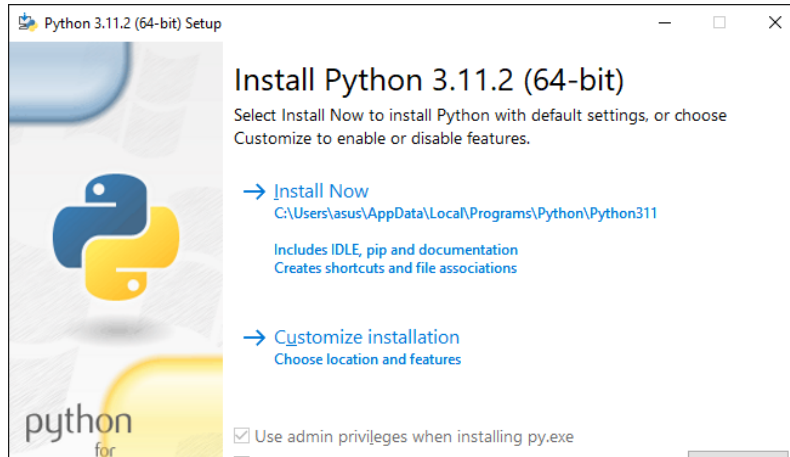
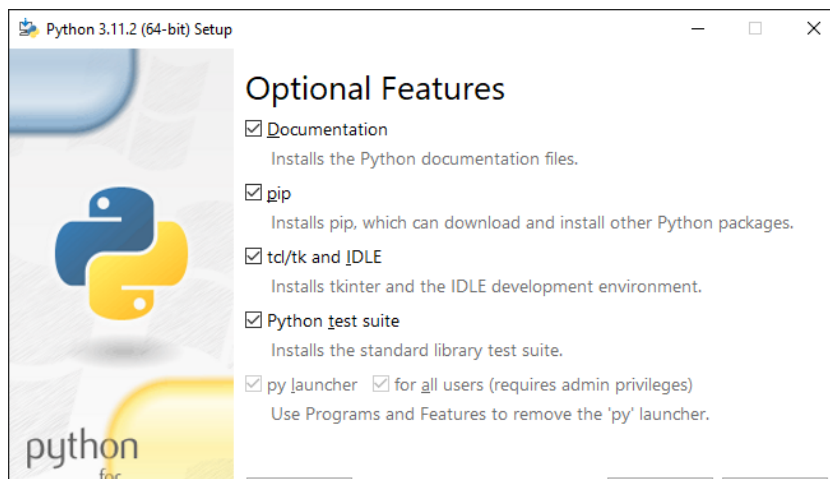


PROSES INSTALASI PYTHON

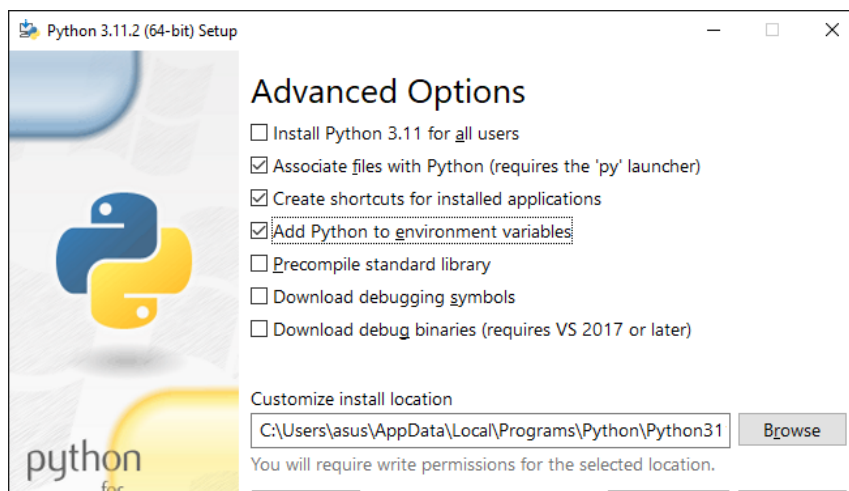
1. Pertama pilih versi Python yang sesuai dengan tipe sistem operasi yang digunakan, kemudian download dan double klik. Setelah itu akan muncul seperti gambar dibawah, selanjutnya pilih Customize installation.



