

TUGAS PYTHON

MODUL I - III

Nama : Nessa Kartika

NIM : 211001039

Kelas : 4 D

MODUL I

1. Modul 1

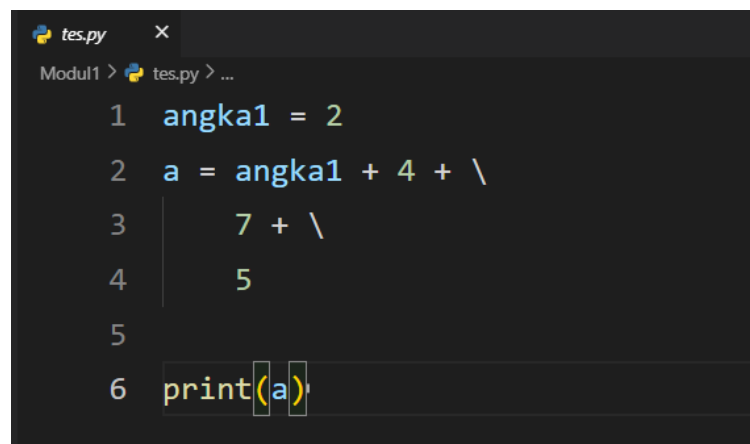
Berikut contoh yang benar untuk memunculkan tampilan output yang ingin kita inginkan :



```
Modul1.py x
Modul1 > Modul1.py
1 print("BARIS")
2 print("INDENTASI")
```

2. Pernyataan Multi Baris

- Di Python, akhir pernyataan adalah karakter baris baru. kita dapat buat pernyataan multi-baris dengan satu karakter Garis miring terbalik (\).



```
tes.py x
Modul1 > tes.py > ...
1 angka1 = 2
2 a = angka1 + 4 + \
3     7 + \
4     5
5
6 print(a)
```

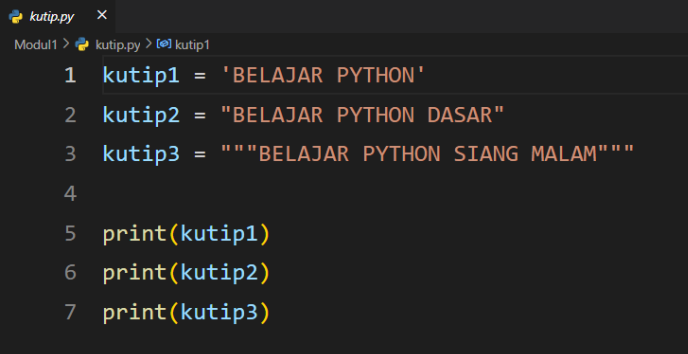
- Pernyataan antara tanda kurung [], { } dan () tidak memerlukan \.



```
Modul1 > namahari.py > ...  
1 namaHari = ['Senin', 'Selasa', 'Rabu', 'Kamis']  
2 print(namaHari)
```

3. Tanda Kutip

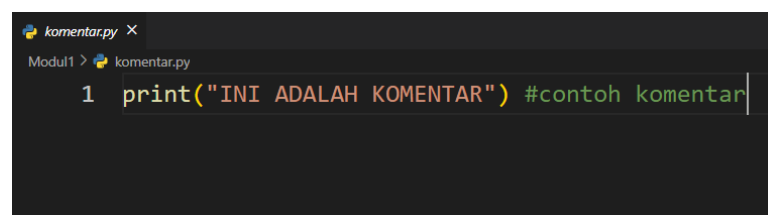
Python menggunakan tanda kutip ganda tunggal ('), ganda (""), dan rangkap tiga (''' atau ''') ke string, selama string dimulai dengan tanda kutip ganda yang sama dan akhir string. Tanda kutip tiga digunakan untuk string multi-baris.



```
Modul1 > kutip.py > kutip1  
1 kutip1 = 'BELAJAR PYTHON'  
2 kutip2 = "BELAJAR PYTHON DASAR"  
3 kutip3 = """BELAJAR PYTHON SIANG MALAM"""  
4  
5 print(kutip1)  
6 print(kutip2)  
7 print(kutip3)
```

4. Komentar

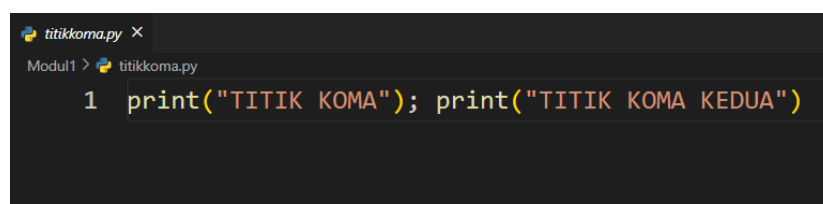
Untuk memberikan komentar python, gunakan tanda (#) untuk menandai. Apapun yang ditandai oleh tanda (#) tidak akan diproses oleh interpreter python.



```
Modul1 > komentar.py  
1 print("INI ADALAH KOMENTAR") #contoh komentar
```

5. Dua Pernyataan Dalam Satu Baris

Titik koma dapat digunakan ketika ada 2 pernyataan dalam satu baris kode.



```
Modul1 > titikkoma.py  
1 print("TITIK KOMA"); print("TITIK KOMA KEDUA")
```

6. Operator Aritmatika

Operator Aritmatika digunakan untuk melaksanakan prosedur Matematika seperti penjumlahan, pengurangan, perkalian, pembagian dan sebagainya.

```
aritmataka.py x
Modul1 > aritmataka.py > [e] a
1 a = 10
2 b = 3
3
4 c = a / b
5 d = a ** b
6 e = a // b
7 print("Hasil : ",c)
8 print("Pangkat : ", d)
9 print("Hasil : ", e)
```

