Home Whiteboard AI Assistant Online Compilers Jobs Tools Articles Corporate T

SQL HTML CSS Javascript Python Java C C++ PHP Scala C#

Python - Match-Case Statement

Python match-case Statement

A Python **match-case** statement

tutorialspoint Categories Sea Library Courses Certifications Login

one or more case blocks. Only the first pattern that matches gets executed. It is also possible to extract components (sequence elements or object attributes) from the value into variables.

With the release of Python 3.10, a pattern matching technique called **match-case** has been introduced, which is similar to the **switch-case** construct available in C/C++/Java etc. Its basic use is to compare a variable against one or more values. It is more similar to pattern matching in languages like Rust or Haskell than a switch statement in C or C++.

Syntax

The following is the syntax of match-case statement in Python -

Python - break Statement
Python - continue Statement
Python - pass Statement
Python - Nested Loops
Python Functions & Modules
Python - Functions
Python - Default Arguments

```
match variable_name:
   case 'pattern 1' : statement 1
   case 'pattern 2' : statement 2
   ...
   case 'pattern n' : statement n
```

Example

The following code has a function named weekday(). It receives an integer argument, matches it with all possible weekday number values, and returns the corresponding name of day.

</> Open Compiler

```
def weekday(n):
   match n:
      case 0: return "Monday"
      case 1: return "Tuesday"
      case 2: return "Wednesday"
      case 3: return "Thursday"
      case 4: return "Friday"
      case 5: return "Saturday"
      case 6: return "Sunday"
      case _: return "Invalid day number"
print (weekday(3))
print (weekday(6))
print (weekday(7))
```

On executing, this code will produce the following output −

```
Thursday
Sunday
Invalid day number
```

The last case statement in the function has "_" as the value to compare. It serves as the wildcard case, and will be executed if all other cases are not true.

Combined Cases in Match Statement

Sometimes, there may be a situation where for more than one cases, a similar action has to be taken. For this, you can combine cases with the OR operator represented by "|" symbol.

Example

The code below shows how to combine cases in match statement. It defines a function named access() and has one string argument, representing the name of the user. For admin or manager user, the system grants full access; for Guest, the access is limited; and for the rest, there's no access.

Open Compiler

```
def access(user):
   match user:
      case "admin" | "manager": return "Full access"
      case "Guest": return "Limited access"
      case _: return "No access"
print (access("manager"))
print (access("Guest"))
print (access("Ravi"))
```

On running the above code, it will show the following result −

```
Full access
Limited access
No access
```

List as the Argument in Match Case Statement

Since Python can match the expression against any literal, you can use a list as a case value. Moreover, for variable number of items in the list, they can be parsed to a sequence with "*" operator.

Example

In this code, we use list as argument in match case statement.

Open Compiler

```
def greeting(details):
   match details:
      case [time, name]:
         return f'Good {time}
{name}!'
      case [time, *names]:
         msg=''
         for name in names:
            msg+=f'Good
{time} {name}!\n'
         return msg

print (greeting(["Morning",
"Ravi"]))
print
(greeting(["Afternoon","Guest
"]))
print (greeting(["Evening",
"Kajal", "Praveen", "Lata"]))
```

On executing, this code will produce the following output −

```
Good Morning Ravi!
Good Afternoon Guest!
Good Evening Kajal!
Good Evening Praveen!
Good Evening Lata!
```

Using "if" in "Case" Clause

Normally Python matches an expression against literal cases. However, it allows you to include if statement in the case clause for conditional computation of match variable.

Example

In the following example, the function argument is a list of amount and duration, and the intereset is to be calculated for amount less than or more than 10000. The condition is included in the **case** clause.

</> Open Compiler

```
def intr(details):
   match details:
      case [amt, duration] if
amt<10000:
         return
amt*10*duration/100
      case [amt, duration] if
amt>=10000:
         return
amt*15*duration/100
print ("Interest = ",
intr([5000,5]))
print ("Interest = ",
intr([15000,3]))
```

On executing, this code will produce the following output −

```
Interest = 2500.0
Interest = 6750.0
```

TOP TUTORIALS

Python Tutorial
Java Tutorial
C++ Tutorial
C Programming Tutorial
C# Tutorial
PHP Tutorial
R Tutorial
HTML Tutorial
CSS Tutorial
JavaScript Tutorial
SQL Tutorial

TRENDING TECHNOLOGIES

Cloud Computing Tutorial
Amazon Web Services Tutorial
Microsoft Azure Tutorial
Git Tutorial
Ethical Hacking Tutorial
Docker Tutorial
Kubernetes Tutorial
DSA Tutorial
Spring Boot Tutorial
SDLC Tutorial

CERTIFICATIONS

Business Analytics Certification
Java & Spring Boot Advanced Certification
Data Science Advanced Certification
Cloud Computing And DevOps
Advanced Certification In Business Analytics
Artificial Intelligence And Machine Learning
DevOps Certification
Game Development

COMPILERS & EDITORS

Online Java Compiler
Online Python Compiler
Online Go Compiler
Online C Compiler
Online C++ Compiler
Online C# Compiler
Online PHP Compiler
Online MATLAB Compiler
Online Bash Compiler
Online SQL Compiler
Online Html Editor

Unix Tutorial

Certification

Front-End Developer Certification

AWS Certification Training

Python Programming Certification

ABOUT US | OUR TEAM | CAREERS | JOBS | CONTACT US | TERMS OF USE | PRIVACY POLICY |

REFUND POLICY | COOKIES POLICY | FAQ'S

tutorialspoint

GET IT ON Google Play

Download on the App Store

Tutorials Point is a leading Ed Tech company striving to provide the best learning material on technical and non-technical subjects.