# PEMROGRAMAN BERORIENTASI OBJEK

Pertemuan 2

# String

- Merupakan karakter atau sekumpulan karakter yang berada di dalam tanda petik.
- Untuk mendeskripsikan suatu string, dapat menggunakan tanda petik tunggal (‘’) atau tanda petik ganda (“”)
- Contoh penulisan string:

```
>>> 'Haloo, ini adalah string'
'Haloo, ini adalah string'
>>> "Haloo, ini adalah string"
'Haloo, ini adalah string'
>>> _
```

# Operasi String

- Terdapat operasi yang dapat dilakukan pada string, yaitu operasi **penjumlahan** dan **penugasan**.
- Contoh operasi string:

```
>>> a="Universitas Mataram"
>>> print(a)
Universitas Mataram
>>> b=a
>>> print(b)
Universitas Mataram
>>> c="Saya adalah mahasiswa "
>>> print(c)
Saya adalah mahasiswa
>>> hasil=c+a
>>> print(hasil)
Saya adalah mahasiswa Universitas Mataram
>>>
```

# Operasi String

- Untuk menampilkan karakter tertentu pada suatu string, dapat dilakukan dengan cara mengakses indeks string yang berada pada tanda [ ].
- Format perintah/: **nama_variabel[indeks]**
- Contoh operasi string:

```
>>> a="Universitas Mataram"
>>> print(a[2])
i
>>> print(a[:2])
Un
>>> print(a[2:])
iversitas Mataram
>>>
```

# Operasi String

- Untuk menambah/menyisipkan suatu string dapat menggunakan operator penjumlahan dan dapat juga dilakukan melalui indeks dari string tersebut.
- Contoh operasi string:

```
>>> str="AKU CINTA INDONESIA"
>>> print(str)
AKU CINTA INDONESIA
>>> str=str[:10]+"TANAH AIR KU"+str[9:]
>>> print(str)
AKU CINTA TANAH AIR KU INDONESIA
>>>
```

# Operasi String

- Untuk menghapus seluruh elemen string dengan menugaskan string kosong pada **str**.
- Untuk menghapus elemen tertentu pada string, dengan menambahkan indeks yang akan dihapus.
- Contoh operasi string:

```
aku cinta tanah air ku Indonesia
>>> str=''
>>> print(str)

>>> str1='aku cinta tanah air ku Indonesia'
>>> str1=str1[:9]+''+str1[22:]
>>> print(str1)
aku cinta Indonesia
```

# Operasi String

- Fungsi **upper()**, digunakan untuk mengubah ukuran huruf menjadi huruf besar/kapital.
- Fungsi **lower()**, digunakan untuk mengubah huruf menjadi huruf kecil.
- Contoh operasi string:

```
>>> kelas='Praktikum pemrograman berorientasi objek'
>>> up=kelas.upper()
>>> lo=kelas.lower()
>>> print(kelas)
Praktikum pemrograman berorientasi objek
>>> print(up)
PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK
>>> print(lo)
praktikum pemrograman berorientasi objek
```

# Operasi String

- Fungsi **len()**, digunakan untuk mengetahui panjang string.

- Contoh operasi string:

```
>>> len(kelas)
40
>>> jumlah=len(kelas)
>>> print('panjang string adalah: ',jumlah)
panjang string adalah:  40
```

# Operasi String

- Kelas string menyediakan beberapa fungsi untuk mengatur rerata tampilan string, yakni: **rjust()**, **ljust()**, dan **center()**.
- Fungsi rjust() untuk mengatur rerata kanan.
- Fungsi ljust() untuk mengatur rerata kiri.
- Fungsi center() untuk mengatur rerata tengah.
- Contoh operasi string:

```
>>> kata="Halooo, Nama Saya Hamzan Wadi"
>>> print kata.center(50)
          Halooo, Nama Saya Hamzan Wadi
>>> print kata.ljust(50)
Halooo, Nama Saya Hamzan Wadi
>>> print kata.rjust(50)
                    Halooo, Nama Saya Hamzan Wadi

>>> _
```