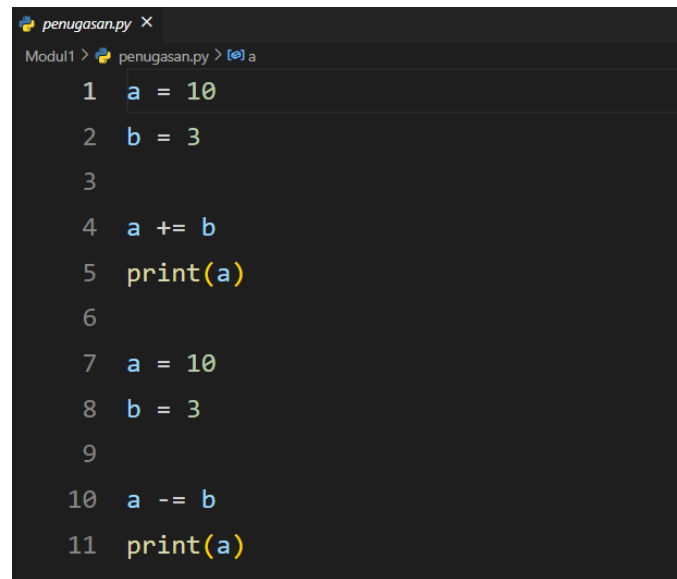
7. Operator Perbandingan

Operator perbandingan digunakan untuk membandingkan 2 buah melakukan Hasil perbandingan benar atau salah tergantung pada keadaan.

```
perbandingan.py x
Modul1 > perbandingan.py > [e] a
1 a = 10
2 b = 3
3
4 c = a < b
5 print(c)
6
7 c = a > b
8 print(c)
9
10 c = a == b
11 print(c)
12
13 c = a != b
14 print(c)
```

8. Operator Penugasan

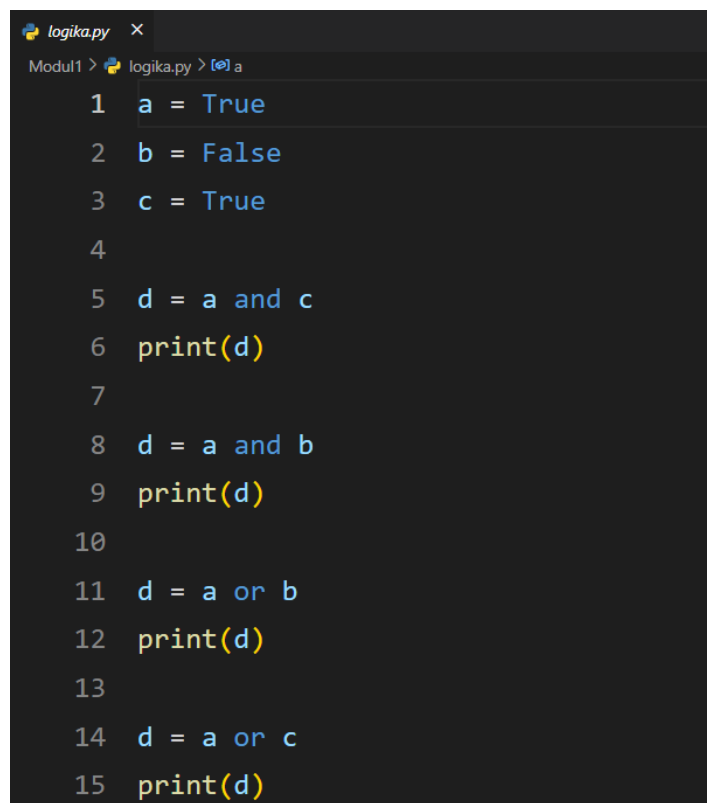
Operator penugasan adalah operator yang digunakan untuk memberi nilai ke variabel.



```
penugasan.py X
Modul1 > penugasan.py > [a] a
1 a = 10
2 b = 3
3
4 a += b
5 print(a)
6
7 a = 10
8 b = 3
9
10 a -= b
11 print(a)
```

9. Operator Logika

Operator logika adalah operator yang digunakan untuk melakukan operasi logika.



```
logika.py X
Modul1 > logika.py > [a] a
1 a = True
2 b = False
3 c = True
4
5 d = a and c
6 print(d)
7
8 d = a and b
9 print(d)
10
11 d = a or b
12 print(d)
13
14 d = a or c
15 print(d)
```

10. Operator Bitwise

Operator bitwise adalah operator yang melakukan operasi bit terhadap operand.

```
bitwise.py x
Modul1 > bitwise.py > [a] a
1 a = 1
2 b = 2
3
4 c = a | b
5 print(c)
6
7 c = a & b
8 print(c)
```

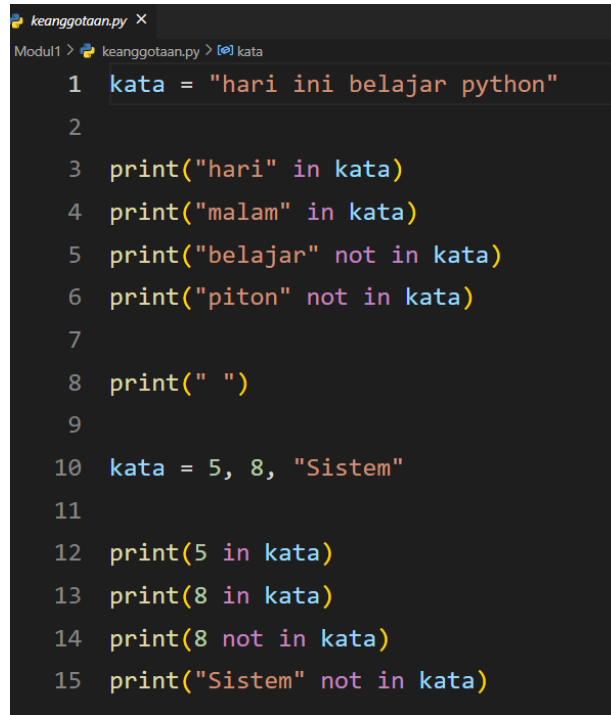
11. Operator Identitas

Operator identitas adalah operator yang menguji apakah dua nilai (atau variabel) berada di lokasi memori yang sama.

```
identitas.py x
Modul1 > identitas.py > [a] a
1 a = 1
2 b = 2
3
4 print(1 is a)
5 print(2 is b)
6 print(3 is a)
7 print(1 is not a)
8 print(2 is not b)
9
10 print(" ")
11
12 print(type(a) is int)
13 print(type(b) is float)
```

