07-while-loop

September 16, 2022

1 While loop

Some times we just need to loop until some condition is satisfied, or even loop indefinitely until the application is stopped.

Cases where we don't really have something to iterate on, and therefore the `for` loop would be a poor choice.

[1]:
```
n = 39
remainders = []

while n > 0:
    remainder = n % 2            # remainder of division by 2
    n //= 2                      # we divide n by 2
    remainders.append(remainder) # we keep track of remainders

# reassign the list to its reversed copy and print it
remainders = remainders[::-1]
print(remainders)                # guess what is the remainders list to 39...
```

```
[1, 0, 0, 1, 1, 1]
```

1.1 continue

The continue statement, tells the looping construct (for or while) to immediately stop execution of the body and go to the next iteration, if any.

[2]:
```
from datetime import date, timedelta
today = date.today()
tomorrow = today + timedelta(days=1) # today + 1 day is tomorrow
products = [
    {'sku': '1', 'expiration_date': today, 'price': 100.0},
    {'sku': '2', 'expiration_date': tomorrow, 'price': 50},
    {'sku': '3', 'expiration_date': today, 'price': 20},
]

while products:
    product = products.pop()
    if product['expiration_date'] != today:
```

```
        continue                # go to the next product
    product['price'] *= 0.8     # equivalent to applying 20% discount
    print('Price for sku', product['sku'], 'is now', product['price'])
```

```
Price for sku 3 is now 16.0
Price for sku 1 is now 80.0
```

1.2 Break

The **break** statement terminates the current loop and resumes execution at the next statement

[3]:
```
items = [0, None, 0.0, True, 0, 7]  # True and 7 evaluate to True
found = False                       # this is called "flag"

while items:
    item = items.pop(0)
    print('scanning item', item)
    if item:                        # item evaluates True?
        found = True                # we update the flag
        break

if found:                           # we inspect the flag
    print('At least one item evaluates to True')
else:
    print('All items evaluate to False')
```

```
scanning item 0
scanning item None
scanning item 0.0
scanning item True
At least one item evaluates to True
```

1.3 else

If the loop ends normally, because of exhaustion of the iterator (**for** loop) or because the condition is finally not met (**while** loop), then the **else** suite (if present) is executed.

In case execution is interrupted by a **break** statement, the **else** clause is not executed.

[4]:
```
people = [('James', 17), ('Kirk', 9), ('Lars', 13), ('Robert', 8)]
driver = None

# old way!
while people:
    person, age = people.pop(0)
    if age >= 18:
        driver = (person, age)
        break
```

```
if driver is None:
    print('Driver not found.')
```

```
Driver not found.
```

[5]:
```
people = [('James', 17), ('Kirk', 9), ('Lars', 13), ('Robert', 8)]
driver = None

# the same loop in a pythonic way
while people:
    person, age = people.pop(0)
    if age >= 18:
        driver = (person, age)
        break
else:                           # <---else clause!
    print('Driver not found.')
```

```
Driver not found.
```

2 Exercises

Go here...

[ ]:
```

```