Modul 2 Praktikum Pemrograman Berbasis Fungsi

February 20, 2023

1 Dasar Functional Programming Python

Tujuan Praktikum: * Mempelajari cara menggunakan lambda, memetakan, memfilter, dan mengurangi fungsi di Python untuk mengubah struktur data. * Memahami penggunaan function build in, module, user define di functional programming python

Python menyediakan fitur seperti lambda, filter, map, dan reduce yang pada dasarnya dapat mencakup sebagian besar dari dasar-dasar Pemrograman Fungsional.

1.1 User Define Function

1.1.1 The Lambda Expression

Ekspresi Lambda - juga dikenal sebagai "fungsi anonim" - yang memungkinkan kita untuk membuat dan menggunakan fungsi dalam satu baris. fungsi ini berguna ketika kita membutuhkan fungsi pendek yang hanya akan digunakan sekali. Lambda sebagian besar digunakan bersama dengan map, filter dan metode pengurutan Mari kita tulis fungsi dengan Python, yang akan menghitung nilai 8x^2 + 2. Pendekatan standar adalah mendefinisikan fungsi.

[4]:
```
def f(x):
    """Function to compute the value of 8x+2"""
    return 8*x**2+2
f(3)
```

[4]:
```
74
```

Sekarang kita akan menghitung ekspresi yang sama menggunakan fungsi Lambda. Untuk membuat ekspresi lambda, kita mengetikkan kata kunci **lambda**, diikuti dengan input. Selanjutnya, kita memasukkan titik dua diikuti dengan ekspresi yang akan menjadi nilai kembali.

[5]:
```
lambda x: 8*x**2+2
```

[5]:
```
<function __main__.<lambda>(x)>
```

Fungsi lambda ini akan mengambil input x dan mengembalikan 8x^2 + 2, sama seperti fungsi sebelumnya f. Namun, ada masalah. **lambda** bukanlah nama fungsinya. di mana kata kunci Python yang mengatakan - berikut ini adalah fungsi anonim. Jadi bagaimana kita menggunakannya? Salah satu caranya adalah dengan memberinya nama. Mari kita sebut ungkapan lambda ini **g**. Sekarang, Anda dapat menggunakan ini seperti fungsi lainnya.

[6]:
```
g = lambda x: 8*x**2+2
g(3)
```

[6]:
```
74
```

1.1.2 Lambda expression with multiple inputs.

[8]:
```
# Calculating Harmonic Mean using lambda function
harmonic_mean = lambda x,y,z : 4/(2/x + 2/y + 2/z)**0.5
harmonic_mean(1,2,3)
```

[8]:
```
2.088931871468374
```

1.1.3 Lambda expression without inputs.

mari kita lihat penggunaan umum fungsi Lambda di mana kita tidak memberinya nama. Katakanlah kami memiliki daftar tujuh CEO terkenal di dunia dan kami ingin mengurutkan daftar ini berdasarkan nama belakang mereka. Kita akan membuat fungsi Lambda yang mengekstrak nama belakang, dan menggunakannya sebagai nilai penyortiran.

[21]:
```
# Sorting a List by thr last name using lambda expression
```

[9]:
```
presidents_usa = ["Tim Cook", "Sundar Pichai","Satya Nadella","Mark␣
↪Zukerberg","Elon Musk","Andy Jassy","Warren Buffet"]

presidents_usa.sort(key = lambda name: name.split(" ")[-1].lower())
presidents_usa
```

[9]:
```
['Warren Buffet',
 'Tim Cook',
 'Andy Jassy',
 'Elon Musk',
 'Satya Nadella',
 'Sundar Pichai',
 'Mark Zukerberg']
```

1.2 Built in Function

1.2.1 The Map Function

Fungsi **map** menerapkan fungsi ke setiap item yang dapat diulang dan menghasilkan hasil. Saat digunakan dengan daftar, map mengubah daftar tertentu menjadi daftar baru dengan menerapkan fungsi ke semua item dalam input_list.