12. Operator Keanggotaan

Operator keanggotaan adalah operator yang digunakan untuk memeriksa apakah suatu Nilai atau variabel adalah anggota atau ditemukan dalam data (string, list, tuple, set, dan dictionary).



```
keanggotaan.py X
Modul1 > keanggotaan.py > kata
1 kata = "hari ini belajar python"
2
3 print("hari" in kata)
4 print("malam" in kata)
5 print("belajar" not in kata)
6 print("piton" not in kata)
7
8 print(" ")
9
10 kata = 5, 8, "Sistem"
11
12 print(5 in kata)
13 print(8 in kata)
14 print(8 not in kata)
15 print("Sistem" not in kata)
```

MODUL II

1. Output

Untuk melakukan operasi output adalah print(). Atau untuk menampilkan data ke perangkat keluaran standar (layar).

- **sep** adalah pemisah(separator) yang berfungsi sebagai tanda pemisah antar objek yang dicetak. Defaultnya adalah tanda spasi.
- **end** adalah karakter yang dicetak di akhir baris. Defaultnya adalah tanda newline (baris baru).
- **file** adalah nama file kemana objek akan dicetak. Defaultnya adalah ke sys.stdout (layar).
- **flush** adalah opsi apakah keluarannya diflush atau tidak. Digunakan untuk menulis data yang belum tertulis dari buffer ke perangkat output (seperti layar).

```
output1.py X
Modul2 > output1.py
1 print(1, 3, 5, 7)
2 #output: 1 3 5 7
3
4 print(1, 2, 3, 4, sep='*')
5 #output 1*2*3*4
6
7 print(1,2,3,4, sep='#', end="&")
8 #output: 1#2#3#4&
```

2. Input

- Input adalah masukan yang kita berikan ke program. Program akan memprosesnya dan menampilkan hasil outputnya.

```
input1.py X
Modul2 > input1.py > ...
1 #Contoh Kode Input
2 a = int(input("Masukkan Nilai A : "))
3 b = int(input("Masukkan Nilai B : "))
4
5 print(a,b)
6
7 #Contoh Kode Input Integer Tanpa Fungsi int()
8 a = int(input("Masukkan Nilai A : "))
9 b = int(input("Masukkan Nilai B : "))
10
11 c = a + b
12 print(c)
13
14 #Fungsi int() Cara Kedua
15 a = int(input("Masukkan Nilai A : "))
16 b = int(input("Masukkan Nilai B : "))
17
18 c = int(a) + int(b)
19 print(c)
```

- **Float()** untuk menginput bilangan pecahan.

```
input_float.py X
Modul2 > input_float.py > [e] a
1 a = input("Masukkan Nilai A : ")
2 b = input("Masukkan Nilai B : ")
3
4 c = float(a) / float(b)
5 print(c)
```

- **abs()** menggunakan fungsi `abs()` kita bisa hilangkan tipe data yang ada minusnya .

```
input_abs.py x
Modul2 > input_abs.py > ...
1 #Fungsi abs() Statis
2 a = -5
3 c = abs(a)
4 print(c)
5
6
7 #Fungsi abs() Dinamis
8 a = int(input("Masukkan Nilai A : "))
9 c = abs(a)
10 print(c)
```

- **pow()** adalah fungsi untuk menghitung pangkat.

```
input_pow.py x
Modul2 > input_pow.py > ...
1 #Fungsi pow() Statis
2 c = pow(2,3)
3 print(c)
4
5
6 #Fungsi pow() Dinamis
7 a = int(input("Masukkan Nilai :"))
8 b = int(input("Masukkan Nilai Pangkat :"))
9
10 c = pow(a,b)
11 print(c)
```

- **sqrt()** untuk mencari akar kuadrat dari suatu pangkat.

```
input_sqrt.py x
Modul2 > input_sqrt.py > ...
1 #Fungsi sqrt() Statis
2 import math
3 c = math.sqrt(36)
4 print(c)
5
6
7 #Fungsi sqrt() Dinamis
8 import math
9 a = input("Masukkan Nilai : ")
10
11 c = math.sqrt(int(a))
12 print(c)
```


- Kemudian ada fungsi lain seperti **max()**, untuk menampilkan nilai paling akhir. **min()**, menampilkan nilai paling awal. **round()** atau **ceil()** untuk pembulatan keatas. Dan **floor()** untuk pembulatan kebawah.

```
input_fungsi_lain.py x
Modul2 > input_fungsi_lain.py
1 import math
2
3 print(max(2,1,5)) #output-nya 5
4
5 print(min(2,1,5)) #output-nya 1
6
7 print(round(5.8)) #output-nya 6
8
9 print(math.floor(5.8)) #output-nya 5
10
11 print(math.ceil(5.8)) #output-nya 6
```

3. Operasi String

- **Len** untuk mengembalikan panjang (jumlah anggota) dari suatu objek.

```
string_len.py x
Modul2 > string_len.py > string1
1 string1 = "Hello World"
2
3 print(len(string1))
```

- **index()**. Mencari posisi suatu nilai.

```
string_index.py x
Modul2 > string_index.py > kata
1 kata = "Hello World"
2 print(kata.index("o"))
```

- **count()**. Menghitung kemunculan nilai tertentu.
- **upper()**. Mengubah string menjadi huruf kapital.
- **lower()**. Mengubah string menjadi huruf kecil.
- **split()**. Memisah string menjadi item

```
string_lain.py x
Modul2 > string_lain.py > kata
1 kata = "Hello World"
2 print(kata.count("o"))
3 print(kata.upper())
4 print(kata.lower())
5
6 kata_baru = kata.split(" ")
7 print(kata_baru)
```

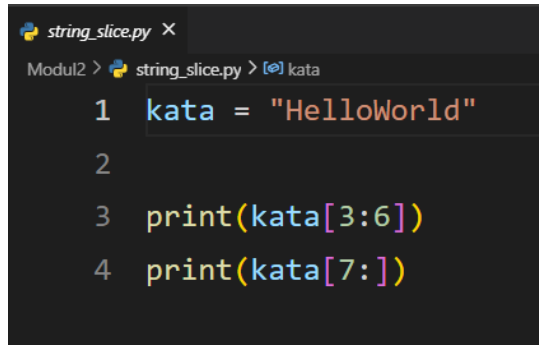
4. Range Slice.

