

mushroom
İÇİN RESMİ DOKÜMANTASYON

Sporları yayın!



VERİ TIPLERİ

INTEGER:

INTEGER veri tipi sadece tam sayıları depolar.

ÖRN: 1, 5, -3

FLOAT:

FLOAT veri tipi ondalıklı sayıları depolar.

ÖRN: 1.2, 5.6, -7.2

STRING:

STRING veri tipi karakter dizileri depolar. STRING'ler tırnak işaretleri arasında kullanılır.

ÖRN: "Hello World"

LIST:

LIST (LİSTE) veri tipi birden fazla değer depolamak için kullanılır. Bu değerler farklı veri tiplerinden olabilirler. LIST elemanlarına indisler kullanılarak ulaşılabilir. İlk elemanın indisi 0'dır ve diğer elemanlar ardışık olarak indis alır. LIST elemanlarına [listAdı/elemanİndisi](#) kullanılarak ulaşılabilir.

LIST elemanları aşağıdaki gibi tanımlanabilir:

```
VAR listAdı= [listElemanı,listElemanı]
```

NOT: Veri belirlenirken ve üzerinde değişiklik yapılırken VAR anahtar kelimesi kullanılır. Bu tüm veri tipleri için geçerlidir.

OPERATÖRLER

Sayılar üzerinde kullanılabilen 5 operatör bulunur (INTEGER ve FLOAT'lar). Bunlar TOPLAMA operatörü (+), ÇIKARMA operatörü (-), ÇARPMA operatörü (*),BÖLME operatörü (/) ve ÜS ALMA operatörüdür (^).

ANAHTAR KELİMELEER

VAR:

Değişkenleri tanımlamak ve üzerlerinde işlem yapmak için kullanılır. Değişkenin tipi otomatik olarak belirlenir.

SÖZ DİZİMİ:

VAR (değişken adı) = (değişken değeri)

VAR (değişken adı) (operatör) (değer)

AND/OR:

Verilen ifadelere göre TRUE (DOĞRU) ya da FALSE (YANLIŞ) döndürür. (AND=VE OR=VEYA)

AND= Her iki ifade de doğruysa TRUE döndürür

OR= İfadelerden birisi doğruysa TRUE döndürür

SÖZ DİZİMİ:

(ifade) Mantıksal Operatör (ifade)

NOT:

NOT (DEĞİL)= İfadenin değerini ters çevirir.

SÖZ DİZİMİ:

NOT (ifade)

IF İFADELERİ:

IF (EĞER) İFADELERİ verilen şart doğruysa bir işlem yapmak için kullanılır. ELIF (O DEĞİL BUYSA) ilk şart doğru değilse kontrol edilecek ek şartlar oluşturmak için kullanılır. ELSE (YOKSA) verilen şartların hiçbiri doğru değilse yapılacak işlemler belirlemek için kullanılır. THEN (SONRA) şart ifadesinden sonra kullanılır. END (BİTİŞ) IF İFADESİ'nin sonunu belirtmek için kullanılır. Bir IF İFADESİ'nin çalışması için ELIF ve ELSE gerekli değildir.

SÖZ DİZİMİ:

IF (şart) THEN (işlem)

ELIF (şart) THEN (işlem)

ELSE (işlem)

END

FOR DÖNGÜLERİ:

FOR DÖNGÜLERİ bir işlemi tekrar ve tekrar uygulamak için kullanılır. Bir FOR DÖNGÜSÜ FOR, TO, THEN, STEP ve END anahtar kelimeleri kullanılarak oluşturulabilir. TO döngünün kaç kere çalışacağını bir indis aralığı belirleyerek belirtmek için kullanılır. Her işlem FOR DÖNGÜSÜNDE bir işlem alır. Genellikle indis "i" ile gösterilir ancak herhangi bir değişken ismi kullanılabilir. THEN indis aralığından sonra kullanılır. STEP indislerin kaçar kaçar artacağını belirlemek için kullanılır. STEP kullanılmazsa bu değer 1 olur. END döngünün sonunda kullanılır.

