alifazahra729 / **Workshop--python-** Public

Code Issues Pull requests Actions Projects Wiki Security Insights Se

master Workshop--python- / teori / **minggu-05** /

alifazahra000 minggu-05 ... 1 minute ago History

..

README.md 1 minute ago

README.md

Ringkasan :

- Bab 7 Ada beberapa cara untuk mempresentasikan output dari suatu program; data dapat dicetak dalam bentuk yang dapat dibaca manusia, atau ditulis ke file untuk digunakan di masa mendatang. Bab ini akan membahas beberapa kemungkinan.
- Untuk menggunakan literal string yang diformat, awali string dengan f atau F sebelum tanda kutip pembuka atau tanda kutip tiga. Sejauh ini kita menemukan dua cara penulisan nilai: pernyataan ekspresi dan fungsi print(). (Cara ketiga menggunakan metode write() objek file; file output standar dapat dirujuk sebagai sys.stdout.
- Metode string str.format() membutuhkan lebih banyak upaya manual. Anda masih akan menggunakan { dan } untuk menandai di mana variabel akan diganti dan dapat memberikan arahan pemformatan yang mendetail, tetapi Anda juga harus memberikan informasi yang akan diformat.
- Terakhir, Anda dapat melakukan semua penanganan string sendiri dengan menggunakan pemotongan string dan operasi penggabungan untuk membuat tata letak apa pun yang dapat Anda bayangkan. String, khususnya, memiliki dua representasi berbeda. Fungsi str() dimaksudkan untuk mengembalikan representasi nilai yang dapat dibaca oleh manusia, sedangkan repr() dimaksudkan untuk menghasilkan representasi yang dapat dibaca oleh penafsir (atau akan memaksa SyntaxError jika tidak ada sintaks yang setara).

Modul string berisi kelas Template yang menawarkan cara lain untuk mengganti nilai menjadi string, menggunakan placeholder seperti $x dan menggantinya dengan nilai dari kamus, tetapi menawarkan lebih sedikit kontrol pemformatan. Literal string terformat (disingkat juga disebut f-string) memungkinkan Anda menyertakan nilai ekspresi Python di dalam string dengan mengawali string dengan f atau F dan menulis ekspresi sebagai {ekspresi}. Penentu format opsional dapat mengikuti ekspresi. Ini memungkinkan kontrol yang lebih besar atas bagaimana nilai diformat. Meneruskan bilangan bulat setelah ':' akan menyebabkan bidang tersebut menjadi jumlah minimum karakter. Ini berguna untuk membuat kolom berbaris. Pengubah lain dapat digunakan untuk mengonversi nilai sebelum diformat. Lihat ekspresi mendokumentasikan diri untuk informasi lebih lanjut tentang = specifier. Untuk referensi tentang spesifikasi format ini, lihat panduan referensi untuk Bahasa Mini Spesifikasi Format. Tanda kurung dan karakter di dalamnya (disebut kolom format) diganti dengan objek yang diteruskan ke metode str.format() . Angka dalam tanda kurung dapat digunakan untuk merujuk ke posisi objek yang diteruskan ke metode str.format(). Jika argumen kata kunci digunakan dalam metode str.format() , nilainya dirujuk dengan menggunakan nama argumen. Jika Anda memiliki format string yang sangat panjang yang tidak ingin Anda pisahkan, alangkah baiknya jika Anda dapat mereferensikan variabel untuk diformat dengan nama, bukan dengan posisi. Ini dapat dilakukan hanya dengan meneruskan dict dan menggunakan tanda kurung siku '[]' untuk mengakses kunci.

Ini juga bisa dilakukan dengan mengirimkan kamus tabel sebagai argumen kata kunci dengan notasi **. Ini sangat berguna dalam kombinasi dengan fungsi bawaan vars(), yang mengembalikan kamus yang berisi semua variabel lokal. Untuk ikhtisar lengkap tentang pemformatan string dengan str.format(), lihat Format Sintaks String. (Note that the one space between each column was added by the way print() works: it always adds spaces between its arguments.) There is another method, str.zfill(), which pads a numeric string on the left with zeros. The str.rjust() method of string objects right-justifies a string in a field of a given width by padding it with spaces on the left. The % operator (modulo) can also be used for string formatting. Given 'string' % values, instances of % in string are replaced with zero or more elements of values. This operation is commonly known as string interpolation. Informasi lebih lanjut dapat ditemukan di bagian Pemformatan String gaya printf. open() mengembalikan objek file, dan paling sering digunakan dengan dua argumen posisional dan satu argumen kata kunci: open(filename, mode, encoding=None).

