An Interpreted Turkish PL

Saim SUNEL

Turkish PL

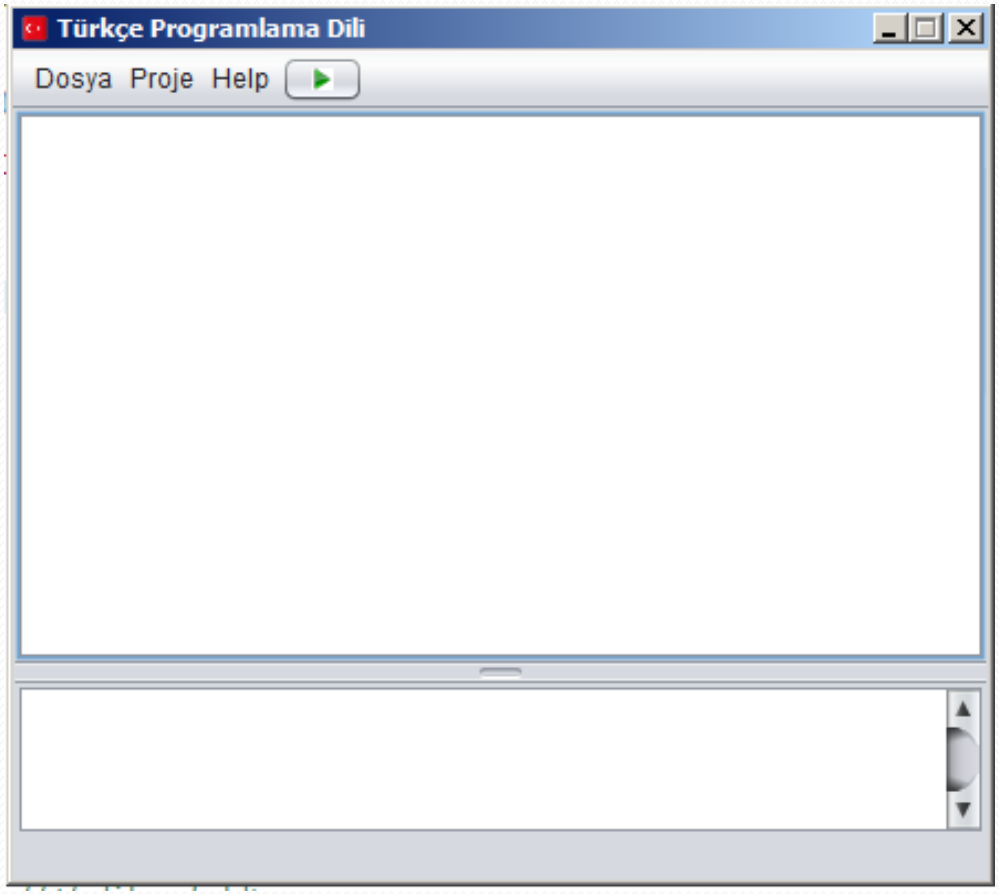
- Turkish Syntax
- Turkish people
- Education