SÖZ DİZİMİ:

FOR i=(ilk değer) TO (son değer) THEN

(işlem)

END

FOR i=(ilk değer) TO (son değer) STEP (artış miktarı) THEN

(işlem)

END

NOT: "i" (ilk değer)'den (son değer)'e kadar döngünün her iterasyonu için bir değer alır. (son değer) dahil edilmez.

WHILE DÖNGÜLERİ:

WHILE DÖNGÜLERİ, FOR DÖNGÜLERİNE benzer bir şekilde kullanılan başka bir tür döngülerdir. Ancak WHILE DÖNGÜLERİ aralıklar yerine şartlar alırlar. Bir WHILE DÖNGÜSÜ'nde WHILE, THEN ve END anahtar kelimelerini kullanılır. Bir WHILE DÖNGÜSÜ şartın karşılanıp karşılanmadığını en başta ve döngünün her iterasyonunda kontrol eder. WHILE anahtar kelimesi döngünün başında kullanılır. THEN şarttan sonra kullanılır. END döngünün sonunda kullanılır.

SÖZ DİZİMİ:

WHILE (şart) THEN

(işlem)

END

CONTINUE/BREAK:

CONTINUE(DEVAM ET) ve BREAK (KIR) FOR ve WHILE döngülerinde kullanılabilirler. CONTINUE mevcut iterasyonu durdurup sonraki iterasyondan devam etmek için kullanılır. BREAK ise mevcut iterasyonu durdurup döngüden çıkar.

FONKSİYONLAR:

FONKSİYONLAR kod paketleri oluşturmak için kullanılırlar. Bu FONKSİYONLAR daha sonra içindeki kodun uygulanması için çağrılabilir. FONKSİYONLAR içlerindeki kodun verdiği sonucu dönerler. FONKSİYONLAR çağrılırken parametre veya parametreler alabilirler. Bu parametreler FONKSİYONUN altında değişkenlerde saklanabilirler. Bu değişkenler FONKSİYON'un dışında bir etki yapamazlar. FUN, RETURN(DÖNDÜR) ve END anahtar kelimeleri FONKSİYON oluşturmada kullanılır. FUN FONKSİYONUN başında kullanılır. RETURN bir değeri döndürmek için kullanılır. END is FONKSİYON'un sonunda kullanılır. Ek olarak "-->" tek satırlık FONKSİYONLAR oluşturmak için kullanılabilir. "-->" RETURN'e bir alternatif olarak kullanılır. "-->" kullanıldığında RETURN ve END anahtar kelimelerine gerek kalmaz.

SÖZ DİZİMİ:

FUN (fonksiyon adı) ((parametreyi saklayacak değişken adı))

(kod parçası)

RETURN (döndürülecek değer)

END

NOT: Fonksiyonda birden fazla parametre kullanılabilir. Bu parametreler virgüller kullanılarak alınabilir:

FUN (fonksiyon adı) ((parametreyi saklayacak değişken adı),(parametreyi saklayacak ikinci değişken adı))

TEK SATIRLIK FONKSİYON SÖZ DİZİMİ:

FUN (fonksiyon adı) ((parametreyi saklayacak değişken adı) -> (döndürülecek değer)

NOT: Fonksiyonlar aşağıdaki gibi çağrılır:

(fonksiyon adı)(parametreler)

NULL/FALSE/TRUE:

NULL(YOK) bir değer sahibi olmamayı ya da 0'ı temsil eder. FALSE(YANLIŞ) ve TRUE(DOĞRU) bir şartın durumunu temsil etmek için kullanılırlar. FALSE 0'ı ve TRUE 1'i temsil etmek için kullanılır.

MATH_PI:

MATH_PI Pi sayısının ilk 15 basamağını döner.

NOT: MATH_PI kabuktan bir fonksiyon gibi çağrılabilir ancak parametrelere ihtiyaç duymaz.

FONKSİYONLAR

PRINT/PRINT_RET:

PRINT(YAZDIR) ekrana veri yazdırmak için kullanılır. Bu veri integer, float, string ya da list olabilir. PRINT_RET yazdırılacak değeri döndürür ancak yazdırmaz. Yazdırılacak değerler parametre olarak verilir.

