



Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	Samuel Natanael
Nama Lengkap	71231050
Minggu ke / Materi	13 / Set

SAYA MENYATAKAN BAHWA LAPORAN PRAKTIKUM INI SAYA BUAT DENGAN USAHA SENDIRI TANPA MENGGUNAKAN BANTUAN ORANG LAIN. SEMUA MATERI YANG SAYA AMBIL DARI SUMBER LAIN SUDAH SAYA CANTUMKAN SUMBERNYA DAN TELAH SAYA TULIS ULANG DENGAN BAHASA SAYA SENDIRI.

SAYA SANGGUP MENERIMA SANKSI JIKA MELAKUKAN KEGIATAN PLAGIASI, TERMASUK SANKSI TIDAK LULUS MATA KULIAH INI.

PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS KRISTEN DUTA WACANA
YOGYAKARTA
2024

BAGIAN 1: MATERI MINGGU INI (40%)

MATERI 1

Set adalah tipe data Python yang digunakan untuk menyimpan kumpulan data unik. Berikut beberapa karakteristik dari Set:

1. Elemen dalam Set disebut sebagai anggota (member).
2. Anggota Set harus bersifat immutable, seperti integer, float, string, dan tuple. Tipe data mutable seperti list dan dictionary tidak dapat menjadi anggota Set.
3. Set sendiri bersifat mutable, artinya Anda dapat menambah atau menghapus anggota dari Set. Karena itu, Set tidak bisa menjadi anggota dari Set lain.

Untuk mendefinisikan Set, Anda bisa menggunakan notasi kurung kurawal {} atau fungsi set(). Berikut adalah contohnya:

1. Untuk mendefinisikan Set yang kosong, harus menggunakan fungsi set() karena menggunakan notasi {} akan menghasilkan dictionary kosong.
2. Ketika menggunakan fungsi set(), Anda harus memasukkan iterable (seperti list) sebagai argumen jika ingin menambahkan beberapa anggota sekaligus.

```
# Membuat set dengan notasi {}
huruf = {"a", "b", "c"}
bilangan = {1, 2, 3}

# Dengan menggunakan fungsi set()
bilangan = [1, 2, 3]
print(type(bilangan))
# <class 'list'>

bilangan = set(bilangan)
print(type(bilangan))
# <class 'set'>

# Membuat set kosong harus menggunakan set()
kosong = set()

# Jika menggunakan notasi {}, maka akan menjadi tipe data dict
dict_kosong = {}
print(type(dict_kosong))
# <class 'dict'>
```

Pengaksesan Set

Set tidak memiliki indeks, sehingga Anda tidak bisa mengakses anggotanya secara langsung menggunakan indeks. Perhatikan contoh program berikut:

```
# Mencari jumlah item dalam set
presiden = {"Jokowi Dodo", "Prabowo Subianto", "Soekarno"}
jumlah_pres = len(presiden)
print(jumlah_pres)

# Print satu satu
for x in presiden:
    print(x)

# 3
# Soekarno
# Jokowi Dodo
# Prabowo Subianto
```

Jika diperhatikan, urutan output yang dihasilkan berbeda dari urutan Set awal. Hal ini dikarenakan Set tidak memiliki indeks, sehingga anggotanya tidak memiliki urutan. Pada tipe data Set, posisi anggota tidaklah penting.

Set adalah tipe data yang mutable, artinya Anda bisa menambah atau mengurangi anggotanya. Program berikut menunjukkan cara menambah anggota ke sebuah Set menggunakan fungsi `add()`:

```
# Membuat set kosong
nama = set()

# Menambahkan item ke dalam set menggunakan fungsi add()
nama.add("SAMUEL")
nama.add("ALDO")
nama.add("RIAN")

# Jumlah data dalam nama
print(len(nama))

# Print satu-satu
for x in nama:
    print(x)

# 3
# RIAN
# ALDO
# SAMUEL
```

Set memiliki mekanisme untuk memeriksa apakah anggota yang akan dimasukkan sudah ada di dalam Set. Jika anggota tersebut tidak ada dalam set, maka akan ditambahkan ke dalam Set. Namun, jika sudah ada, fungsi `add()` tidak akan menambahkannya lagi. Pengecekan duplikasi ini dilakukan secara otomatis oleh fungsi `add()`, sehingga Anda tidak perlu melakukannya secara manual.