Range Slice menampilkan range karakter dari a mendekati b (limit b), yang diformulasikan dengan:

nama_variabel[a:b]

a = index karakter yang mulai dicetak.

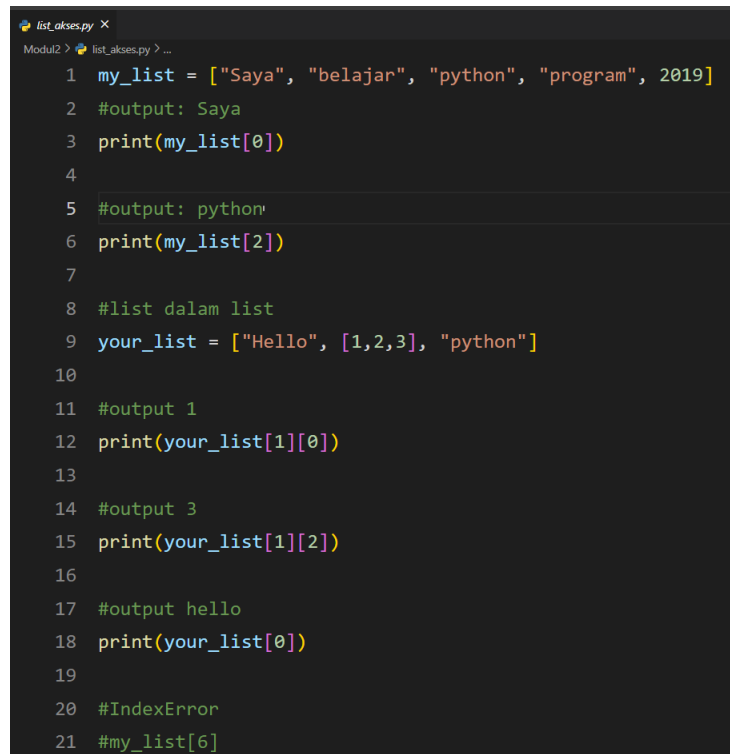
b = akhir karakter yang dicetak, b tidak dicetak atau kosong untuk dicetak ke karakter terakhir.



```
string_slice.py x
Modul2 > string_slice.py > kata
1 kata = "HelloWorld"
2
3 print(kata[3:6])
4 print(kata[7:])
```

5. Mengakses Anggota List

Mengakses anggota list dengan menggunakan indeksnya dengan format **namalist[indeks]**.



```
list_akses.py x
Modul2 > list_akses.py > ...
1 my_list = ["Saya", "belajar", "python", "program", 2019]
2 #output: Saya
3 print(my_list[0])
4
5 #output: python
6 print(my_list[2])
7
8 #list dalam list
9 your_list = ["Hello", [1,2,3], "python"]
10
11 #output 1
12 print(your_list[1][0])
13
14 #output 3
15 print(your_list[1][2])
16
17 #output hello
18 print(your_list[0])
19
20 #IndexError
21 #my_list[6]
```

6. List Dengan Indeks Negatif

Python mendukung indeks negatif, yaitu. H. urutan dimulai dengan anggota terakhir. Indeks. Anggota terakhir adalah -1, lalu -2, dst.

```
list_negatif.py X
Modul2 > list_negatif.py > my_list
1 my_list = ['p','y','t','h','o','n']
2
3 # output: n
4 print(my_list[-1])
5
6 #output: h
7 print(my_list[-3])
```

7. Memotong (Slicing) List

Bisa mengakses anggota list dari range tertentu dengan menggunakan operator slicing titik dua (:).

```
list_memotong.py X
Modul2 > list_memotong.py > ...
1 #List Dengan Indeks Negatif
2 my_list = ['p', 'y', 't', 'h', 'o', 'n']
3
4 #output: n
5 print(my_list[-1])
6
7 #output: h
8 print(my_list[-3])
9
10 #Memotong List
11 my_list = ['p', 'y', 't', 'h', 'o', 'n', 's', 'a', 'y', 'a']
12
13 #anggota list dari 3 s/d 5 (dari h s/d n)
14 print(my_list[3:6])
15
16 #anggota list dari 4 s/d yang terakhir
17 print(my_list[4:])
18
19 #anggota list dari 0 s/d 4
20 print(my_list[:5])
```

8. Mengubah Anggota List

List berbeda dengan string dan tuple yang bersifat immutable. Sedangkan List adalah tipe data yang bersifat mutable.

```
list_memotong.py  list_ubah_anggota.py X
Modul2 > list_ubah_anggota.py > ...
1  #misal ada nilai yang salah
2  ganjil = [1,3,4,7,9]
3  print("Item Awal : ", ganjil)
4
5  #ubah item ke 3 (indeks ke 2)
6  ganjil[2] = 5
7  print(ganjil)
8
9  #mengubah sekali banyak
10 ganjil[2:5] = [11,13,15]
11 print(ganjil)
```

9. Mengurutkan Anggota List

Untuk mengurutkan anggota list bisa menggunakan metode **sort()**.

```
list_memotong.py  list_urut_anggota.py X
Modul2 > list_urut_anggota.py > alfabet
1  alfabet = ['a','b','c','d','e','f','g','h','i','j']
2  alfabet.sort()
3  print(alfabet)
4  #output ['a','b','c','d','e','f','g','h','i','j']
```

10. Membalik Urutan List

Untuk membalik dengan urutan sebaliknya bisa dengan menggunakan argumen **reverse=True**.

```
list_memotong.py  list_balik_urutan.py X
Modul2 > list_balik_urutan.py > alfabet
1  alfabet = ['a','c','d','e','b']
2  alfabet.reverse()
3  print(alfabet)
4  #output ['b','e','d','c','a']
```

11. Menyisipkan Anggota List

Untuk menghapus anggota list bisa menggunakan metode `remove()`, `pop()`, atau kata kunci `del`. Fungsi `pop()` selain menghapus anggota list, juga mengembalikan nilai indeks anggota tersebut.

