Programmazione avanzata a.a. 2021-22 A. De Bonis

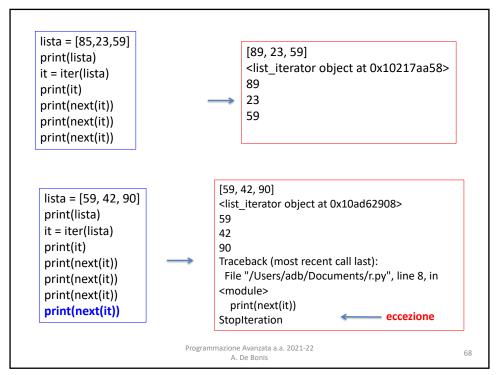
Introduzione a Python VII lezione

66

Iteratori

- Se una classe supporta l'iteratore possiamo ottenere un riferimento ad esso tramite la funzione iter()
 - Si invoca iter passando come argomento un'istanza della classe
- Per ottenere il prossimo elemento nella classe invochiamo next() sull'iteratore ottenuto
- Viene lanciata un'eccezione quando non ci sono più elementi nell'istanza della classe

Programmazione Avanzata a.a. 2021-22 A. De Bonis



68

Gestire l'eccezione [ista=[59, 42, 90] print(lista) it = iter(lista) print(it) while True: try: print(next(it)) except Exception as e: break

Gestire l'eccezione Ad una clausola try possono essere associate piu` clausole except.

- Una clausola except puo` includere piu` eccezioni (all'interno di una tupla)
 - esempio: ... except (RuntimeError, TypeError, NameError): pass
- Una clausola except che include un'eccezione di un certo tipo A puo` gestire solo eccezioni di tipo A o di sottoclassi di A.
- L'ultima clausola except puo` omettere i nomi delle eccezioni. Questo uso di except deve essere fatto con molta cautela perche' potrebbe nascondere un vero errore di programmazione. Puo` essere usato per stampare un messaggio di errore e lanciare nuovamente l'eccezione.
- Lo statement try ... except ha una clausola else opzionale che, quando presente, deve seguire tutte le clausole except. È utile per inserire codice che deve essere eseguito quando la clausola try non lancia` un' eccezione.
- Nello statement try puo` contenere una clausola finally che viene eseguita immediatamente prima che venga completato lo staement try. Questa clausola viene eseguita sia nel caso in cui il try produca un'eccezione sia nel caso non venga prodotta un'eccezione. (vedere esempi nella documentazione)

Programmazione Avanzata a.a. 2021-22 A. De Bonis

70

70

Gestire l'eccezione file exampleExceptions1.py import sys Se non viene lanciata f = open(sys.argv[2], sys.argv[1]) l'eccezione OSError, ogni print(sys.argv[2], 'has', len(f.readlines()), 'lines') altro tipo di eccezione viene f.close() catturare dalla seconda except OSError: clausola except print('cannot open', sys.argv[2]) except: print('errore inatteso') #viene rilanciata l'eccezione prodotta nel try raise eseguiamo lo script con python exampleExceptions1.py r filename se il file filename non esiste se il file filename esiste e contiene tre linee cannot open filename filename has 3 lines Programmazione Avanzata a.a. 2021-22 71

Gestire l'eccezione file exampleExceptions1.py import sys Se non viene lanciata f = open(sys.argv[2], sys.argv[1]) l'eccezione OSError, ogni print(sys.argv[2], 'has', len(f.readlines()), 'lines') altro tipo di eccezione viene f.close() catturare dalla seconda except OSError: clausola except print('cannot open', sys.argv[2]) except: print('errore inatteso') #viene rilanciata l'eccezione prodotta nel try eseguiamo lo script con python exampleExceptions1.py y filename errore inatteso Traceback (most recent call last): File "exampleExceptions1.py", line 5, in <module> f = open(arg, sys.argv[1]) ValueError: invalid mode: 'y' Programmazione Avanzata a.a. 2021-22 A. De Bonis

72

Gestire l'eccezione file exampleExceptions2.py import sys In questo esempio (dalla documentazione) la modalita` di for arg in sys.argv[1:]: apertura del file è fissata nel codice try: e sulla linea di comando vengono f = open(arg, 'r') passati i nomi dei file. except OSError: Se non viene lanciata l'eccezione print('cannot open', arg) OSError, vuol dire che il file è aperto in lettura e nella clausola else viene print(arg, 'has', len(f.readlines()), 'lines') stampato il numero di linee del file f.close() eseguiamo lo script con python exampleExceptions2.py file1 file2 file3 file1 e file3 non esistono; file2 esiste e contiene 4 linee cannot open file1 file2 has 4 lines cannot open file3 Programmazione Avanzata a.a. 2021-22