Python Cheat Sheet: Set Methods

"A puzzle a day to learn, code, and play" → Visit [finxter.com](https://blog.finxter.com)

Method	Description	Example
<code>set.add(x)</code>	Add an element to this set	<pre>>>> s = {1, 2, 3} >>> s.add(4) # {1, 2, 3, 4}</pre>
<code>set.clear()</code>	Remove all elements from this set	<pre>>>> s = {1, 2, 3} >>> s.clear() # set()</pre>
<code>set.copy()</code>	Create and return a flat copy of this set	<pre>>>> s = {1, 2, 'Alice'} >>> s.copy() # Returns: {1, 2, 'Alice'}</pre>
<code>set.difference(x)</code>	Return a new set with elements of this set except the ones in the given set arguments.	<pre>>>> {1, 2, 3}.difference({3, 2}) {1}</pre>
<code>set.difference_update(iter)</code>	Remove all elements from this set that are members of any of the given set arguments.	<pre>>>> s = {1, 2, 3} >>> s.difference_update({1, 2}) # s == {3}</pre>
<code>set.discard(x)</code>	Remove an element from this set if it is a member, otherwise do nothing.	<pre>>>> s = {'Alice', 'Bob', 'Cloe'} >>> s.discard('Bob') # s == {'Alice', 'Cloe'}</pre>
<code>set.intersection()</code>	Return a new set of elements that are members of this and the set argument(s).	<pre>>>> {1, 2, 3, 4}.intersection({3, 4, 5}) {3, 4}</pre>
<code>set.intersection_update()</code>	Removes all elements from this set that are not members in all other specified sets.	<pre>>>> s = {1, 2, 3, 4} >>> s.intersection_update({3, 4, 5}) # s == {3, 4}</pre>
<code>set.isdisjoint(x)</code>	Return True if their intersection is the empty set.	<pre>>>> {1, 2, 3, 4}.isdisjoint({'Alice', 'Bob'}) True</pre>
<code>set.issubset()</code>	Return True if all elements of this set are members of the specified set argument.	<pre>>>> t = {'Alice', 'Bob', 'Carl', 'Tiz'} >>> {'Alice', 'Bob'}.issubset(t) True</pre>
<code>set.issuperset()</code>	Return True if all elements of the specified set argument are members of this set.	<pre>>>> {'Alice', 'Bob', 'Carl'}.issuperset({'Alice'}) True</pre>
<code>set.pop()</code>	Remove and return a random element from this set. <code>KeyError</code> if set is empty.	<pre>>>> s = {'Alice', 'Bob', 'Carl'} >>> s.pop() 'Alice'</pre>
<code>set.remove()</code>	Remove and return a specific element from this set as defined in the argument. If the set doesn't contain element, raise <code>KeyError</code> .	<pre>>>> s = {'Alice', 'Bob', 'Cloe'} >>> s.remove('Bob') # s == {'Alice', 'Cloe'}</pre>
<code>set.symmetric_difference()</code>	Return new set with elements in either this or the specified set argument, but not both.	<pre>>>> {1, 2, 3}.symmetric_difference({2, 3, 4}) {1, 4}</pre>
<code>set.symmetric_difference_update()</code>	Replace this set with the symmetric difference, i.e., elements in either this set or the specified set argument, but not both.	<pre>>>> s = {1, 2, 3} >>> s.symmetric_difference_update({2, 3, 4}) >>> s {1, 4}</pre>
<code>set.union()</code>	Create and return new set with all elements in this or any of the specified sets.	<pre>>>> {1, 2, 3, 4}.union({3, 4, 5}) {1, 2, 3, 4, 5}</pre>
<code>set.update()</code>	Update this set with all elements that are in this or any of the specified set arguments.	<pre>>>> s = {1, 2, 3, 4} >>> s.update({3, 4, 5}) # s == {1, 2, 3, 4, 5}</pre>