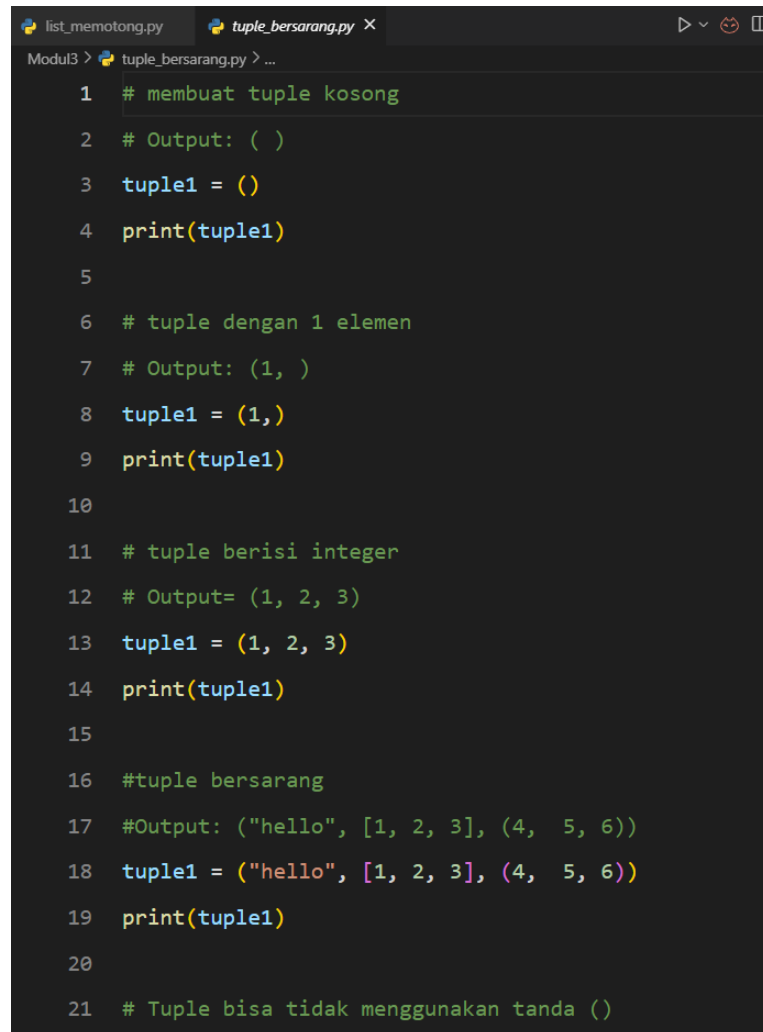
```
list_memotong.py  list_hapus_anggota.py X
Modul2 > list_hapus_anggota.py > my_list

1 my_list = ['p', 'y', 't', 'h', 'o', 'n', 's', 'a', 'y', 'a']
2 my_list.remove('p')
3 #output ['y', 'y', 't', 'h', 'o', 'n', 's', 'a', 'y', 'a']
4 print(my_list)
5
6 my_list = ['p', 'y', 't', 'h', 'o', 'n', 's', 'a', 'y', 'a']
7 my_list.remove('n')
8 #remove hanya menghapus elemen pertama yang dijumpai
9 #output: ['p', 'y', 't', 'h', 'o', 'n', 's', 'a', 'y', 'a']
10 print(my_list)
11
12 #output'y'
13 print(my_list.pop(1))
14
15
16 del my_list[2]
17 print(my_list)
18
19 my_list.clear()
20 #output []
21 print(my_list)
```

MODUL III

1. Membuat Tuple

Tuple dibuat dengan cara menaruh setiap anggota **dalam tanda kurung ()**. Dan masing-masing anggota dipisahkan dengan **tanda koma**. Boleh tidak menggunakan **tanda kurung()** tetapi sebaiknya tetap menggunakan agar lebih mudah pembacaan kode.

A screenshot of a Python IDE with two tabs: 'list_memotong.py' and 'tuple_bersarang.py'. The 'tuple_bersarang.py' tab is active, showing a script with 21 lines of code. The code demonstrates how to create tuples in Python, including empty tuples, single-element tuples, tuples with integers, and tuples containing mixed data types like strings, lists, and other tuples. Each example is followed by a print statement to show the output.

```
1 # membuat tuple kosong
2 # Output: ( )
3 tuple1 = ()
4 print(tuple1)
5
6 # tuple dengan 1 elemen
7 # Output: (1, )
8 tuple1 = (1,)
9 print(tuple1)
10
11 # tuple berisi integer
12 # Output= (1, 2, 3)
13 tuple1 = (1, 2, 3)
14 print(tuple1)
15
16 #tuple bersarang
17 #Output: ("hello", [1, 2, 3], (4, 5, 6))
18 tuple1 = ("hello", [1, 2, 3], (4, 5, 6))
19 print(tuple1)
20
21 # Tuple bisa tidak menggunakan tanda ()
```