Argumen pertama adalah string yang berisi nama file. Data mode biner dibaca dan ditulis sebagai objek byte. 'r+' membuka file untuk membaca dan menulis. Jika Anda tidak menggunakan kata kunci with, maka Anda harus memanggil f. tutup untuk menutup file dan segera membebaskan semua sumber daya sistem yang digunakan olehnya. Panggilan Peringatan f. menulis tanpa menggunakan kata kunci with atau memanggil f. close mungkin menghasilkan argumen f. Contoh-contoh lainnya di bagian ini akan mengasumsikan bahwa sebuah objek file bernama f telah dibuat. Untuk membaca isi file, panggil f. readsize, yang membaca sejumlah data dan mengembalikannya sebagai string dalam mode teks atau objek byte dalam mode biner. size adalah argumen numerik opsional. Ketika ukuran dihilangkan atau negatif, seluruh isi file akan dibaca dan dikembalikan itu masalah Anda jika file tersebut dua kali lebih besar dari memori mesin Anda. F. readline membaca satu baris dari file, karakter baris baru n ditinggalkan di akhir string, dan hanya dihilangkan di baris terakhir file jika file tidak berakhir di baris baru. Ini membuat nilai pengembalian tidak ambigu jika f.

Untuk membaca baris dari file, Anda dapat mengulang objek file. Jika Anda ingin membaca semua baris file dalam daftar, Anda juga dapat menggunakan listf atau f. readlines. F. writestring menulis isi string ke file, mengembalikan jumlah karakter yang ditulis. F. kirim mengembalikan bilangan bulat yang memberikan posisi objek file saat ini dalam file yang direpresentasikan sebagai jumlah byte dari awal file saat dalam mode biner dan nomor buram saat dalam mode teks. Untuk mengubah posisi objek file, gunakan f. seekoffset, dari mana. Posisi dihitung dari penambahan offset ke titik referensi, titik referensi dipilih oleh argumen wherece. Dalam file teks yang dibuka tanpa b dalam string mode, hanya pencarian relatif ke awal file yang diperbolehkan, pengecualian sedang mencari hingga akhir file dengan seek0, 2 dan satu-satunya nilai offset yang valid adalah yang dikembalikan dari f. String dapat dengan mudah ditulis dan dibaca dari file. Angka membutuhkan sedikit usaha, karena metode read hanya mengembalikan string, yang harus diteruskan ke fungsi seperti int, yang mengambil string seperti 123 dan mengembalikan nilai numeriknya 123. Saat Anda ingin menyimpan tipe data yang lebih kompleks seperti daftar dan kamus bersarang, parsing dan serialisasi dengan tangan menjadi rumit. Daripada membuat pengguna terus-menerus menulis dan men-debug kode untuk menyimpan tipe data yang rumit ke file, Python memungkinkan Anda untuk menggunakan format pertukaran data populer yang disebut JSON JavaScript Object Notation. Modul standar yang disebut json dapat mengambil hierarki data Python, dan mengubahnya menjadi representasi string, proses ini disebut serialisasi.

Varian lain dari fungsi dumps, disebut dump, hanya membuat serialisasi objek ke file teks. Catatan File JSON harus dikodekan dalam UTF-8. Gunakan encodingutf-8 saat membuka file JSON sebagai file teks untuk membaca dan menulis. Teknik serialisasi sederhana ini dapat menangani daftar dan kamus, tetapi membuat serialisasi instance kelas arbitrer di JSON membutuhkan sedikit usaha ekstra. Referensi untuk modul json berisi penjelasan tentang ini. Lihat juga pickle - modul pickle Berlawanan dengan JSON, pickle adalah protokol yang memungkinkan serialisasi objek Python kompleks yang sewenang-wenang.

Give feedback