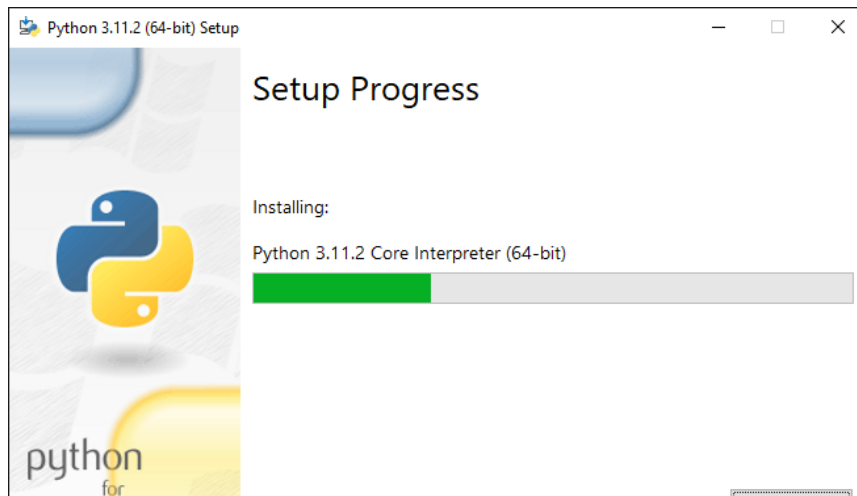
2. Kemudian klik next untuk melanjutkan proses instalasi.



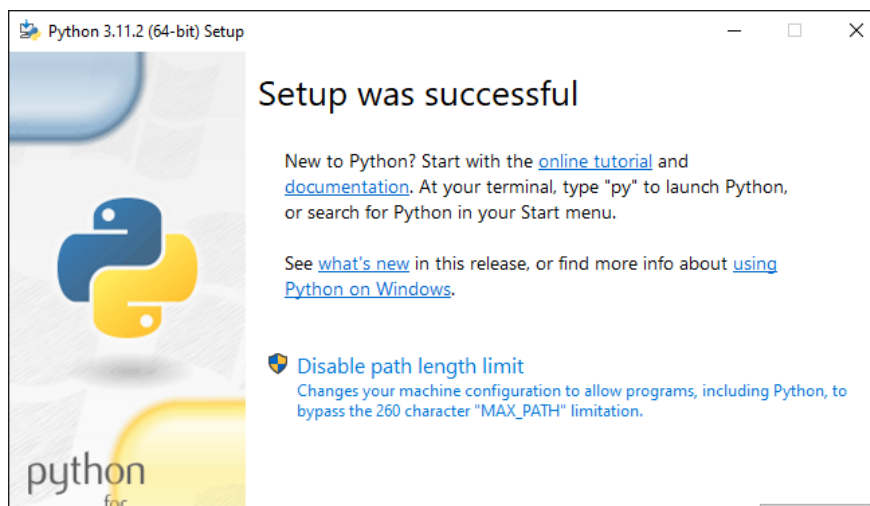
3. Selanjutnya atur direktori atau lokasi dimana Anda ingin menginstal Python, kemudian klik Install.



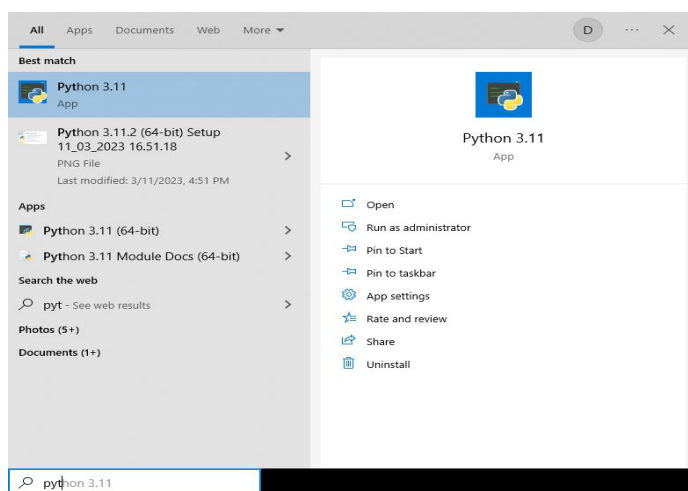
4. Tunggu hingga proses selesai.