The Language

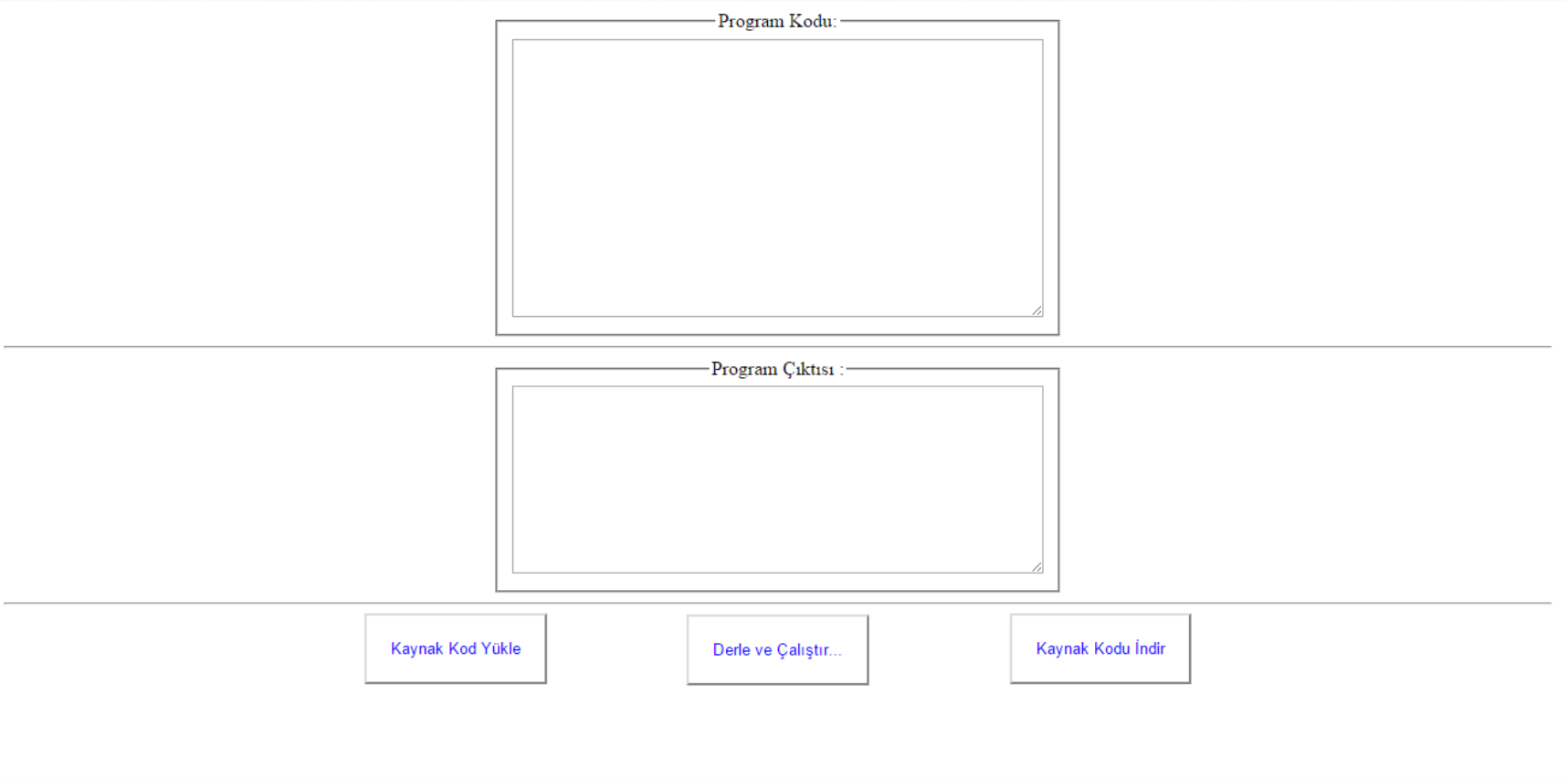
- Easily comprehensible structures …
- C like language
- Pointerless
- Heteregenous Arrays …
- Object Oriented (!)
- On Desktop
- On web

Desktop Program

- Written In Java
- Lots of platforms
- File save load
- Syntax check
- Code run
- Output
- Documentation



WebApp Program



The WebSite Program

- Run any device supports browser
- User upload , download file
- Store , Load the code on database
- Language Documentation
- Language Examples
- Communicate with the author

Data Types

- Array
- Tamsayı
- NoktalıSayı
- Karakter
- Dinamik
- Bool
- Değişken Grubu (struct)
- Referans
- Sınıf

Examples

- Karakter a = “Deneme" , b = “" , deneme = "25"
- Tamsayı x = 236
- NoktalıSayı deger = 125.36
- göster "a nın değeri : " + deneme + "\n"
- göster 2^3 + 3*4 / 2 +1 + 1 + 75+ "\nDenemelik:\n bu bir denemedir " + (karekök (39) + 3^5)
- göster "\n\n\n" + (x + deger)