Gambar 13.1 Methods dalam Set (diambil dari <https://blog.finxter.com/python-set-methods/>)

```

# 4 Fungsi untuk menghapus anggota dari set
# remove(), menghapus 5, akan menghasilkan error jika tidak ada 5 dalam set
bilangan = {1,2,3,4,5,6,7,8,9,0}
bilangan.remove(5)
print(f'Remove 5 = {bilangan}')

# discard(), sama seperti remove(), tetapi tidak memunculkan error
bilangan = {1,2,3,4,5,6,7,8,9,0}
bilangan.discard(10)
print(f'Discard 10 = {bilangan}')

# pop(), menghapus secara random
bilangan = {1,2,3,4,5,6,7,8,9,0}
bilangan.pop()
print(f'pop = {bilangan}')

# clear(), Menghapus seluruh elemen
bilangan = {1,2,3,4,5,6,7,8,9,0}
bilangan.clear()
print(f'clear = {bilangan}')

# Remove 5 = {0, 1, 2, 3, 4, 6, 7, 8, 9}
# Discard 10 = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
# pop = {1, 2, 3, 4, 5, 6, 7, 8, 9}
# clear = set()

```

Untuk mengubah nilai salah satu anggota di dalam Set, yang dapat dilakukan adalah mengganti anggota tersebut dengan cara menghapus anggota yang ingin diubah, kemudian menambahkan anggota baru dengan nilai yang diinginkan.

```

# set bilangan
bilangan = {1,2,3,4,5,6,7,8,9}
print(bilangan)

# remove 1 dan 2 dari bilangan
bilangan.remove(1)
bilangan.remove(2)

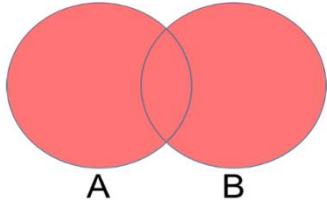
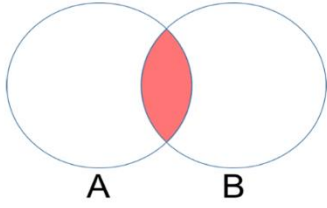
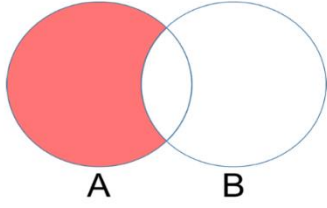
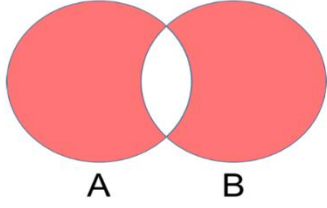
# Menambahkan 10 dan 11 ke bilangan
bilangan.add(10)
bilangan.add(11)

print(bilangan)

```

Setiap kali ada operasi penambahan dan penghapusan maka urutan anggota di dalam Set biasanya akan berubah.

Operasi-Operasi pada Set

Set Operation	Venn Diagram	Interpretation
Union		$A \cup B$, is the set of all values that are a member of A , or B , or both.
Intersection		$A \cap B$, is the set of all values that are members of both A and B .
Difference		$A \setminus B$, is the set of all values of A that are not members of B
Symmetric Difference		$A \triangle B$, is the set of all values which are in one of the sets, but not both.

