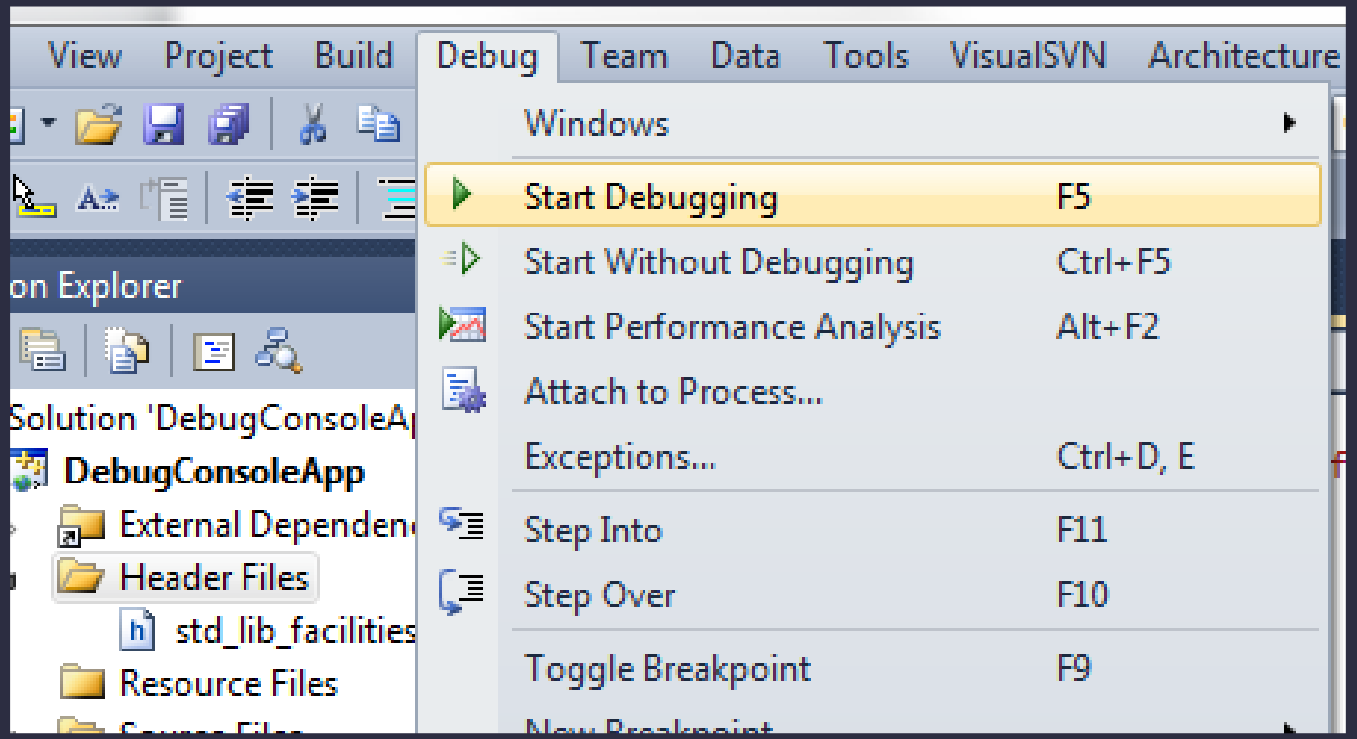
- Duraklama Noktası (Breakpoint)
  - Program çalışırken uygulamanın duraklamasını istediğiniz satırın seçilmesidir.
  - Duraklamasını istediğiniz satırın yanındaki yere tıkladığınızda kırmızı bir yuvarlak oluşur. Programı debug modunda çalıştırdığınızda o satıra gelip uygulama duracaktır.



