9. Tipe Data List

9.1 Tujuan Praktikum

Setelah mempelajari Bab ini, mahasiswa diharapkan dapat:

1. Dapat memahami sifat-sifat list dan dapat mendefinisikan list.
2. Dapat memahami dan melakukan operasi-operasi CRUD (Create, Read, Update dan Delete) pada list.
3. Dapat memahami dan menggunakan fungsi-fungsi Python yang berkaitan dengan list.
4. Dapat memahami dan menggunakan list sebagai parameter fungsi.

9.2 Alat dan Bahan

Praktikum ini membutuhkan perangkat komputer yang memiliki spesifikasi minimum sebagai berikut:

1. Terkoneksi ke Internet dan dapat mengunduh package-package Python.
2. Mampu menjalankan sistem operasi Windows 10 atau Ubuntu Linux.

Perangkat lunak yang diperlukan untuk mendukung praktikum ini adalah sebagai berikut:

1. Python 3.7 atau 3.8 yang terinstall menggunakan Anaconda atau Installer Python lainnya.
2. Web Browser (Mozilla Firefox, Microsoft Edge atau Google Chrome).
3. Command Prompt (jika menggunakan Windows).
4. Terminal (jika menggunakan Linux).
5. Editor Python (Visual Studio Code, PyCharm, Spyder atau editor-editor lainnya yang mendukung Python).

9.3 Materi

9.3.1 Sifat-sifat List

List pada Python adalah rangkaian nilai-nilai yang dapat diakses menggunakan satu nama tunggal. Apa bedanya dengan String? String adalah rangkaian dari karakter-karakter, sedangkan

list dapat berisi karakter, integer, float maupun tipe data lainnya. List juga bisa berisi list lainnya. Rangkaian nilai-nilai tersebut dituliskan di dalam **[]** seperti contoh berikut ini:

```
nilai_ujian = [80,75,70,90,81,84,92,71,65,80,70]
nama_pahlawan = ['Sukarno', 'Diponegoro', 'Jend. Sudirman', 'Cut Nya Dhien']
nilai_campuran = ['Javascript', 20, 34.4, True]
list_dalam_list = [23, [22, 20], 45]
```

List juga memiliki perbedaan lain dengan String, yaitu list bersifat mutable, sedangkan String bersifat immutable. Mutable artinya nilainya dapat diubah secara langsung, seperti yang ditunjukkan pada kode program berikut ini:

```
# definisikan list berisi 4 nilai
data = [10,20,30,40]
# ubah nilai index ke-0 menjadi 50
data[0] = 50
# isinya sekarang: 50, 20, 30, 40
print(data)

# definisikan sebuah string
nama = 'Yuan Lukito'
# ubah karakter index ke-0 menjadi Z
nama[0] = 'Z'
# tidak akan mencapai baris ini karena muncul error
# TypeError: 'str' object does not support item assignment
print(nama)
```

Perbedaan berikutnya, jika ada dua string yang isinya sama, keduanya menunjuk (merujuk/mengacu) pada object yang sama. Sedangkan pada list jika ada dua list dengan isi yang sama, keduanya menunjuk pada object yang berbeda. Hal tersebut dapat ditunjukkan dengan Python shell berikut ini:

```
>>> a = 'banana'
>>> b = 'banana'
>>> a is b
True

>>> a = [1, 2, 3]
>>> b = [1, 2, 3]
>>> a is b
False
```

9.3.2 Operasi isi List

Beberapa operasi yang dapat digunakan dalam memproses sebuah list, ialah:

1. Penambahan elemen pada list dengan operator +:

```
>>> bil1 = [1,2,3,4]
>>> bil2 = [5,6,7]
>>> bilTotal = bil1 + bil2
>>> print(bilTotal)
[1, 2, 3, 4, 5, 6, 7]
```

2. Perulangan elemen pada list dengan operator *:

```
>>> bil1 = [1,2,3,4]
>>> bilTotal = [1,2,3,4]*2
>>> print(bilTotal)
[1, 2, 3, 4, 1, 2, 3, 4]
```