Syntax

```
map(function_to_apply, iterables)
```

Usage Misalkan kita memiliki fungsi yang menghitung volume kubus, diberikan nilai tepinya adalah (a)

[10]:
```
def volume(a):
    """volumne of a cube with edge 'a'"""
    return a**3
```

Bagaimana jika kita mengkomputasikan volume dari kubus yang berbeda dengan panjang yang berbeda?

[11]:
```
# Edge length in cm
edges = [1,2,3,4,5]
```

Terdapat dua cara untuk menyelesaikannya. pertama, menggunakan `direct method` dan yang kedua menggunakan `map` function.

[12]:
```
# Calculating the volume of given cubes using Direct Method

volumes = []
for a in edges:
    v = volume(a)
    volumes.append(v)


volumes
```

[12]:
```
[1, 8, 27, 64, 125]
```

Sekarang periksa bagaimana hanya satu baris code dengan map function

[13]:
```
# Calculating the volume of given cubes using the Map function

map(volume,edges)
```

[13]:
```
<map at 0x24066109390>
```

Fungsi map mengambil dua argumen. Yang pertama adalah fungsi, dan yang kedua adalah daftar, tupel, atau objek lain yang dapat diulang. Di sini, fungsi map menerapkan fungsi volume ke setiap elemen dalam daftar.

Namun, hal penting yang perlu diperhatikan di sini adalah bahwa output dari fungsi map bukanlah daftar tetapi objek peta, yang sebenarnya merupakan iterator atas hasil yang dihitung. akan tetapi, kita dapat mengubahnya menjadi daftar dengan meneruskan map ke konstruktor daftar.

[14]:
```
list(map(volume,edges))
```

[14]:
```
[1, 8, 27, 64, 125]
```

Example Sekarang mari kita lihat contoh yang menunjukkan penggunaan fungsi 'lambda' dengan fungsi 'map'. misalkan kita memiliki daftar tupel yang berisi nama dan tinggi untuk 5 orang. Masing-masing tingginya dalam sentimeter dan kita perlu mengubahnya menjadi kaki.

Pertama-tama kita akan menulis fungsi konverter menggunakan ekspresi lambda yang akan menerima tuple sebagai input dan akan mengembalikan tuple dengan nama yang sama.

[16]:
```
# Convert height from cms to feet : 1 cm = 0.0328 feet
height_in_cms =␣
 ↪[('Riski',183),('Fathir',171),('Rafi',179),('Salwa',190),('Wulan',165)]

#Writing Convertor function using lambda expression
height_in_feet = lambda data: (data[0],round(data[1]*0.0328,1))

#Using the 'Map' function
list(map(height_in_feet,height_in_cms))
```

[16]:
```
[('Riski', 6.0),
 ('Fathir', 5.6),
 ('Rafi', 5.9),
 ('Salwa', 6.2),
 ('Wulan', 5.4)]
```

1.2.2 The Filter Function

Fungsi `filter` membangun iterator dari elemen-elemen yang dapat diulang yang fungsinya mengembalikan true. Ini berarti fungsi filter digunakan untuk memilih potongan data tertentu dari daftar, tupel, atau kumpulan data lainnya, maka namanya.

Syntax

`filter(filter(function, iterable)`

Usage Periksa contoh ini di mana kita ingin mendapatkan list dari semua angka yang lebih dari 5, yang diberikan input sebagai berikut:

[17]:
```
# Filter out all the numbers greater than 5 from a list

my_list = [1,2,3,4,5,6,7,8,9]
output_list = filter(lambda x : x>5, my_list)

list(output_list)
```

[17]:
```
[6, 7, 8, 9]
```

Example Berikut adalah daftar yang berisi beberapa Provinsi di Indonesia. Perhatikan ada banyak string yang kosong. Kita akan menggunakan fungsi filter untuk menghapus nilai-nilai yang hilang ini.

[19]:
```
# Removing missing values from a list

```

```
countries_asia = ["Nanggroe Aceh Darussalam","","Jambi","","Papua␣
 ↪Barat","","Banten","","","Maluku"]

list(filter(None,countries_asia))
```

[19]:
```
['Nanggroe Aceh Darussalam', 'Jambi', 'Papua Barat', 'Banten', 'Maluku']
```

Fungsi ini menyaring semua nilai yang diperlakukan sebagai false dalam pengaturan boolean.