INPUT/INPUT_INT:

INPUT, string tipinde bir veriyi kullanıcıdan almak için kullanılır. INPUT_INT ise integer tipinde bir veri alır. Girdi integer tipinde değilse hata verir ve yeni bir girdi ister. Her iki fonksiyon da parametre almaz.

CLEAR/CLS:

Hem CLEAR hem de CLS ekranı temizlemek için kullanılır. İkisi de parametre almaz.

IS_NUM/IS_STR/IS_LIST/IS_FUN:

IS_NUM parametre bir sayıysa (INTEGER ya da FLOAT) 1, değilse 0 döner. IS_STR parametre bir STRING ise 1, değilse 0 döner.

IS_LIST parametre bir LIST ise 1, değilse 0 döner. IS_FUN parametre bir FONKSİYON ise 1, değilse 0 döner.

APPEND:

APPEND (EKLE) bir listenin sonunda veri eklemek için kullanılır. Tek seferde sadece bir değer eklenebilir. Ekleme yapılacak listeyi ilk ve eklenecek veriyi ikinci parametre olarak alır.

POP:

POP bir liste elemanını indisine göre çıkarmak için kullanılır. Çıkarma yapılacak listeyi ilk ve çıkarılacak elemanın indisini ikinci parametre olarak alır.

EXTEND:

EXTEND iki listeyi birleştirmek için kullanılır. İkinci liste ilk listenin sonuna eklenir. Parametre olarak iki liste alır.

KABUĞA ÖZEL KOMUTLAR

SETUP:

SETUP keywords.txt ve functions.txt dosyalarının dosya yollarını alarak varsayılan anahtar kelimeleri ve yerleşik fonksiyonları değiştirmek için kullanılır.

ADD TO PATH:

ADD TO PATH mushroom'u sistemin Path değişkenine eklemek için kullanılır. Bunu yaparak mushroom'a herhangi bir altdizinden bir terminale "mushroom" yazarak ulaşabilirsiniz.

TRANSLATE:

TRANSLATE ikişer adet keywords.txt ve functions.txt dosya yollarını alarak kendi özelleştirilmiş dilinizde yazdığınız kodu başka bir özelleştirilmiş dile çevirmenizi sağlar. Bu komut varsayılan dile çeviri yapmak için de kullanılabilir.

VSCODE:

VSCODE Visual Studio Code eklentinizi kendi özelleştirilmiş dilinize uygun hale getirmek için kullanılır. keywords.txt, functions.txt ve VSCode dosyaları arasında bulunan mush.tmlanguage.json dosyalarının yollarını alır (komut sizi mush.tmlanguage.json dosyasına yönlendirecek).

RUN:

RUN çalıştırmak istediğiniz kodun dosya yolunu alır ve onu çalıştırır.

EK NOTLAR

- mushroom noktalı virgül (;) olmadan yeni satırları algılayabilir, ancak noktalı virgülleri birden fazla satır kodu tek satıra sığdırmak için kullanabilirsiniz.
- mushroom .mush dosya tipini kullanır ancak .txt dosyalarını da çalıştırabilir.

Örnekler

mushroom ile kod yazmaya başlamadan önce indirilen mushroom.zip dosyası ayıklanmalıdır.

Bunun yanında opsiyonel olarak Visual Studio Code ve mushroom eklentisi kullanılabilir.

Merhaba Dünya

PRINT fonksiyonu ekrana yazı yazdırmak için kullanılır.

```
helloWorld.mush
1 PRINT("hello world!")
```

Kodu çalıştırmak için kabukta RUN komutu çalıştırılır ve çalıştırılacak .mush dosyasının dosya yolu girdi olarak verilir.