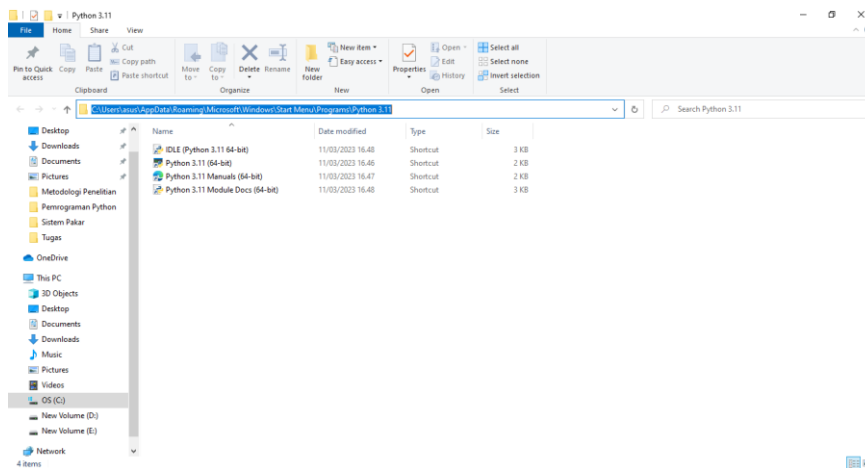
5. Setelah selesai dan sukses, klik Close.



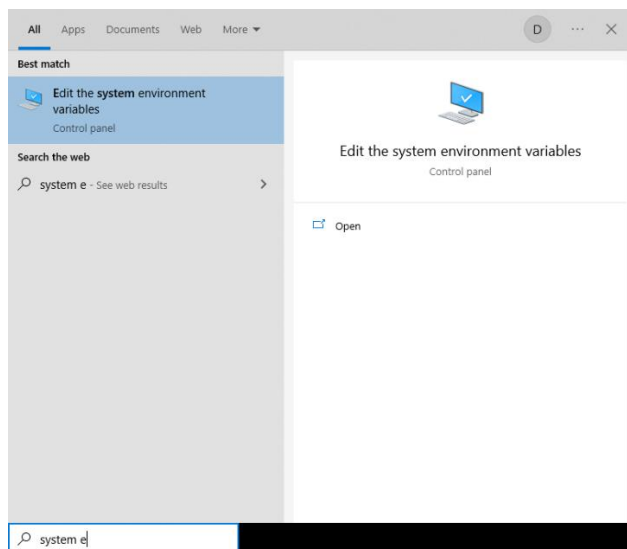
6. Kemudian kita bisa mengecek apakah Python sudah diinstal dengan melakukan searching atau pencarian di Windows.



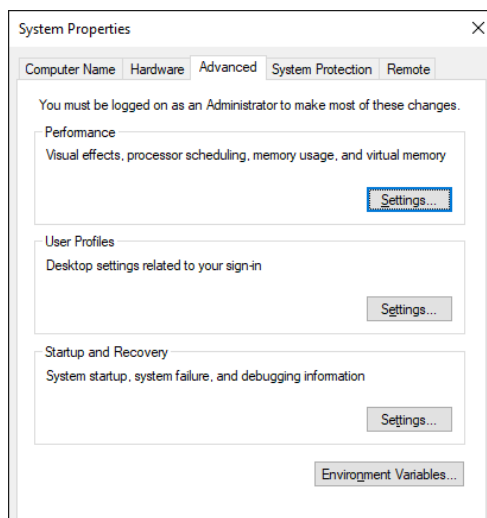
7. Kemudian copy alamat direktori Python sebagai lokasi mensetting Environment