2. Mengakses anggota tuple

Untuk mengakses anggota tuple lewat indeksnya, gunakan format **namatuple(indeks)**.

```
list_memotong.py  tuple_akses.py X
Modul3 > tuple_akses.py > tuple1
1 tuple1 = ('p','y','t','h','o','n')
2 #Output: 'p'
3 print(tuple1[0])
4
5 # Output: 'y'
6 print(tuple1[1])
7
8 # Output: 'n'
9 print(tuple1[-1])
10
11 # Output: 'o'
12 print(tuple1[-2])
13
14 # IndexError
15 #Print(tuple1[6])
```

Kita juga bisa mengakses satu range anggota tuple dengan menggunakan operator titik dua (:).

```
list_memotong.py  tuple_akses_range.py X
Modul3 > tuple_akses_range.py > tuple1
1 tuple1 = ('p','r','o','g','r','a','m','m','i','n','g')
2 # akses dari index 0 s/d 2
3
4 #Output: ('p','r','o')
5 print(tuple1[:3])
6
7 #Akses dari index 2 s/d 5
8 # output: ('o','g','r','a')
9 print(tuple1[2:6])
10
11 #Akses dari index 3 sampai akhir
12 #Output: ('g','r','a','m','m','i','n','g')
13 print(tuple1[3:])
```

3. Mengubah Anggota Tuple

Setelah tuple dibuat, anggota tuple tidak dapat diubah atau dihapus. Tetapi jika anggota tuple-nya adalah tuple bersarang dengan anggota list, maka item dalam daftar dapat dimodifikasi.

```
list_memotong.py  tuple_ubah_anggota.py X
Modul3 > tuple_ubah_anggota.py > tuple1
1 tuple1 = (2, 3, 4, [5, 6])
2 # Kita tidak bisa mengubah anggota tuple
3 # bila kita hilangkan tanda komentar # pada baris ke 6
4 # akan muncul error: # TypeError: 'tuple' object does not support item assignment
5
6 # tuple1[1] = 8
7
8 # tapi list di dalam tuple bisa diubah
9 # output: (2, 3, 4, [7, 6])
10 tuple1[3][0] = 7
11 print(tuple1)
12
13 # tuple bisa diganti secara keseluruhan dengan penugasa kembali
14 # output: ('p','y','t','h','o','n')
15 tuple1 = ('p','y','t','h','o','n')
16 print(tuple1)
17
18 # anggota tuple juga tidak bisa dihapus menggunakan del
19 # perintah berikut akan menghasilkan error
20 # kalau anda menghilangkan tanda komentar #
```

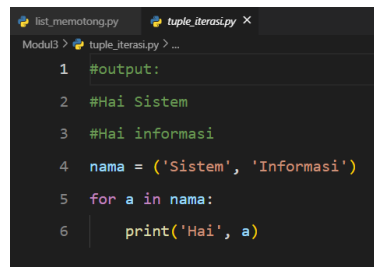
4. Menguji Keanggotaan Tuple

Gunakan operator **in** atau **not in** untuk menguji apakah sebuah objek adalah anggota dari tuple atau tidak.

```
list_memotong.py  tuple_uji_anggota.py X
Modul3 > tuple_uji_anggota.py > tuple1
1 tuple1 = (1, 2, 3, 'a','b','c')
2
3 # menggunakan in
4 # output: True
5 print(3 in tuple1)
6
7 # output: False
8 print('2' in tuple1)
9
10 # output false
11 print('e' in tuple1)
12
13 # menggunakan not in
14 # output True
15 print('k' not in tuple1)
```


5. Iterasi pada tuple

Gunakan **for** untuk melakukan iterasi pada tiap anggota yang terdapat pada tuple.



```
list_memotong.py  tuple_iterasi.py X
Modul3 > tuple_iterasi.py > ...
1 #output:
2 #Hai Sistem
3 #Hai informasi
4 nama = ('Sistem', 'Informasi')
5 for a in nama:
6     print('Hai', a)
```

6. Metode dan Fungsi Bawaan Tuple

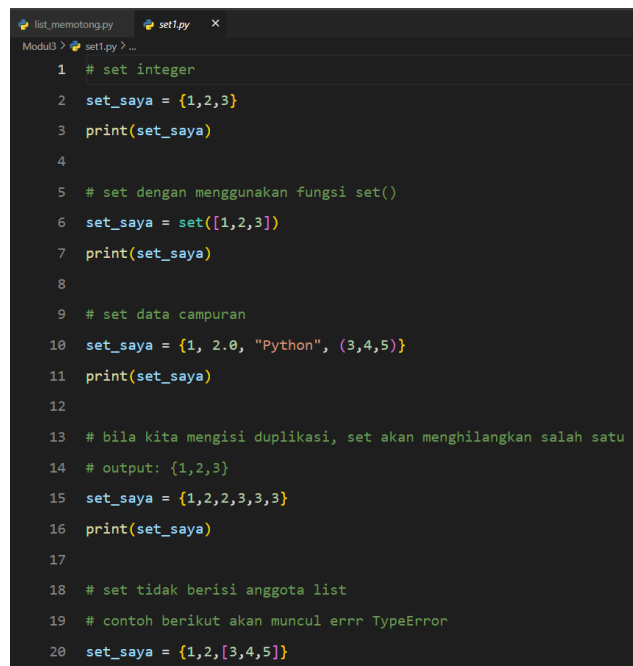
Ada dua metode pada tuple yaitu **metode count()** dan **metode index()**.



```
list_memotong.py  tuple_fungsi.py X
Modul3 > tuple_fungsi.py > tuple1
1 tuple1 = ('p','y','t','h','o','n','s','a','y','a')
2 # count
3 # output: 2
4 print(tuple1.count('a'))
5
6 # index
7 # Output 4
8 print(tuple1.index('n'))
```