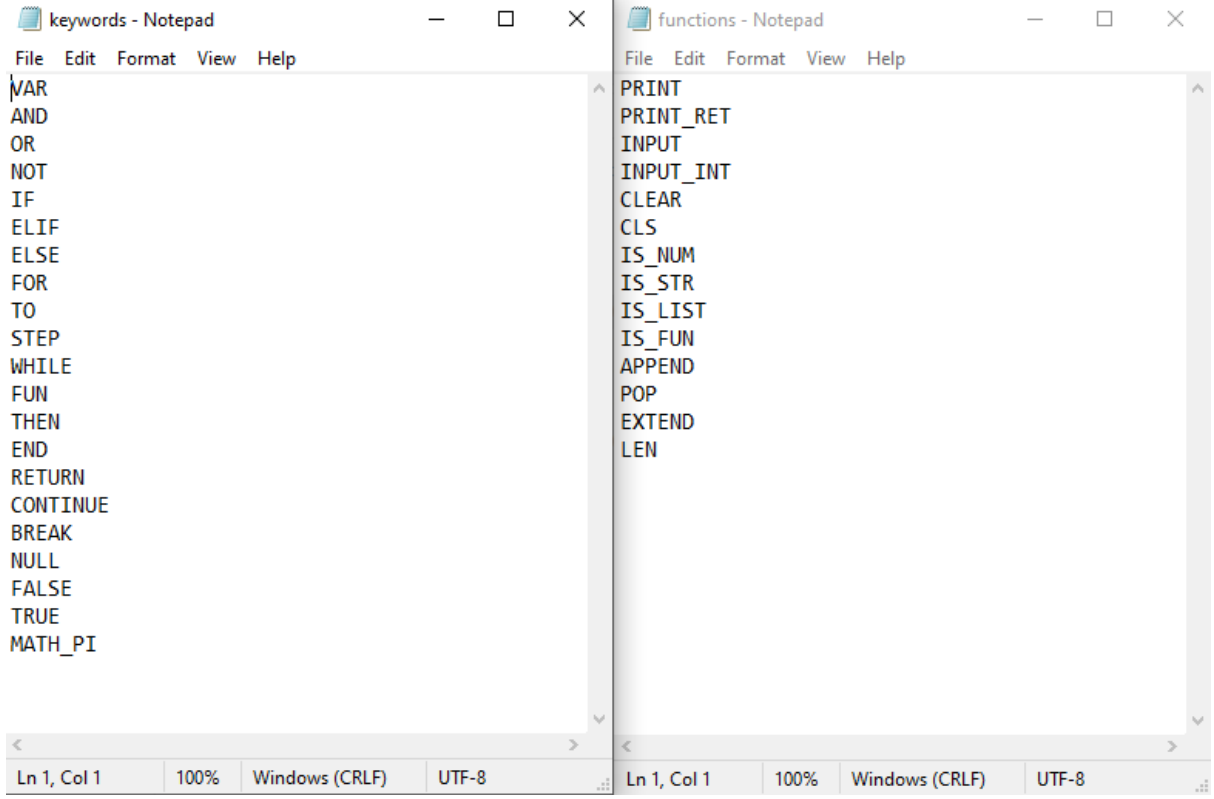
```
mushroom > RUN
Enter File Path:C:\Users\kberk\Desktop\mushroomÖrnekleri\helloWorld.mush
hello world!
Ø
```

Özelleştirme

mushroom tüm anahtar kelimelerini keywords.txt ve functions.txt adlı iki dosyadan alır.