Gambar 13.2 Operasi-operasi di Set (diambil dari

<https://www.facebook.com/MathTeacherGon/photos/a.2171943986399575/3031840407076591/?type=3>)

```
# Set bilangan
bilangan1 = {1,3,4,5,6,7,9}
bilangan2 = {2,5,6,8}

# Union
union = bilangan1 | bilangan2
# atau
union = bilangan1.union(bilangan2)

# Intersection
intersection = bilangan1 & bilangan2
```

```
# atau
intersection = bilangan1.intersection(bilangan2)

# Difference
diff1 = bilangan1 - bilangan2 # hanya ada di bilangan 1
diff2 = bilangan2 - bilangan1 # hanya ada di bilangan 2

# Symmetric Difference
SymDiff = bilangan1 ^ bilangan2

print(union) {1, 2, 3, 4, 5, 6, 7, 8, 9}
print(intersection) {5, 6}
print(diff1) {1, 3, 4, 7, 9}
print(diff2) {8, 2}
print(SymDiff) {1, 2, 3, 4, 7, 8, 9}
```

BAGIAN 2: LATIHAN MANDIRI (60%)

Pada bagian ini anda menuliskan jawaban dari soal-soal Latihan Mandiri yang ada di modul praktikum. Jawaban anda harus disertai dengan source code, penjelasan dan screenshot output.

SOAL 1

```
12,1.py > ...
1  # Meminta input jumlah kategori
2  n = int(input('Masukkan jumlah kategori : '))
3  data = {}
4
5  # Looping untuk memasukan nama kategori
6  for i in range(n):
7      nama_kategori = input(f'Masukkan nama kategori ke-{i+1} : ')
8      print(f'Masukkan 5 Aplikasi di Kategori {nama_kategori}')
9
10     # Looping untuk memasukan nama aplikasi
11     aplikasi = []
12     for j in range(5):
13         nama_aplikasi = input(f>Nama Aplikasi ke-{j+1}: ')
14         aplikasi.append(nama_aplikasi)
15
16     # Memasukan aplikasi ke dalam kategori
17     data[nama_kategori] = aplikasi
18
19 print()
20 print("Daftar Aplikasi")
21
22 # Menampilkan seluruh dictionary
23 for x in data:
24     print(f'{x} = {data.get(x)}')
25
26 daftar_aplikasi_list = []
27
28 # ambil semua daftar aplikasi dari setiap kategori
29 for aplikasi in data.values():
30     daftar_aplikasi_list.append(set(aplikasi))
31
32 # lakukan intersection ke semua set yang ada
33 hasil = daftar_aplikasi_list[0]
34 for i in range(1, len(daftar_aplikasi_list)):
35     hasil = hasil.intersection(daftar_aplikasi_list[i])
36
37 # Menampilkan aplikasi yang muncul di setiap kategori
38 print()
39 print(("Aplikasi yang hanya muncul di setiap kategori :"))
40 for x in hasil:
41     print(x)
```

```

41 |     print(x)
42 | print()
43 |
44 | set_app = set()
45 |
46 | # Memasukan semua aplikasi ke dalam set
47 | for x in data.values():
48 |     for aplikasi in x:
49 |         set_app.add(aplikasi)
50 |
51 | satu = []
52 | dua = []
53 |
54 | # Looping setiap aplikasi untuk dihitung jumlah kemunculannya
55 | for x in set_app:
56 |     count = 0
57 |     for kategori in data:
58 |         # Jika didalam kategori tersebut ada aplikasi itu, maka count akan bertambah
59 |         if x in data[kategori]:
60 |             count += 1
61 |     # Memasukan aplikasi ke dalam list sesuai jumlah kemunculannya
62 |     if count == 2:
63 |         dua.append(x)
64 |     elif count == 1:
65 |         satu.append(x)
66 |
67 | # Print hasil
68 | print("Aplikasi yang hanya muncul di satu kategori saja:")
69 | for x in satu:
70 |     print(x)
71 |
72 | if n > 2:
73 |     print("\nAplikasi yang hanya muncul tepat di dua kategori sekaligus:")
74 |     for x in dua:
75 |         print(x)