Examples

- Array liste = { { 12 , 45 } , 12 , 45 , 36 ,56 }
- Array ikinci = liste
- ikinci [0] [2] = 333
- ikinci [“ilk indeks"] = { 1 , 2 , 3 , 4 }
- ikinci ["s"] ["s"] = { 3659 , 45 , 123 }
- ikinci ["den"] [“name"] = 25
- göster "Son hali : "+ ikinci ["s"] ["s"][2] +"\n"
- göster "den : " + ikinci ["den"] [“name"]
- göster “b : " + ikinci [" ilk indeks "][3] +"\n\n"
- göster “a : " + liste ["0"]["2"] + "\n\n"
- kapat

Examples

- Karakter isim = “Deneme" ;
- Referans ref
- ref -> isim
- ref = "bu bir denemedir" ;
- göster "referabs ref : " + ref + "\n\n" ;
- göster "isim : " + isim + "\n\n“

Examples

- Değişken Grubu Kare
- <
- Tamsayı x , y=25,
- Karakter isim = "denemelik"
- , deneme iç , Bool geldimi = doğru , Referans ref

- , Array liste = { 0 , 1 , "saim" , "deneme" ,{"sp , "s"}}
- >
- Karakter yazı = "Denemelik"
- Kare var
- Kare x
- var["str"] = x

- göster "den den den : " + var["str"].liste[1] + "\n\n"

- göster "son son : " + var+ "\n\n“
- kapat

Program Control Structures

- Logic expression doğru ise / doğruysa <
- Codes…
- >
- [veya logic expression doğru ise / doğruysa <

- Codes..
- >] *
- [doğru değilse <
- Codes…
- >]

Example

- Array den = { 12 , 13 }
- den [0] == 12 doğru ise <
- göster "den 0 12 ye eşit"
- >
- veya liste [10] == 36 doğru ise <
- göster " liste 10 36"
- >
- veya liste [10] == 24 doğru ise <
- göster " liste 10 24"
- >
- doğru değilse <

- göster "liste 10 25 değil \n\n"
- >
- liste ["10"] , liste [11] , liste [12] = 27 , 36 , "denemelik“
- liste.ekle(“isim soyisim”)
- liste.ekle (256)
- liste.ekle (368)
- kapat

Program Control Structures

- Logic expression doğru olduğu sürece
- <
- Codes..
- >

Example

- Tamsayı b = 0

- b < 10 doğru olduğu sürece
- <
- göster b + "\n"
- b = b + 1
- >
- göster "program bitti“
- kapat

Program Control Structures

- Burand itibaren
- <
- Codes…
- >
- Logic expressionss doğru olduğu sürece

Example

- Tamsayı sayı = 0
- buradan itibaren
- <
- göster sayı + " saim\n"
- sayı = sayı + 1
- >
- sayı > 10 doğru olduğu sürece

Program Control Structures

- döngü var = value , logic expression doğru sürece , assignment or increment
- <
- Codes…
- >

Example

- Array liste = { 1 , 2 , 3 , 4 , 5 , 6 , "deneme" }
- döngü x = 0 , x < boyut (liste) doğru olduğu sürece , x = x + 1
- <
- göster "array eleman : " + liste [x] + "\n\n"
- göster "s : " + x + "\n\n"
- >
- döngü r = 0 ,r > -23 doğru olduğu sürece , r = r+ 2
- <
- >
- kapat

Loop Control commands

- döngüden çık (break)
- baştan devam et (continue)

- Tamsayı a = 5
- a >=0 doğru olduğu sürece
- <
- a == 1 doğru ise < baştan devam et >
- a = a - 1
- >

Program Control Structures

- Seç (value)
- <

- Value [(veya value)*] ise :

- [hiçbiri değilse : <
- Codes…
- >]

- >

Example

- Karakter x = "Denemedir...3"

- seç (x)
- <
- "Deneme" ise:
- göster "x Deneme"

- "Denemedir" ise:
- göster "x Denemedir"

- "Denemedir..." veya "Denemedir...2" ise:
- göster "x Denemedir..."