# Operasi String

- Fungsi index() untuk mengetahui posisi karakter dalam string.
- Fungsi replace() untuk mengganti karakter pada suatu string.
- Contoh operasi string:

```
>>> kata="Haloo, Selamat datang"
>>> print kata.index('a')
1
>>> print kata.index('d')
15
>>> print kata.index('Se')
7
>>> _
```

```
>>> print(kelas2)
Praktikum pemrograman berorientasi objek
>>> kelas2=kelas.replace('Praktikum','praktik')
>>> print(kelas2)
praktik pemrograman berorientasi objek
```

# Operasi String

- Untuk menampilkan string terformat menggunakan fungsi **print()** diikuti dengan karakter **%s**. Format:

```
print("%s %s %s"%(srt1, str2, str3))
```

- Contoh operasi string:

```
>>> a="satu"
>>> b="dua"
>>> c="tiga"
>>> print("%s %s %s." %(a,b,c))
satu dua tiga.
>>> print("Saya mempunyai %s mangga"%(c))
Saya mempunyai tiga mangga
>>> print("Andi mempunyai %s jambu, dan saya mempunyai %s mangga" %(c,b))
Andi mempunyai tiga jambu, dan saya mempunyai dua mangga
>>>
```

# Operasi String

- Fungsi input() untuk mengisi masukan dari keyboard, tetapi fungsi ini hanya berlaku untuk string.

- Contoh operasi string:

```
>>> data1=input("String 1 : ")
String 1 : Nama saya
>>> data2=input("String 2 : ")
String 2 : Dwitya Citta Prihatining
>>> hasil=data1+data2
>>> print(hasil)
Nama saya Dwitya Citta Prihatining
>>> _
```

# List

- List adalah struktur data pada python yang mampu menyimpan lebih dari satu data, yang nilainya setiap anggotanya dapat diubah.
- List dibuat seperti variabel tetapi menggunakan tanda *bracket* [ ], dan setiap isinya/anggotanya dipisah menggunakan tanda koma. Setiap anggota list disebut juga elemen list.
- Contoh:

```
#Membuat list kosong
warna=[]
#Membuat list dengan isi 1 elemen
hobi=[“membaca”]
#Membuat list dengan isi banyak elemen
buah=[“jeruk”, “apel”, “manga”]
```

# List

- List dapat diisi dengan tipe data apa saja, yakni: string, integer, float, double, boolean, object, atau mencampur beberapa tipe data.
- Mengakses isi list menggunakan nomor indeks.
- Nomor indeks list selalu dimulai dari nol (0).
- Contoh:

```
>>> buah=["apel","mangga","jeruk","anggur"]
>>> print(buah[2])
jeruk
```

- Mengganti isi list:

```
>>> buah=["apel","mangga","jeruk","anggur"]
>>> buah[2]="jambu"
>>> print(buah)
['apel', 'mangga', 'jambu', 'anggur']
```

# List

- Menambah isi/elemen list dengan menggunakan 2 fungsi/method, yaitu:

  **append()**, untuk menambahkan isi dari belakang

  **insert()**, untuk menambahkan isi dari indeks tertentu
- Fungsi **extend()**, untuk menggabungkan suatu list dengan list atau data dari tipe lain.
- Menghapus isi list, dengan fungsi:

  **del**, untuk menghapus elemen list yang terletak pada indeks tertentu

  **remove()**, untuk menghapus elemen dengan nilai tertentu yang terdapat didalam list

  **pop()**, untuk menghapus elemen terakhir pada list

# List

- Untuk memotong isi list atau menampilkan elemen list tertentu dapat menggunakan indeks dan operator slice (:)
- Untuk menggabungkan elemen list dapat menggunakan operator (+) dan (*)
- Untuk mengurutkan elemen list menggunakan fungsi **sorted()** dan **sort()**
- Untuk membalik urutan elemen list menggunakan **reverse()**
- Mencari nilai maksimum dan minimum pada elemen list menggunakan fungsi **max()** dan **min()**