3. Pengaksesan pada elemen list: Elemen pada list dapat diakses dalam bentuk per elemen maupun sekelompok elemen. Pengaksesan list menggunakan indeks dari elemen tersebut. Indeks pada list dimulai dari indeks 0.

```
>>> nama = ["kuncoro","anton","dida","yuan"]
>>> nama[0]
'kuncoro'
>>> nama[3]
'yuan'

#indeks akses dihitung dari belakang
>>> nama[-2]
'dida'
>>> nama[-3]
'anton'
>>> nama[-4]
'kuncoro'
```

Contoh di atas memperlihatkan sebuah list berisi 4 buah elemen nama. Pengaksesan dilakukan dengan variabel list[indeksElemen]. Jika indeks elemen yang dimasukkan tidak ada dalam list tersebut, maka pyhton akan mengeluarkan error seperti berikut ini:

```
>>> nama[4]
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
IndexError: list index out of range
```

Elemen nama[4] pada list tersebut tidak ada, karena pada list **nama** nomor indeks dimulai dari 0 sampai 3. Pengaksesan list juga dapat dilakukan untuk banyak indeks.

```
# Seluruh elemen diakses
>>> nama[:]
['kuncoro', 'anton', 'dida', 'yuan']
# elemen diakses pada indeks 1-2
>>> nama[1:3]
['anton', 'dida']
# elemen diakses dari indeks awal hingga 2
>>> nama[:3]
['kuncoro', 'anton', 'dida']
# elemen diakses dari indeks 1 hingga akhir
>>> nama[1:]
['anton', 'dida', 'yuan']
```

4. Penggantian nilai pada elemen list: Penggantian nilai pada sebuah list juga dapat dilakukan dengan mengakses banyak elemen.

```
#Penggantian pada indeks elemen 1 dan 2
>>> nama[1:3] = ["felix","ryan"]
```

```
>>> nama
['kuncoro', 'felix', 'ryan', 'yuan']
```

9.3.3 Metode dan Fungsi Untuk List

Python memeberikan beberapa metode yang dapat digunakan untuk melakukan operasi pada sebuah list. Beberapa metode tersebut ialah:

1. **append** : metode yang digunakan untuk menambahkan elemen baru dan dianggap sebagai kesatuan objek pada bagian akhir list

```
>>> nama
['kuncoro', 'felix', 'ryan', 'yuan']
>>> nama.append(['bejo','tejo'])
>>> nama
['kuncoro', 'felix', 'ryan', 'yuan', ['bejo','tejo']]
```

2. **extend** : metode yang digunakan untuk menambahkan elemen pada sebuah list, dan memperlakukan setiap element baru sebagai elemen list secara individual.

```
>>> nama
['kuncoro', 'felix', 'ryan', 'yuan', 'bejo']
>>> nama.extend(["tejo","ujo"])
>>> nama
['kuncoro', 'felix', 'ryan', 'yuan', 'bejo', 'tejo', 'ujo']
```

3. **sort** : metode yang digunakan untuk mengurutkan elemen pada sebuah list dimulai dari nilai terkecil hingga terbesar.

```
>>> nama.sort()
>>> nama
['bejo', 'felix', 'kuncoro', 'ryan', 'tejo', 'ujo', 'yuan']
```

Beberapa metode di atas merupakan metode bertipe void. Metode di atas akan memodifikasi list tersebut dan memiliki nilai balik berupa None. Sehingga jika anda secara tidak sengaja mengetikkan nama = nama.sort(), anda akan mendapatkan hasil None, dan isi elemen pada variabel list nama akan hilang.