- hiçbiri değilse:
- göster "sıradışı bir değer"
- >
- kapat

Functions

- Fonksiyon function_name (parameters)
- {
- Codes …
- [sonuç value]
- }

- Function call:
- Function_name (parameters)

Example

- Değişken Grubu deneme
- < Karakter isim = "Deneme“ >
- fonksiyon deneme (Tamsayı sayı , NoktalıSayı deger , Array liste , Referans x)
- {
- göster "girilen sayılar : " + sayı + " " + deger + "\n\n"
- göster "global sayı : " + sayp + "\n\n"
- göster "Ref x değer : " + x + "\n\n"
- x = 17
- göster "Ref : " + liste [1] .ismi + "\n\n"
- liste [0] = "denemelik"
- göster "girilen sayılar : " + sayı + " " + liste [0] + "\n\n"
- } deneme h
- Tamsayı sayp = 15
- sayp = 1003
- Array list = { “sp" , 12 , 3 }
- deneme (12 , 32.4 , { "dnemee" , h } , sayp) ;
- göster "original : " + list [0] + "\n\n"
- göster "Tamam" ;
- göster "fonksiyondan sonra sayı : " + sayp + "\n\n"
- kapat

Example

- /* normal matematik fonksiyonları */
- f (x , y) = x + 5 +y + y*y
- g (x , y) = x^2 + y^2

  göster "fonksiyon sonucu : " + f (16 , 10)
- göster "sonuç : " + g (4 , 3) + "\n\n"
- kapat

Example

- fonksiyon faktoriyel (Tamsayı a)
- {
- a == 1 doğru ise < sonuç 1 >
- sonuç a * faktoriyel (a -1)
- }
- göster “8 fakto : " + faktoriyel (8)
- kapat

Exception Handling

- Hata olabilir <
- Codes…
- >
- [hata variable definition <
- >]+
- hata gönder value

Example

- /* try catch içinde try catch , hata herhangi bir try da yakalanmazsa javadaki gibi hata mesajı veriyor... */
- hata olabilir
- <hata olabilir<
- hata gönder "Saim“
- >
- hata Karakter var <
- göster "Hata striing...\n\n" + var + "\n\n"
- hata gönder { 12 , 13 , 14 }
- >
- hata Tamsayı deneme
- <
- göster "Tamsayı hatası..." + deneme >
- >
- hata Array liste < göster "Gelen hata nesnesi Array ve içindekiler : \n\n"
- döngü b = 0 , b < boyut (liste) doğru olduğu sürece , b = b+1
- < göster liste [b] + "\n“
- >
- >
- göster "Program bitti..."
- kapat

Classes

- Sınıf class_name
- {
- [özel/genel variable definition] *
- [özel/genel fonksiyon function_name (parameters)
- {
- Codes..
- }
-]*
- }

Example

- Sınıf Denem {
- özel Tamsayı b , c
- genel Karakter deneme = “isminiz"
- genel NoktalıSayı iki = 25.36
- genel fonksiyon ekranayaz (Karakter isim)
- {
- göster isim + "\n\n"
- }
- }
- Denem deneme = yeni Denem ( )
- deneme.ekranayaz (“Deneme”)
- kapat

Example

- Sınıf İlk
- { genel Tamsayı kl
- özel Tamsayı jk
- genel fonksiyon deneme (Tamsayı deneme)
- { göster "İlk sınıf göster fonksiyonu" + "\n\n" + deneme +"\n\n“
- göster "\n\n Sınıfın kl : " + kl + "\n\n"
- kl = 251
- jk =157
- sonuç 369
- } genel fonksiyon al ( )
- { /* deneme implemente edilecek*/
- sonuç jk
- } } İlk a = yeni İlk ()
- a.kl = 1258
- göster "a nkta kl : " + a.kl
- Tamsayı k = a.deneme (15)
- göster "\n\n" + a.kl + "\n\n“
- göster “sonuç : " + a.al ()
- göster a.jk
- kapat

Type Casting

- (Tamsayı) , (Karakter) , (Bool) , (NoktalıSayı)

- Tamsayı o = (Tamsayı) 25.36
- NoktalıSayı l = (NoktalıSayı) “125.36”

Example

- Tamsayı a , b
- a = (Tamsayı) oku ("Birinci sayıyı giriniz : ")
- b = (Tamsayı) oku ("İkinci sayıyı giriniz : ")
- göster "Toplam : " + (a + b) + "\n\n\n“
- Karakter isim = oku ("isminizi girin : ")
- göster "isim : " + isim + "\n\n“
- kapat