# Dictionary

- Berfungsi untuk membuat data berkelompok yang nilai setiap elemen/anggotanya dapat diubah, tetapi indeksnya dapat ditentukan sendiri.
- Dibuat dengan menggunakan tanda kurung kurawal { }, dan setiap anggotanya berupa pasangan kunci-nilai (key-value). Antara kunci da nilai dipisahkan menggunakan tanda titik dua atau colon :
- Contoh:

```
>>> d={1:100, 2:200, 3:300, 4:400, 5:500}
>>> d
{1: 100, 2: 200, 3: 300, 4: 400, 5: 500}
```

```
>>> d[1]
100
>>> d[2]
200
>>> d[3]
300
>>> d[6]
Traceback (most recent call last):
  File "<pyshell#23>", line 1, in <module>
    d[6]
KeyError: 6
```

# Dictionary

- Untuk pengaksesan sama seperti list, menggunakan bracket [ ]
- Contoh:

```
>>> d[1]
100
>>> d[2]
200
>>> d[3]
300
>>> d[6]
Traceback (most recent call last):
  File "<pyshell#23>", line 1, in <module>
    d[6]
KeyError: 6
```

- Cara lain menggunakan fungsi **get()**

```
>>> d.get(4)
400
>>> d.get(5)
500
```

- Untuk melihat nilai dari suatu dictionary, menggunakan **values()**

```
>>> d.values()
dict_values([100, 200, 300, 400, 500])
```

# Dictionary

- Untuk menghapus satu elemen dari dictionary, menggunakan fungsi **del**
- Untuk menghapus semua elemen dictionary menggunakan fungsi **clear()**
- Untuk membuat sub-dictionary dari dictionary yang sudah ada sebelumnya, menggunakan fungsi **extractdict()**
- Untuk menyalin dictionary ke dictionary lain, menggunakan fungsi **copy()**

# Tuple

- Berfungsi untuk membuat data berkelompok yang nilai setiap elemen/anggotanya bersifat tidak dapat diubah, tidak dapat ditambah, maupun tidak dapat dikurangi.
- Dibuat dengan menggunakan tanda kurung ( ) atau tidak.

```
>>> t=(100,200,300,400)
>>> t
(100, 200, 300, 400)
>>> nilai=10,20,30,40
>>> nilai
(10, 20, 30, 40)
```

- Cara mengakses tuple menggunakan indeks yang dimulai dari 0.

```
>>> t[0]
100
>>> nilai[2]
30
```

# Tuple

- Untuk mengetahui indeks dari nilai elemen tertentu, menggunakan fungsi **index()**
- Untuk menghitung banyaknya elemen yang yang sama di dalam tuple, menggunakan fungsi **count()**

# Set (Himpunan)

- Berfungsi untuk membuat data berkelompok yang nilai setiap elemen/anggotanya bersifat unik, tidak terurut, dan tidak memiliki duplikasi data.
- Dibuat dengan menggunakan tanda kurung ( ).
- Elemen set tidak dapat diindeks dan tidak mendukung operator slice (:).
- Set digunakan untuk melakukan operasi himpunan matematika seperti gabungan, irisan, selisih, dsb.

```
>>> a=[10,20,30,20,40]
>>> s=set(a)
>>> s
{40, 10, 20, 30}
```

# Set (Himpunan)

- Untuk menambah elemen ke set, menggunakan fungsi **add()** atau **update()**
- Untuk menghapus elemen set, menggunakan fungsi **discard()** dan **clear()**
- Untuk menyalin set dapat menggunakan fungsi **copy()**
- Untuk mencari irisan menggunakan fungsi **intersection()**
- Untuk mencari selisih dari dua set menggunakan fungsi **difference()**
- Untuk menggabungkan dua set menggunakan fungsi **union()**
- Untuk membuat elemen set tidak dapat diubah menggunakan fungsi **frozenset()**

# Tugas

- Mencoba fungsi untuk string, list, dictionary, tuple, dan set.
- Upload ke github
- Deadline: 19 Februari 2022, 23.59

# Terima Kasih