Metode lainnya yang sering digunakan pada sebuah list ialah metode untuk menghapus elemen pada list. Ada beberapa cara yang dapat digunakan untuk menghapus elemen pada sebuah list, yaitu:

1. **pop** : Jika indeks elemen sudah diketahui, dan ingin mendapatkan nilai elemen yang dihapus.

```
>>> huruf = ['a','b','c','d']
>>> huruf.pop(3)
'd'
>>> huruf
['a', 'b', 'c']
```

2. **del** : Jika indeks elemen sudah diketahui, dan tidak memerlukan nilai elemen yang dihapus.

```
>>> huruf = ['a','b','c','d']
>>> del huruf[2]
>>> huruf
['a', 'b', 'd']

>>> huruf = ['a','b','c','d']
>>> del huruf[1:3]
```

```
>>> huruf
['a', 'd']
```

3. **remove** : Jika yang diketahui ialah nilai elemen yang akan dihapus.

```
>>> huruf = ['a','b','c','d']
>>> huruf.remove("c")
>>> huruf
['a', 'b', 'd']
```

Penghapusan dengan metode **remove** akan menghasilkan nilai balik None.

List juga memberikan fungsi yang sudah *built-in* yang dapat langsung digunakan secara cepat tanpa harus menuliskan dalam bentuk program perulangan.

```
>>> angka = [10,20,30]
>>> len(angka)
3
>>> max(angka)
30
>>> min(angka)
10
>>> sum(angka)
60
```

1. len() -> untuk mendapatkan banyaknya elemen pada list
2. max() -> untuk mendapatkan nilai maksimum elemen pada list
3. min() -> untuk mendapatkan nilai minimum elemen pada list
4. sum() -> untuk mendapatkan total nilai elemen pada list. Metode ini hanya bisa digunakan pada list dengan tipe elemennya ialah sebuah angka.

Fungsi len, max, min hanya dapat digunakan pada list dengan nilai elemennya yang dapat dibandingkan satu sama lainnya.

9.3.4 Perbedaan List dan String

String merupakan deretan karakter yang mengisi setiap indeks, sedangkan list merupakan deretan nilai yang mengisi setiap indeksnya. Nilai dari setiap elemen list bisa diisi dengan berbagai tipe data, sedangkan string hanya diisi oleh sebuah karakter. Tetapi list dari sebuah karakter bukan berarti sebuah string. Proses pengubahan sebuah string menjadi list karakter dapat dilakukan dengan menggunakan:

```
>>> a = "banana"
>>> b = list(a)
>>> print(b)
['b', 'a', 'n', 'a', 'n', 'a']
```

Pada list, fungsi split dapat digunakan untuk memecah sebuah kalimat menjadi beberapa kata, dengan pemisah default berupa spasi.

```
>>> kalimat = "saya sedang makan"
>>> kata2 = kalimat.split()
>>> print(kata2)
['saya', 'sedang', 'makan']

>>> kalimat = "saya-sedang-makan"
>>> kata2 = kalimat.split('-')
```

```
>>> print(kata2)
['saya', 'sedang', 'makan']
```

Untuk mengembalikan sebuah list yang berisi kata menjadi sebuah string, dapat kita gunakan fungsi join.

```
>>> print(kata2)
['saya', 'sedang', 'makan']
>>> delimiter = ' '
>>> t = delimiter.join(kata2)
>>> type(t)
<class 'str'>
>>> print(t)
saya sedang makan
```

Pada sebuah list, jika kita melakukan pengisisan nilai variabel list a dari variabel list b, maka dua variabel tersebut akan merujuk pada objek yang sama. Hal ini dinamakan aliasing. Sehingga jika terjadi perubahan pada elemen pada list b, maka akan berdampak juga pada elemen list a.

```
>>> a =[1,2,3]
>>> b=a
>>> b is a
True
>>> b[0] = 17
>>> print(a)
[17, 2, 3]
```

Berbeda dengan jika kita membuat dua variabel dan mengisikan nilai pada masing-masing variabel secara terpisah, sehingga pada akhirnya dua variabel ini tidak akan merujuk pada objek yang sama.

```
>>> a =[1,2,3]
>>> b =[1,2,3]
>>> b is a
False
>>> b[0] = 17
>>> print(a)
[1, 2, 3]
>>>
```