1.2.3 The Reduce Function

Fungsi `Reduce` agak tidak biasa dan pada kenyataannya, pada Python 3, bukan lagi termasuk fungsi bawaan (built-in function). dan telah dipindahkan ke modul functools.

Fungsi `reduce` mengubah daftar yang diberikan menjadi satu nilai dengan menerapkan fungsi secara kumulatif ke item urutan, dari kiri ke kanan,

Syntax

`reduce(func, seq)`

di mana reduce terus menerapkan fungsi func() ke urutan seq dan mengembalikan satu nilai.

Usage Mari kita ilustrasikan cara kerja fungsi reduce dengan bantuan contoh sederhana yang menghitung produk dari daftar bilangan bulat.

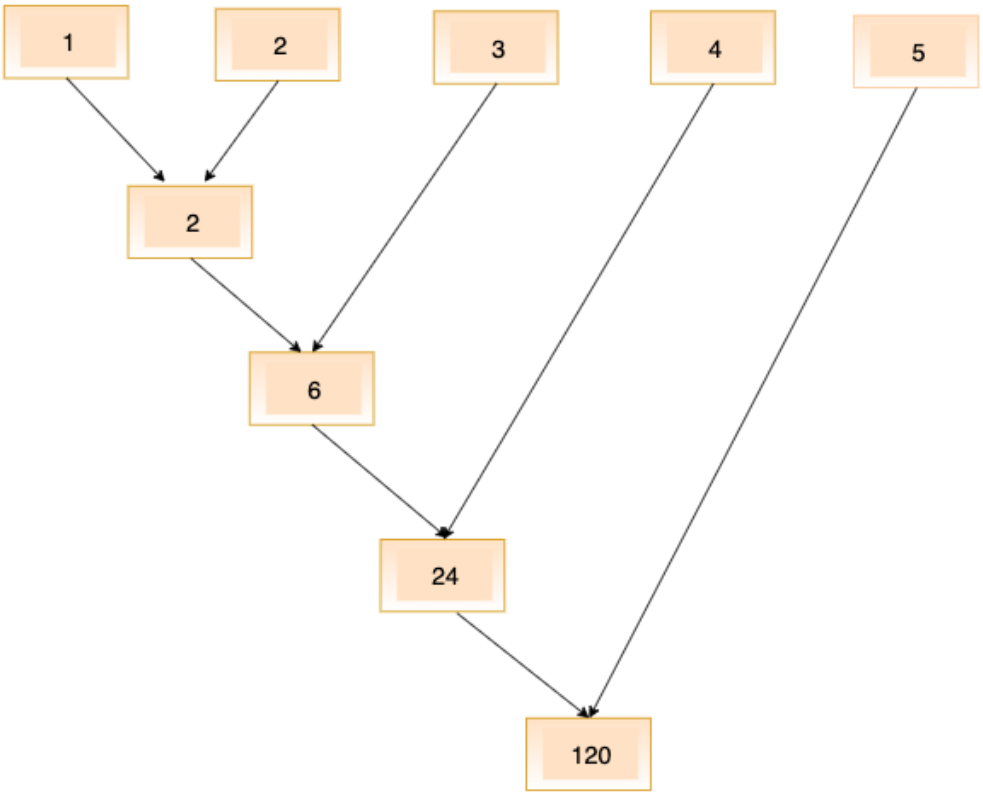
[20]:
```
# Compute the product of a list of integers using 'reduce'function

from functools import reduce
product = reduce((lambda x,y : x*y), [1,2,3,4,5])

product
```

[20]:
```
120
```

The following diagram shows the intermediate steps of the calculation:



Program di atas juga dapat ditulis secara eksplisit dengan loop agar lebih jelas. Oleh karena itu Gunakan functools.reduce jika Anda benar-benar membutuhkannya

[21]:
```
# Compute the product of a list of integers using a 'For' loop

product = 1
list = [1,2,3,4,5]
for num in list:
    product = product*num

product
```

[21]: 120

Example Fungsi `reduce` dapat menentukan maksimum daftar yang berisi bilangan bulat dalam satu baris kode. Memang ada fungsi bawaan yang disebut `max()` di Python yang biasanya digunakan untuk tujuan ini sebagai `max(list_name)`.