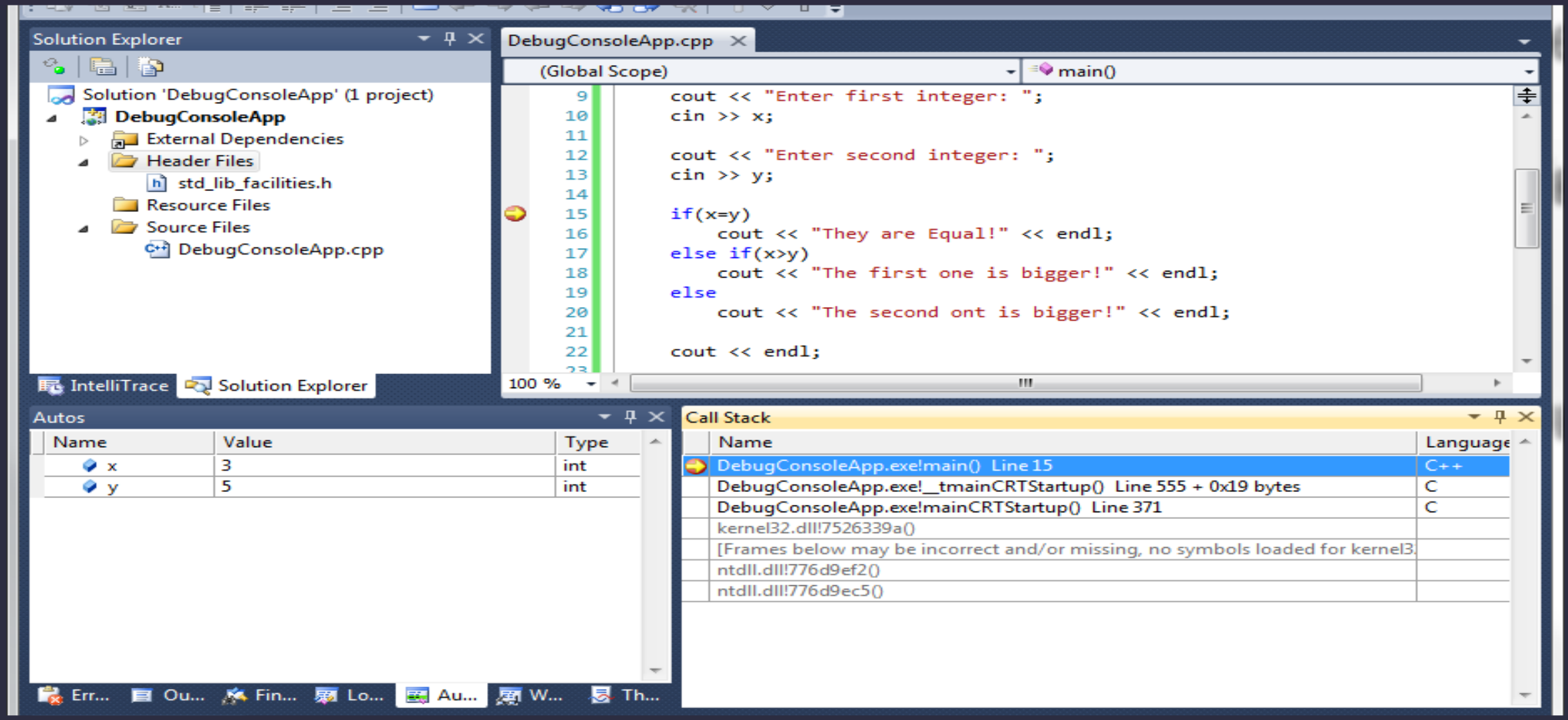
Hata Ayıklama Teknikleri



Hata Ayıklama Teknikleri

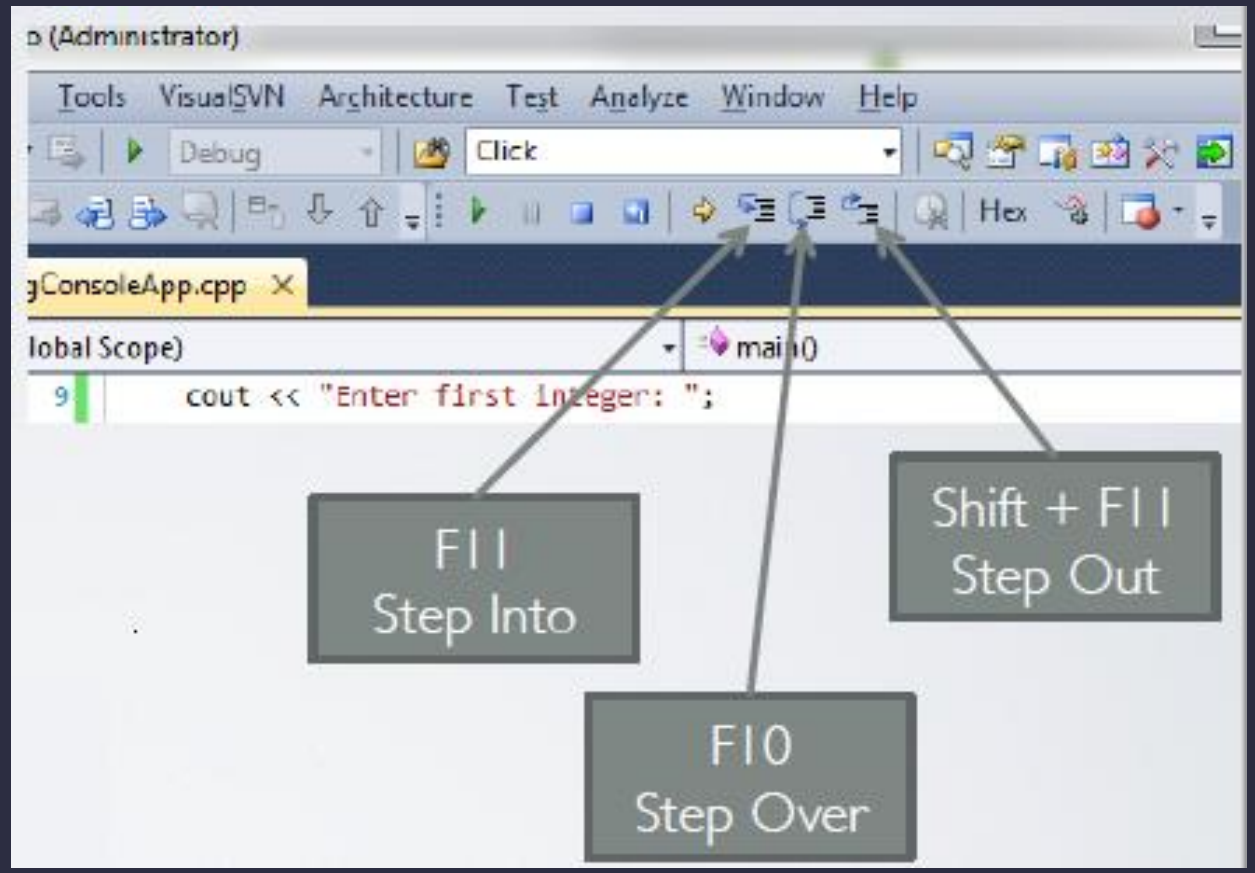


Hata Ayıklama Teknikleri



Hata Ayıklama Teknikleri

- Yukarıdaki Step Into ve Step Over tuşları ile kodun içinde hata ayıklamaya devam edebilirsiniz.
- Step Into, debug yaparken satırda bir fonksiyon varsa, o fonksiyonun içine girerek debug yapmaya devam etmeyi sağlar
- Step Over, debug yaparken fonksiyon varsa, içerisine girmeden devam etmeyi sağlar.



Hata Ayıklama Teknikleri

- Locals Ekranı
  - Değişken ismi
  - Değişken değeri
  - Değişken türünü gösterir.