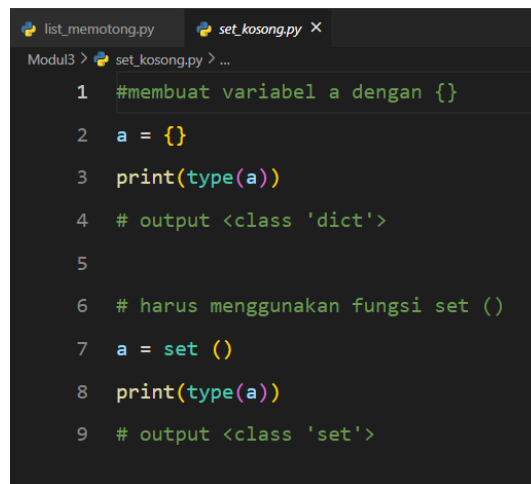
7. Membuat set

Set dibuat dengan meletakkan anggota - anggotanya di dalam tanda kurung kurawal {}, dipisahkan menggunakan tanda koma.



```
list_memotong.py  set1.py X
Modul3 > set1.py > ...
1 # set integer
2 set_saya = {1,2,3}
3 print(set_saya)
4
5 # set dengan menggunakan fungsi set()
6 set_saya = set([1,2,3])
7 print(set_saya)
8
9 # set data campuran
10 set_saya = {1, 2.0, "Python", (3,4,5)}
11 print(set_saya)
12
13 # bila kita mengisi duplikasi, set akan menghilangkan salah satu
14 # output: {1,2,3}
15 set_saya = {1,2,2,3,3,3}
16 print(set_saya)
17
18 # set tidak berisi anggota list
19 # contoh berikut akan muncul error TypeError
20 set_saya = {1,2,[3,4,5]}
```

Set kosong:

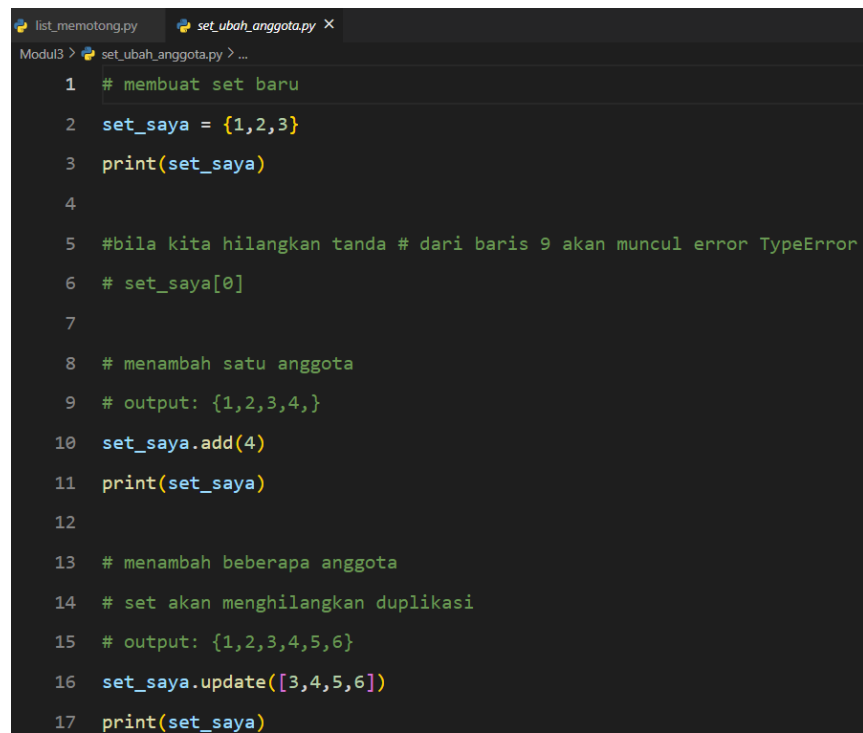


```
list_memotong.py  set_kosong.py X
Modul3 > set_kosong.py > ...

1  #membuat variabel a dengan {}
2  a = {}
3  print(type(a))
4  # output <class 'dict'>
5
6  # harus menggunakan fungsi set ()
7  a = set ()
8  print(type(a))
9  # output <class 'set'>
```

8. Mengubah Anggota set

Untuk menambah satu anggota ke dalam set, kita bisa menggunakan fungsi `add()`, dan untuk menambah beberapa anggota sekaligus gunakan **`update()`**.



```
list_memotong.py  set_ubah_anggota.py X
Modul3 > set_ubah_anggota.py > ...

1  # membuat set baru
2  set_saya = {1,2,3}
3  print(set_saya)
4
5  #bila kita hilangkan tanda # dari baris 9 akan muncul error TypeError
6  # set_saya[0]
7
8  # menambah satu anggota
9  # output: {1,2,3,4,}
10 set_saya.add(4)
11 print(set_saya)
12
13 # menambah beberapa anggota
14 # set akan menghilangkan duplikasi
15 # output: {1,2,3,4,5,6}
16 set_saya.update([3,4,5,6])
17 print(set_saya)
```

9. Menghapus anggota set

Untuk menghapus anggota set gunakan fungsi **discard()** dan **remove()**.

```
list_memotong.py  set_hapus_anggota.py X
Modul3 > set_hapus_anggota.py > ...
1  # membuat set baru
2  set_saya = {1,2,3,4,5}
3  print(set_saya)
4
5  # menghaus 4 dengan discard
6  # output : {1,2,3,5}
7  set_saya.discard(4)
8  print(set_saya)
9
10 # menghapus 5 dengan remove
11 # output : {1,2,3}
12 set_saya.remove(5)
13 print(set_saya)
14
15 # print yang mau dihapus tidak ada dalam set
16 # discard tidak akan muncul error
17 # output: {1,2,3}
18 set_saya.discard(6)
```

Untuk menghapus seluruh anggota gunakan fungsi **clear()**.

```
list_memotong.py  set_hapus_anggota2.py X
Modul3 > set_hapus_anggota2.py > ...
1  # membuat set baru
2  # output: set berisi anggota yang unik
3  set_saya = set("HelloPython")
4  print(set_saya)
5
6  # pop anggota
7  # output: anggota acak
8  print(set_saya.pop())
9  print(set_saya)
10
11 # mengosongkan set
12 # output: set()
13 set_saya.clear
14 print(set_saya)
```