Lanciare un'eccezione

- Quando si lancia un'eccezione con raise, l'eccezione puo` essere creata invocando il
 costruttore della classe con o senza argomenti. Se non si specificano degli argomenti
 allora si puo` usare semplicemente il nome della classe (senza le parentesi tonde).
- Gli argomenti vengono memorizzati nella variabile args dell'eccezione istanziata
- In una clausola except il nome dell'eccezione puo` essere seguita da una variabile. In
 questo caso la variabile si lega all'istanza dell'eccezione con gli argomenti
 dell'eccezione memorizzati in .args. L'istanza dell'eccezione definisce __str__() in
 modo che gli argomenti possano essere stampati direttamente senza dover fare
 riferimento ad .args.
- Esempio presente nella documentazione:

```
>>> try:
... raise Exception('spam', 'eggs')
... except Exception as inst:
... print(type(inst)) # the exception instance
... print(inst.args) # arguments stored in .args
... print(inst) # __str__ allows args to be printed directly,
... # but may be overridden in exception subclasses
... print('x =', x)
... print('y =', x)
... print('y =', y)
...
(class 'Exception'>
('spam', 'eggs')
('spam', 'eggs')
('spam', 'eggs')
('spam', 'eggs')
X = spam
Y = eggs

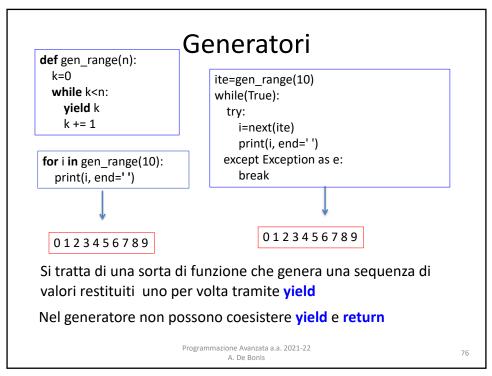
A. De Bonis
```

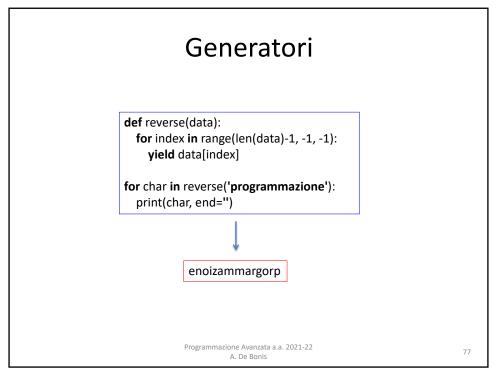
74

Generatori

- Una funzione generatore è un modo semplice ed immediato per creare un iteratore (detto generatore)
 - I metodi del generatore __iter__() e __next__() sono creati automaticamente
 - __iter__(): restituisce l'iteratore stesso
 - __next__(): viene utilizzato nei cicli per ottenere il prossimo elemento
- La sintassi per definire una funzione generatore è simile a quella usata per definire una funzione, ma al posto di return si usa yield
- Quando si incontra un yield l'esecuzione del generatore è sospesa, viene restituito il valore indicato da yield
- next() comincia l'esecuzione di un generatore o la riprende dall'ultima espressione yield eseguita ripartendo da dove l'esecuzione era stata sospesa

Programmazione Avanzata a.a. 2021-22





qualche dettaglio in piu`su iter()

- iter() restituisce un oggetto iteratore e puo` prendere in input uno o due argomenti
- In assenza del secondo argomento, il primo argomento deve essere un collezione che supporta l'iterazione. Nel caso la collezione non supporti il protocollo dell'iterazione viene lanciato TypeError.
- Se viene fornito anche il secondo argomento allora il primo argomento deve essere un callable. In questo caso l'iteratore restituito da iter() invoca il callable ogni volta che viene invocato il suo metodo __next__(). Il secondo argomento svolge il ruolo di sentinella e se next() restituisce un valore uguale a questo argomento viene lanciata l'eccezione Stoplteration.

Programmazione Avanzata a.a. 2021-22 A. De Bonis

78