```

```
Masukkan jumlah kategori : 3
Masukkan nama kategori ke-1 : a
Masukkan 5 Aplikasi di Kategori a
Nama Aplikasi ke-1: 1
Nama Aplikasi ke-2: 2
Nama Aplikasi ke-3: 3
Nama Aplikasi ke-4: 4
Nama Aplikasi ke-5: 5
Masukkan nama kategori ke-2 : b
Masukkan 5 Aplikasi di Kategori b
Nama Aplikasi ke-1: 1
Nama Aplikasi ke-2: 6
Nama Aplikasi ke-3: 7
Nama Aplikasi ke-4: 8
Nama Aplikasi ke-5: 9
Masukkan nama kategori ke-3 : c
Masukkan 5 Aplikasi di Kategori c
Nama Aplikasi ke-1: 1
Nama Aplikasi ke-2: 2
Nama Aplikasi ke-3: 3
Nama Aplikasi ke-4: 6
Nama Aplikasi ke-5: 9

Daftar Aplikasi
a = ['1', '2', '3', '4', '5']
b = ['1', '6', '7', '8', '9']
c = ['1', '2', '3', '6', '9']

Aplikasi yang hanya muncul di setiap kategori :
1

Aplikasi yang hanya muncul di satu kategori saja:
8
4
5
7

Aplikasi yang hanya muncul tepat di dua kategori sekaligus:
3
2
6
9
```

SOAL 2

```
12,2.py > ...
1  # Data
2  List = [1,2,3,4,5]
3  Set = {1,2,3,4,5}
4  Tuple = (1,2,3,4,5)
5
6  # Mengubah data menjadi tipe data yang diinginkan menggunakan set(), list() atau tuple()
7  listset = set(List)
8  setlist = list(Set)
9  tupset = set(Tuple)
10 settup = tuple(Set)
11
12 # Print semuanya
13 print("List :", List)
14 print("List => Set:", listset)
15 print()
16
17 print("Set :", Set)
18 print("Set => List:", setlist)
19 print()
20
21 print("Tuple :", Tuple)
22 print("Tuple => Set:", tupset)
23 print()
24
25 print("Set :", Set)
26 print("Set => Tuple:", settup)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
List : [1, 2, 3, 4, 5]
List => Set: {1, 2, 3, 4, 5}

Set : {1, 2, 3, 4, 5}
Set => List: [1, 2, 3, 4, 5]

Tuple : (1, 2, 3, 4, 5)
Tuple => Set: {1, 2, 3, 4, 5}

Set : {1, 2, 3, 4, 5}
Set => Tuple: (1, 2, 3, 4, 5)
```

SOAL 3

```
12,3.py > ...
1  # Tampilkan pesan error jika file tidak ditemukan/tidak bisa dibaca.
2  def cari_file(file):
3      try:
4          with open(file, 'r') as file:
5              lines = file.read().lower()
6              # Mengembalikan kalimat-kalimat dalam file
7              return lines
8      except:
9          print(f"File '{file}' tidak dapat ditemukan/dibaca.")
10         exit()
11
12  file1 = str(input("Masukkan nama file pertama: ")) # 12/1.txt
13  lines1 = cari_file(file1)
14  file2 = str(input("Masukkan nama file kedua: ")) # 12/2.txt
15  lines2 = cari_file(file2)
16
17  # Mengsplit kalimat menjadi kata dan diubah menjadi set sehingga tidak berisikan kata berulang
18  kata1 = set(lines1.split())
19  kata2 = set(lines2.split())
20
21  # Digabung menjadi satu
22  kata = kata1.union(kata2)
23  print(kata)
24
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\Ngampus\Semester 2\PrakAlPro> & D:/Python/python.exe "d:/Ngampus/Semester 2/PrakAlPro/12,3.py"
Masukkan nama file pertama: 12/1.txt
Masukkan nama file kedua: 12/2.txt
```

Github : <https://github.com/SamuelN1508/PrakAlPro12.git>