10. Operasi Gabungan (Union)

Set union menggunakan **palang** (**|**) atau bisa juga menggunakan fungsi **union()**.

```
list_memotong.py  set_union.py X
Modul3 > set_union.py > ...
1  # Membuat set A dan B
2  A = {1, 2, 3, 4, 5}
3  B = {4, 5, 6, 7, 8}
4
5  # Gabungan menggunakan operator
6  # output: {1, 2, 3, 4, 5, 6, 7, 8}
7  print(A | B)
8
9  # menggunakan fungsi union()
10 # output: {1, 2, 3, 4, 5, 6, 7, 8}
11 A.union(B)
12
13 # output: {1, 2, 3, 4, 5, 6, 7, 8}
14 B.union(A)
```

11. Operasi Irisan (Intersection)

Irisan dilakukan dengan menggunakan operator **jangkar** (**&**). Atau bisa juga menggunakan fungsi **intersection()**.

```
list_memotong.py  set_intersection.py X
Modul3 > set_intersection.py > ...
1  # Membuat set A dan B
2  A = {1, 2, 3, 4, 5}
3  B = {4, 5, 6, 7, 8}
4
5  # Irisan menggunakan operator &
6  # output: {4,5}
7  print(A & B)
8  # Menggunakan fungsi intersection()
9  # output: {4,5}
10 A.intersection(B)
11
12 #output: {4,5}
13 B.intersection(A)
```

12. Operasi Selisih (Difference)

```
list_memotong.py X set_difference.py X
Modul3 > set_difference.py > ...
1 # membuat A dan B
2 A = {1, 2, 3, 4, 5}
3 B = {4, 5, 6, 7, 8}
4
5 # Menggunakan operator - pada A
6 # Output: {1, 2, 3}
7 print(A - B)
8
9 #Output: {1, 2, 3}
10 A.difference(B)
11
12 #Menggunakan operator - pada B
13 # output: {8, 6, 7}
14 print(B - A)
15
16 # Output: {8,6,7}
17 B.difference(A)
```

13. Operasi Komplemen (Symmetric Difference)

Komplemen dilakukan dengan menggunakan **operator ^**. Atau menggunakan fungsi `symmetric_difference()`.

```
list_memotong.py X set_symmetric_difference.py X
Modul3 > set_symmetric_difference.py > ...
1 # membuat A dan B
2 A = {1, 2, 3, 4, 5}
3 B = {4, 5, 6, 7, 8}
4
5 # menggunakan operator ^ pada A
6 # output: {1, 2, 3, 6, 7, 8}
7 print(A ^ B)
8
9 # output: {1, 2, 3, 6, 7, 8}
10 A.symmetric_difference(B)
11
12 # Menggunakan operator ^ pada B
13 # output: {1, 2, 3, 6, 7, 8}
14 print(B ^ A)
15
16 # output: {1, 2, 3, 6, 7, 8}
17 B.symmetric_difference(A)
```

14. Membuat Dictionary

Untuk membuat Dictionary tempatkan anggotanya di dalam **kurung kurawal**{ } dan dipisahkan oleh **tanda koma**.

```
list_memotong.py dictionary1.py X
Modul3 > dictionary1.py > ...
1 # Membuat dictionary kosong
2 dict1 = {}
3 print(dict1)
4
5 # Dictionary dengan kunci integer
6 dict1 = {1: 'sepatu', 2: 'tas'}
7 print(dict1)
8
9 #dictionary dengan kunci campuran
10 dict1 = {'warna': 'merah', 1: [2,3,5]}
11 print(dict1)
12
13 # membuat dictionary menggunakan fungsi dict()
14 dict1 = dict([('1','sepatu'), ('2','bola')])
15 print(dict1)
16
17 dict1 = dict(m=8, n=9, o=10)
18 print(dict1)
```

15. Mengakses Anggota Dictionary

Gunakan fungsi **get()** untuk mengakses

```
list_memotong.py dictionary_akses_anggota.py X
Modul3 > dictionary_akses_anggota.py > dict_saya
1 dict_saya = {'nama': 'Budi', 'usia': 27}
2
3 #output: Budi
4 print(dict_saya.get('nama'))
5
6 #output 27
7 print(dict_saya.get('usia'))
8
9 #output None
10 print(dict_saya.get('alamat'))
11
12 # Mengakses kunci yang tidak tersedia menyebabkan KeyError
13 # dict_saya['alamat']
```

16. Mengubah Anggota Dictionary

```
list_memotong.py dictionary_ubah_anggota.py X
Modul3 > dictionary_ubah_anggota.py > [e] dict_saya
1 dict_saya = {'nama': 'Ikhsan', 'usia': 35}
2
3 #mengupdate nilai
4 dict_saya['usia'] = 36
5 #output: {'nama': 'Ikhsan', 'usia': 36}
6 print(dict_saya)
7
8 #menambahkan anggota
9 dict_saya['alamat'] = 'Tanjungpinang'
10 # Output: {'alamat': 'Tanjungpinang', 'nama': 'Ikhsan', 'usia': 36}
11 print(dict_saya)
```

17. Menghapus Anggota Dictionary

Gunakan fungsi `pop()` untuk menghapus anggota tertentu, dan gunakan `popitem()` untuk menghapus semua anggota dictionary.

```
list_memotong.py dictionary_hapus_anggota.py X
Modul3 > dictionary_hapus_anggota.py > ...
1 # Membuat dictionary baru
2 dict_saya = {1:1, 2:4, 3:9, 4:16, 5:25}
3
4 # menghapus anggota tertentu
5 # output: 9
6 print(dict_saya.pop(3))
7
8 # menghapus anggota secara acak
9 #output: (5, 25)
10 print(dict_saya.popitem())
11
12 # yang tersisa adalah {1:1, 2:4, 4:16}
13 print(dict_saya)
14
15 # delete 5
16 del dict_saya[2]
17
18 # output: {2:4, 4:16}
19 print(dict_saya)
20
21 # menghapus semua anggota
```