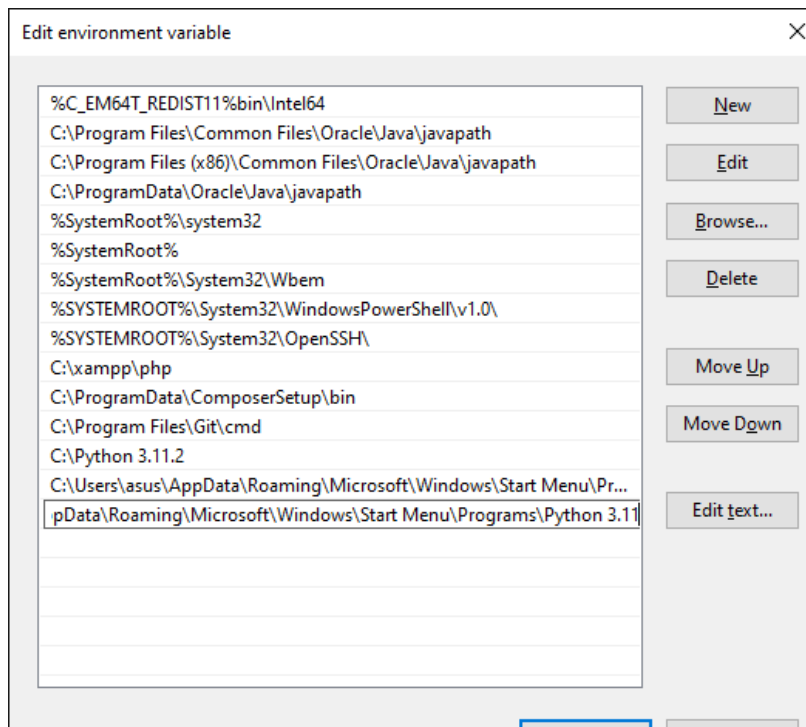
8. Kemudian buka System Environment Variables untuk men-setting Path. Buka Control Panel lalu cari System Environment Variables atau bisa langsung search melalui pencarian Windows seperti gambar dibawah.



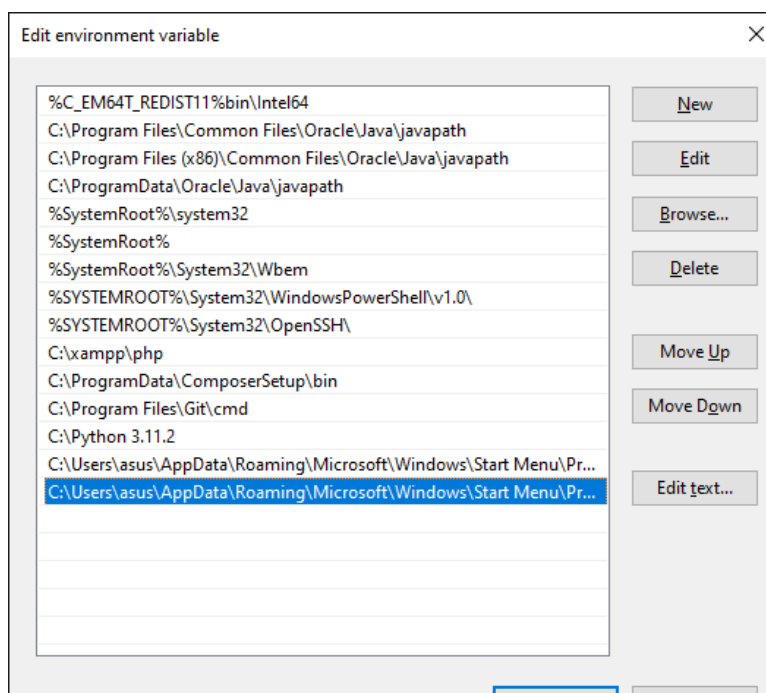
9. Setelah muncul seperti gambar dibawah klik Environment Variables.



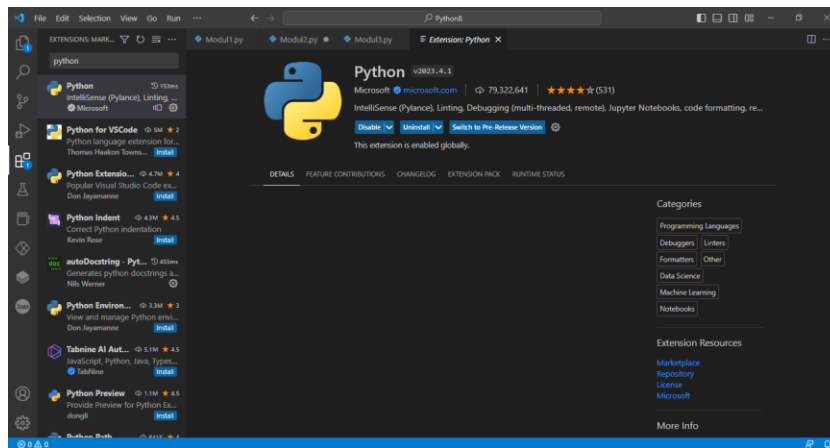
10. Pilih bagian System variables Path lalu klik Edit seperti pada gambar dibawah.



