Match

September 17, 2024

1 Match/Case

The *match/case* construct was introduced in Python 3.10. Until then, Python didn't have an equivalent of switch/case in languages like C, C++ and Java. You could do multiple dispatch using dictionary lookup of functions to call, but that was not as readable as you would like it to be.

As usual, when Python introduced the functionality, it went over and beyond the implementation in other languages.

1.1 The simple form

Match/case will execute the first matching block only.

Note that you can use the reserved word *other* or *_* for the default case.

[19]:
```
designer = 'Bjarne'

match designer:
    case 'Larry':
        print(f'{designer} is the designer of Perl')
    case 'Dennis':
        print(f'{designer} is the designer of C')
    case 'Guido':
        print(f'{designer} is the designer of Python')
    case 'Brendan':
        print(f'{designer} is the designer of Javascript')
    case _:
        print(f'{designer} is not the designer of a notable programming␣
 ↪language')
```

```
Bjarne is not the designer of a notable programming language
```

1.2 Sequence matching

You can do matching on sequences. Literals have to be exactly matched, while variables get bound to whatever is in the matching position.

[15]:
```
sentence = 'Gurkorna cyklar i motvind'.split()

match sentence:
```

```
    case [a, b, 'i', 'motvind', 'nu']:
        print (a)
    case [a, b, 'i', 'motvind']:
        print (a, b)
    case [a, b, 'i', 'medvind']:
        print (a, b)
    case ['Gurkorna', 'cyklar', c, d]:
        print(c, d)
    case other:
        pass
```

```
Gurkorna cyklar
```

1.3 Alternatives and *args

You can use the | operator to indicate alternatives.

You can gather multiple positions in a single variable using a * in front of the variable, akin to how it is used in sequence unpacking.

[10]:
```
def file_handler(command):
    match command.split():
        case ['show']:
            print('List all files and directories: ')
            # code to list files
        case ['remove' | 'delete', *files] if '--ask' in files:
            del_files = [f for f in files if len(f.split('.'))>1]
            print('Please confirm: Removing files: {}'.format(del_files))
            # code to accept user input, then remove files
        case ['remove' | 'delete', *files]:
            print('Removing files: {}'.format(files))
            # code to remove files        case ['remove', *files]:

file_handler('show')
```

```
List all files and directories:
```

[11]:
```
file_handler('delete --ask valuable.c precious.py')
```

```
Please confirm: Removing files: ['valuable.c', 'precious.py']
```

[12]:
```
file_handler('delete whatever.pl')
```

```
Removing files: ['whatever.pl']
```