9.3.5 List Sebagai Parameter Fungsi

Tipe data list juga dapat digunakan pada parameter sebuah fungsi. Beberapa hal penting yang harus dipahami dalam menggunakan tipe data list baik itu operasi yang berdampak pada pembuatan varabel list baru ataupun memodifikasinya. Sebagai contoh bahwa fungsi **append** akan memodifikasi sebuah list, dan operasi + akan membuat variabel list baru.

```
>>> t1 = [1, 2]
>>> t2 = t1.append(3)
>>> print(t1)
[1, 2, 3]
>>> print(t2)
None
```

```
>>> t3 = t1 + [3]
>>> print(t3)
[1, 2, 3]
>>> t2 is t3
False
```

Perbedaan hasil dan operasi pada sebuah list sangat penting saat membuat sebuah fungsi yang akan melakukan operasi modifikasi sebuah list. Sebagai contoh, fungsi **hapus** ini digunakan untuk menghapus indeks pertama pada sebuah list, maka fungsi **hapus** dapat kita tulis seperti ini:

```
def hapus(inputList):
    return inputList[1:]

huruf = ['a','b','c']
hasil = hapus(huruf)

>>>print(hasil)
['b','c']
```

Dari fungsi **hapus** tersebut, maka nilai pada variabel huruf akan tetap, dan hasil modifikasi diisikan ke variabel **hasil**.

9.4 Kegiatan Praktikum

Kegiatan praktikum akan dilakukan untuk memahami kasus-kasus yang dapat diselesaikan dengan menggunakan variabel dengan tipe **List** :

Kasus 9.1 Buatlah sebuah fungsi yang dapat memberikan nilai balik **true** jika 2 buah list memiliki paling tidak 1 elemen dengan nilai sama.

Pembahasan Soal 1

Fungsi yang dibuat memiliki 2 buah argument list. Untuk dapat mengecek apakah 2 buah list memiliki paling tidak 1 elemen yang sama, proses yang harus dilakukan ialah membandingkan setiap elemen dari kedua list tersebut. Untuk membandingkan elemen tiap list, pengaksesan setiap elemen perlu dilakukan. Karena ada 2 buah list, maka pada program kita akan menggunakan 2 tingkat perulangan. Jika telah ditemukan nilai elemen list yang sama, maka akan keluar dari fungsi serta memberikan nilai balik **true**. Tetapi jika sampai elemen indeks terakhir berlum ditemukan nilai elemen list yang sama maka fungsi akan memberikan nilai balik **false**. Perhatikan kode program berikut:

```
def common_data(list1, list2):
    result = False
    for x in list1:
        for y in list2:
            if x == y:
                result = True
                return result

print(common_data([1,2,3,4,5], [5,6,7,8,9]))
print(common_data([1,2,3,4,5], [6,7,8,9]))
```

Kasus 9.2 Buatlah sebuah fungsi dengan argument bertipe list untuk mencari bilangan terbesar kedua dari sebuah list. Bilangan yang diinputkan bisa juga terjadi duplikasi nilai.

Pembahasan Soal 2

Fungsi yang dibuat membutuhkan 1 buah argument list. Untuk dapat mencari bilangang terbesar kedua, yang dibutuhkan pertama ialah mencari bilangan unik terlebih dahulu dari list bilangan yang ada. Untuk mendapatkan bilangan unik, variabel dengan tipe data **set** dapat digunakan. Tipe data **set** akan menghasilkan nilai unik dari sebuah list. Selanjutnya, proses pengurutan dapat dilakukan. Sehingga sudah urut, akan dengan mudah menentukan index elemen yang menyimpan nilai terbesar kedua dari list yang diberikan.

```
def terbesarkedua(bilangan):
    if (len(bilangan)<2):
        return
    if ((len(bilangan)==2)  and (bilangan[0] == bilangan[1]) ):
        return

    set_item = set()
    unik_item = []
    for x in bilangan:
        if x not in set_item:
            unik_item.append(x)
            set_item.add(x)
    unik_item.sort()

    return  unik_item[1]

print(terbesarkedua([1, 2, -8, -2, 0, -2]))
print(terbesarkedua([1, 1, 0, 0, 2, -2, -2]))
print(terbesarkedua([1, 1, 1, 0, 0, 0, 2, -2, -2]))
```