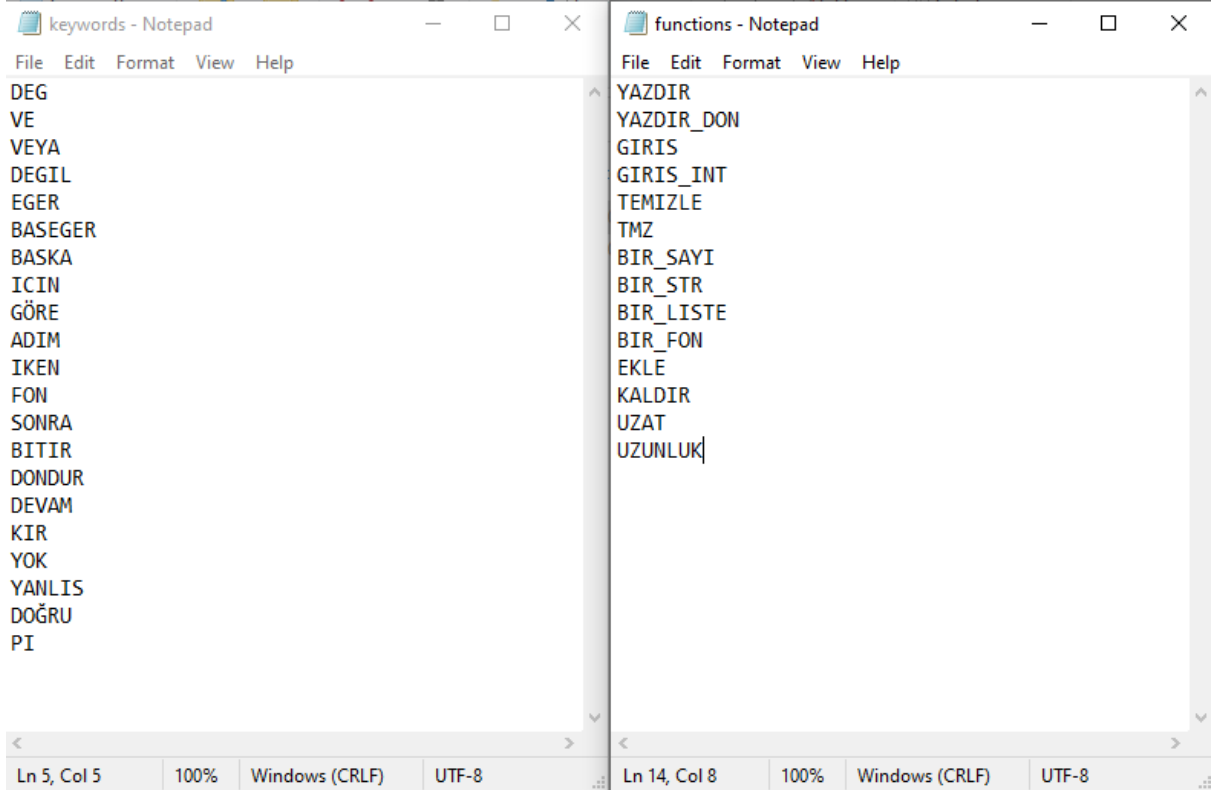
Bu dosyadaki kelimeler değiştirilerek dil istenilen hale getirilebilir.

Aşağıdaki iki dosyada varsayılan anahtar kelimeler bulunmaktadır.



The image shows two Notepad windows side-by-side. The left window, titled 'keywords - Notepad', contains a list of default keywords: VAR, AND, OR, NOT, IF, ELIF, ELSE, FOR, TO, STEP, WHILE, FUN, THEN, END, RETURN, CONTINUE, BREAK, NULL, FALSE, TRUE, and MATH_PI. The right window, titled 'functions - Notepad', contains a list of default functions: PRINT, PRINT_RET, INPUT, INPUT_INT, CLEAR, CLS, IS_NUM, IS_STR, IS_LIST, IS_FUN, APPEND, POP, EXTEND, and LEN. Both windows have a status bar at the bottom showing 'Ln 1, Col 1', '100%', 'Windows (CRLF)', and 'UTF-8'.

Örneğin bu iki dosya aşağıdaki iki dosya ile değiştirilerek dil Türkçe'ye çevrilebilir.



The image shows two Notepad windows side-by-side, displaying customized keywords and functions in Turkish. The left window, titled 'keywords - Notepad', contains a list of Turkish keywords: DEG, VE, VEYA, DEGIL, EGER, BASEGER, BASKA, ICIN, GÖRE, ADIM, IKEN, FON, SONRA, BITIR, DONDUR, DEVAM, KIR, YOK, YANLIS, DOĞRU, and PI. The right window, titled 'functions - Notepad', contains a list of Turkish functions: YAZDIR, YAZDIR_DON, GIRIS, GIRIS_INT, TEMIZLE, TMZ, BIR_SAYI, BIR_STR, BIR_LISTE, BIR_FON, EKLE, KALDIR, UZAT, and UZUNLUK. Both windows have a status bar at the bottom showing 'Ln 5, Col 5' and 'Ln 14, Col 8' respectively, along with '100%', 'Windows (CRLF)', and 'UTF-8'.

