Programming Lab

Lezione 5

Le eccezioni ed il flusso try-except

Stefano Alberto Russo

Le eccezioni

In Python gli errori si chiamano “eccezioni”:

```
1
2   my_var = 'Ciao'
3   print(mia_variablie)
4
```

```
> python lezione5.py
Traceback (most recent call last):
  File "lezione5.py", line 3, in <module>
    print(mia_variablie)
NameError: name 'mia_variablie' is not defined
```

```
1
2   my_var = 'Ciao'
3   float(my_var)
4
```

```
> python lezione5.py
Traceback (most recent call last):
  File "lezione5.py", line 3, in <module>
    float(my_var)
ValueError: could not convert string to float: 'Ciao'
```

Cosa sono le eccezioni

Le eccezioni sono oggetti. Come tutto in Python.

→ *Estendono la classe base “Exception”*

Esempi:

```
Exception
    ArithmeticError
        FloatingPointError
        ZeroDivisionError
    AttributeError
    SyntaxError
    NameError
    TypeError
    ValueError
```

Il costrutto try-except

Permette di gestire le eccezioni (errori).

```
1
2   my_var = 'Ciao'
3
4   try:
5       my_var = float(my_var)
6   except:
7       print('Non posso convertire "my_var" a valore numerico!')
8
```

```
› python lezione5.py
Non posso convertire "my_var" a valore numerico!
```

Il costrutto try-except

Permette di gestire le eccezioni (errori).

```
1
2   my_var = 'Ciao'
3
4   try:
5       my_var = float(my_var)
6   except:
7       print('Non posso convertire "my_var" a valore numerico!')
8       print('Uso il valore di default "0.0" per "my_var"')
9       my_var = 0.0
10
```

```
> python lezione5.py
Non posso convertire "my_var" a valore numerico!
Uso il valore di default "0.0" per "my_var"
```

Il costrutto try-except

...e posso avere l’eccezione dentro l’ except:

```
1
2   my_var = 'Ciao'
3
4   try:
5       my_var = float(my_var)
6   except Exception as e:
7       print('Non posso convertire "my_var" a valore numerico!')
8       print('La variabile "my_var" valeva: "{}"'.format(my_var))
9       print('Ed ho avuto questo errore: "{}"'.format(e))
10
```

```
> python lezione5.py
Non posso convertire "my_var" a valore numerico!
La variabile "my_var" valeva: "Ciao"
Ed ho avuto questo errore: "could not convert string to float: 'Ciao'"
```

Il costrutto try-except

...e posso avere l'eccezione dentro l' except:

```
1
2   my_var = 'Ciao'
3
4   try:
5       my_var = float(my_var)
6   except Exception as e:
7       print('Non posso convertire "my_var" a valore numerico!')
8       print('La variabile "my_var" valeva: "{}"'.format(my_var))
9       print('Ed ho avuto questo errore: "{}"'.format(e))
10
```

Per usare l'eccezione dentro l'except, devo sempre specificare che eccezione voglio gestire, se le voglio gestire tutte allora uso la classe base Exception

```
> python lezione5.py
Non posso convertire "my_var" a valore numerico!
La variabile "my_var" valeva: "Ciao"
Ed ho avuto questo errore: "could not convert string to float: 'Ciao'"
```

Il costrutto try-except

Posso infatti anche gestire solo eccezioni specifiche:

```
try:

    my_var = float(my_var)

except ValueError:
    print('Non posso convertire "my_var" a valore numerico!')
    print('Ho avuto un errore di VALORE. "my_var" valeva "{}".'.format(my_var))

except TypeError:
    print('Non posso convertire "my_var" a valore numerico!')
    print('Ho avuto un errore di TIPO. "my_var" era di tipo "{}".'.format(type(my_var)))

except Exception as e:
    print('Non posso convertire "my_var" a valore numerico!')
    print('Ho avuto un errore generco: "{}"'.format(e))
```

Esercizio (parte 1)

Modificate l'oggetto **CSVFile** della lezione precedente in modo che stampi a schermo un messaggio di errore se si cerca di aprire un file non esistente.

Potete fare questo controllo:

a) nella funzione `get_data()`, oppure

b) nell' `__init__()` (basta che leggiate la prima riga per vedere se il file esiste)

Esercizio (parte 2)

Estendete l'oggetto **CSVFile** chiamandolo **NumericalCSVFile** e facendo in modo che converta automaticamente a numero tutte le *colonne* tranne la prima (della data).

Poi, aggiungete questi due campi al file "shampoo_sales.csv":

```
01-01-2015,
01-02-2015,ciao
```

e gestite gli errori che verranno generati in modo che le linee vengano saltate senza bloccare il programma ma che venga stampato a schermo l'errore