Kasus 9.3 Buatlah sebuah fungsi untuk mengecek apakah sebuah list A memiliki subset dari list B. Contoh: list A = [4,2,5,2,1,4,5] list B = [2,5] maka hasilnya ialah benar, bahwa list A memiliki subset list B yang bernilai [2,5]

Pembahasan Soal 3

Proses pengecekan yang pertama harus dilakukan ialah apakah subset B kosong/ sama dengan A/ list B lebih panjang dari list A. Selanjutnya proses yang dilakukan ialah pengecekan antara subset list B dengan list A. Hal ini membutuhkan pemahanan pengaksesan indeks pada setiap elemen list.

```
def apakahSublist(A, B):
    sub_set = False
    if B == []:
        sub_set = True
    elif B == A:
        sub_set = True
    elif len(B) > len(A):
        sub_set = False
    else:
        for i in range(len(A)):
            if A[i] == B[0]:
                n = 1
                while (n < len(B)) and (A[i+n] == B[n]):
```

```
                n += 1

            if n == len(B):
                sub_set = True
        return sub_set

a = [2,4,3,5,7]
b = [4,3]
c = [3,7]
print(apakahSublist(a, b))
print(apakahSublist(a, c))
```

9.5 Latihan Mandiri

Latihan 9.1 Buatlah sebuah program untuk mencari 3 nilai terbaik pada sebuah list. Program akan menampilkan 3 nilai terbaik dimulai dari yang paling tinggi. ■

Latihan 9.2 Buatlah sebuah program yang meminta masukan dari user dan menyimpannya dalam bentuk tipe data list dalam bentuk angka. Saat user memasukkan masukan **'done'**, maka program akan menampilkan nilai maksimum dan minimum dari deretan angka yang sudah user masukkan. Gunakan fungsi list max() dan min() pada program ini. ■

Latihan 9.3 Carilah artikel berita pada media massa online. Simpan dalam bentuk file *.txt. Buatlah program untuk membaca file *.txt tersebut dan membaca file baris demi baris kalimat. Untuk setiap kalimat, pecahlah menjadi setiap kata dengan fungsi split(). Simpan kata pada variabel **kata** dengan tipe data **list**. Hasil akhir program ialah variabel **kata** akan menyimpan kata unik pada file berita tersebut. Hasil luaran seperti pada Gambar 9.1 . ■

```
========== RESTART: C:/Users/Kuncoro Saputra/Desktop/test.py ==========
=======ISI BERITA=======
Sementara itu, laporan Bank Sentral AS menunjukkan perekonomian terbesar dunia i
ni akan semakin memburuk ke depan. Termasuk tingkat pengangguran yang diramalkan
 akan terus naik, akibat terhentinya aktivitas ekonomi karena wabah. Dana Monete
r Internasional (IMF) juga sempat meramalkan, pertumbuhan ekonomi global akan me
lambat di tahun 2020. Hal ini terjadi akibat pandemi virus corona yang berlangsu
ng sejak awal tahun 2020, menyebabkan industri mengalami banyak kerugian.


=======Kata Unik pada Berita=======
['Sementara', 'itu,', 'laporan', 'Bank', 'Sentral', 'AS', 'menunjukkan', 'pereko
nomian', 'terbesar', 'dunia', 'ini', 'akan', 'semakin', 'memburuk', 'ke', 'depan
', '', 'Termasuk', 'tingkat', 'pengangguran', 'yang', 'diramalkan', 'terus', 'na
ik,', 'akibat', 'terhentinya', 'aktivitas', 'ekonomi', 'karena', 'wabah', 'Dana'
, 'Moneter', 'Internasional', '(IMF)', 'juga', 'sempat', 'meramalkan,', 'pertumb
uhan', 'global', 'melambat', 'di', 'tahun', '2020', 'Hal', 'terjadi', 'pandemi',
 'virus', 'corona', 'berlangsung', 'sejak', 'awal', '2020,', 'menyebabkan', 'ind
ustri', 'mengalami', 'banyak', 'kerugian']
```

Gambar 9.1: Contoh Program