Bu deęişiklięi yapmak için kabukta SETUP komutu alıřtırılır ve yeni kullanılacak keywords.txt ve functions.txt dosyalarının dosya yolları girdi olarak verilir.

Deęişiklik yapıldıktan sonra mushroom'un yeniden başlatılması gerekir.

```
mushroom > SETUP
Enter keywords.txt file path:
"C:\Users\kberk\Desktop\mushroomÖrnekleri\keywords.txt"
Enter functions.txt file path:
"C:\Users\kberk\Desktop\mushroomÖrnekleri\functions.txt"
Please restart mushroom
mushroom >
```

Visual Studio kullanılıyorsa kod renklendirmesinin yeni anahtar kelimelerle alıřması için mushroom eklentisinin de özelleřtirilmesi gerekir.

Bu iřlem kabukta VSCODE komutunun alıřtırılmasıyla yapılır.

```
mushroom > VSCODE
Enter keywords.txt File Path:
"C:\Users\kberk\Desktop\TürkeŞablon\keywords.txt"
Enter functions.txt File Path:
"C:\Users\kberk\Desktop\TürkeŞablon\functions.txt"
Enter mush.tmLanguage.json File Path:
(located in Users>User>.vscode>extensions>k.berkboz.mushroom-0.0.1>.syntaxes>)
"C:\Users\kberk\.vscode\extensions\k-berkboz.mushroom-0.0.3\syntaxes\mush.tmLanguage.json"
mushroom >
```

Özelleřtirilmiř Merhaba Dünya

```
≡ merhabaDünya.mush
1  YAZDIR(["merhaba dünya"])
```

```
mushroom > RUN
Enter File Path:"C:\Users\kberk\Desktop\mushroomÖrnekleri\merhabaDünya.mush"
merhaba dünya
0
```

Deęişken Tanımlama

Deęişken tanımlamak için VAR anahtar kelimesi kullanılır.

```
≡ exampleCode.mush
1  VAR x=5
2  PRINT(x)
3  VAR y= "test"
4  PRINT(y)
5  VAR z= [1,2,3]
6  PRINT(z)
```

```
mushroom > RUN
Enter File Path:C:\Users\kberk\Desktop\mushroomÖrnekleri\exampleCode.mush
5
test
1, 2, 3
0
```

IF Yapısı

IF yapısı bir programda karar mekanizması olarak kullanılır.

Kodun farklı durumlarda farklı komutlar çalıştırmasını sağlar.

```
exampleCode.mush
1  VAR x=5
2  IF (x>5) THEN
3  PRINT("x>5")
4  ELIF (x<5) THEN
5  PRINT("x<5")
6  ELSE
7  PRINT("x=5")
8  END
```

```
mushroom > RUN
Enter File Path:C:\Users\kberk\Desktop\mushroomÖrnekleri\exampleCode.mush
x=5
0
```

Döngüler

Döngüler bir komutu birden fazla kez gerçekleştirmek için kullanılır.

mushroom'da FOR ve WHILE olmak üzere iki çeşit döngü bulunur.

```
exampleCode.mush
1  FOR i= 0 TO 10 STEP 2 THEN
2  PRINT(i)
3  END
4  PRINT("=====")
5  VAR x= 0
6  WHILE (x<10) THEN
7  PRINT(x)
8  VAR x=x+2
9  END
10 PRINT["====="]
```

```
mushroom > RUN
Enter File Path:C:\Users\kberk\Desktop\mushroomÖrnekleri\exampleCode.mush
0
2
4
6
8
=====
0
2
4
6
8
=====
0
```

Fonksiyonlar

Fonksiyonlar bir kod parçasının paketlenip tekrar tekrar kullanılmasını sağlar.

```
☰ exampleCode.mush
1  FUN kare(x)
2  RETURN x*x
3  END
4
5  PRINT(kare(5))
```

```
mushroom > RUN
Enter File Path:C:\Users\kberk\Desktop\mushroomÖrnekleri\exampleCode.mush
25
0
```