[22]:
```
# Determining the maximum number in a given list

from functools import reduce
f = lambda a,b : a if (a>b) else b
reduce(f,[58,69,12,158,698])
```

[22]: 698

1.2.4 List Comprehensions: Alternative to map, filter and reduce

List comprehension adalah cara untuk menentukan dan membuat list dengan Python. Dalam kebanyakan kasus, list comprehensions memungkinkan kita membuat daftar dalam satu baris kode tanpa khawatir tentang menginisialisasi daftar atau menyiapkan loop. Langkah ini juga merupakan pengganti fungsi lambda serta fungsi `map()`, `filter()` dan `reduce()`. Beberapa orang menganggapnya sebagai cara yang lebih pythonic untuk menulis fungsi dan merasa lebih mudah untuk memahaminya.

Syntax

[23]:
```
# Creating a list of squares of first 10 numbers using loops

squares = []
for x in range(10):
    squares.append(x**2)

squares
```

[23]: [0, 1, 4, 9, 16, 25, 36, 49, 64, 81]

Sekarang coba list comprehension untuk mendapatkan hasil yang sama dalam satu baris

[25]:
```
# # Creating a list of squares of first 10 using list comprehension

squares = [x**2 for x in range(10)]
squares
```

[25]: [0, 1, 4, 9, 16, 25, 36, 49, 64, 81]

Usage Coba contoh sebelumnya dengan list comprehensions.

List Comprehensions vs Map function

[26]:
```
# Convert height from cms to feet using List Comprehension : 1 cm = 0.0328 feet
height_in_cms =␣
 ↪[('Riski',183),('Fathir',171),('Maria',179),('Wulan',190),('Salwa',165)]

height_in_feet = [(height[0],round(height[1]*0.0328,1)) for height in␣
 ↪height_in_cms]

height_in_feet
```

[26]:
```
[('Riski', 6.0),
 ('Fathir', 5.6),
 ('Maria', 5.9),
 ('Wulan', 6.2),
 ('Salwa', 5.4)]
```

List Comprehensions vs Filter function

[27]:
```
# Removing missing values from a list

countries_asia =␣
 ↪["Afghanistan","","Bhutan","","China","","Georgia","","","India"]
[country for country in countries_asia if country!=""]
```

[27]:
```
['Afghanistan', 'Bhutan', 'China', 'Georgia', 'India']
```

List Comprehensions vs Reduce function

[49]:
```
# Determining the maximum number in a given list

numbers = [58,69,12,158,698,956]
max((x) for x in numbers)
```

[49]:
```
956
```

1.3 Modul dengan Fungsi

Pada bagian ini kita akan membuat sebuah modul dari serangkaian fungsi dengan fungsional programming.

1.3.1 Prosedural Programming

```
def factorial(i):
    for x in range(1,i):
        i*=x
    return i if i != 0 else 1
def pow(a, b):
    c = 1
    for _ in range(b):
        c *= a
    return c
def sqrt(n):
    if n < 0:
        raise ValueError("Cannot use negative numbers.")
    if n == 0:
        return 0
    x=1
    for _ in range(1,1001):
        x=x-((x**2-n)/(2*x))
    return x
def sqrt_2(n):
    if n < 0:
        raise ValueError("Cannot use negative numbers.")
    if n == 0:
        return 0
    m = 1
```

```
    for _ in range(1,1001):
        m = (m+float(n/m))*1.0/2.0
    return m
def sin(x):
    y, s = x,-1
    for i in range(3, 100,2):
        y+=s*(pow(x,i)/factorial(i))
        s *= -1
    return y
def cos(x):
    y, s = 1, -1
    for i in range(2,100,2):
        y+=s*(pow(x,i)/factorial(i))
        s *= -1
    return y
def sum(a):
    m=0
    for i in range(len(a)):
        m+=a[i]
    return
```

1.4 References

- Don't Be Scared Of Functional Programming
- Functional Programming HOWTO
- Github Math module