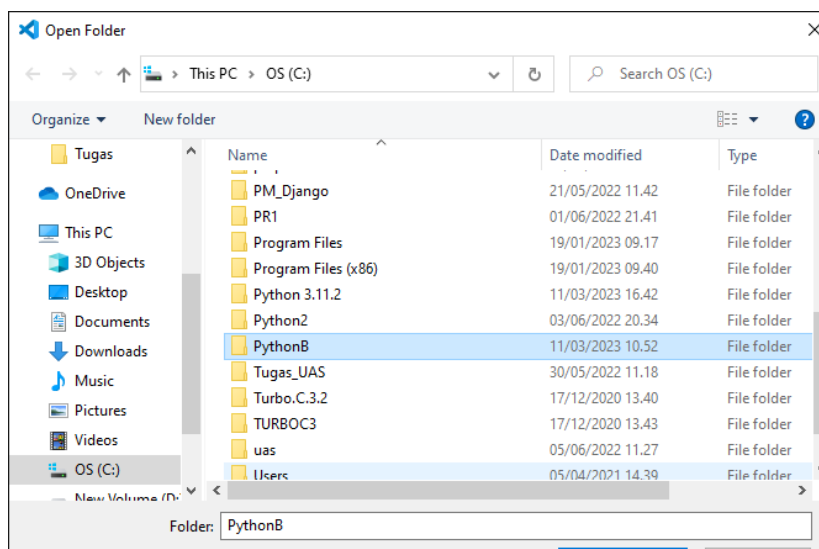
11. Klik tombol New kemudian paste alamat yang telah di copy sebelumnya sebagai lokasi, setelah selesai klik OK.



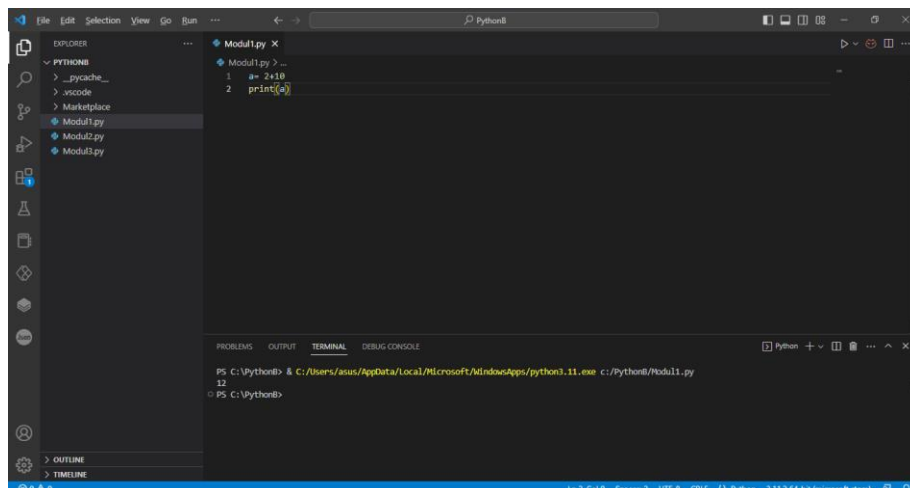
12. Kemudian buka VS Code yang sudah terinstal, lalu klik menu Extensions pada Activity Bar (sebelah kiri). Ketik python, kemudian Install dan reload VS Code, seperti pada gambar dibawah.



13. Pada bagian Side Bar, klik Open Folder untuk membuka folder untuk menjalankan folder proyek kemudian select folder, seperti contoh dibawah.



14. Setelah folder tampil. Klik New File untuk membuat file Python, Ketik nama file dengan ekstensi.py dibelakang nama file. Dan pilih menu Debug pada Activity Bar, kemudian jalankan project. Sepeti contoh dibawah file dengan nama modul1.py akan menampilkan penjumlahan sederhana seperti pada gambar dibawah.



MODUL 1

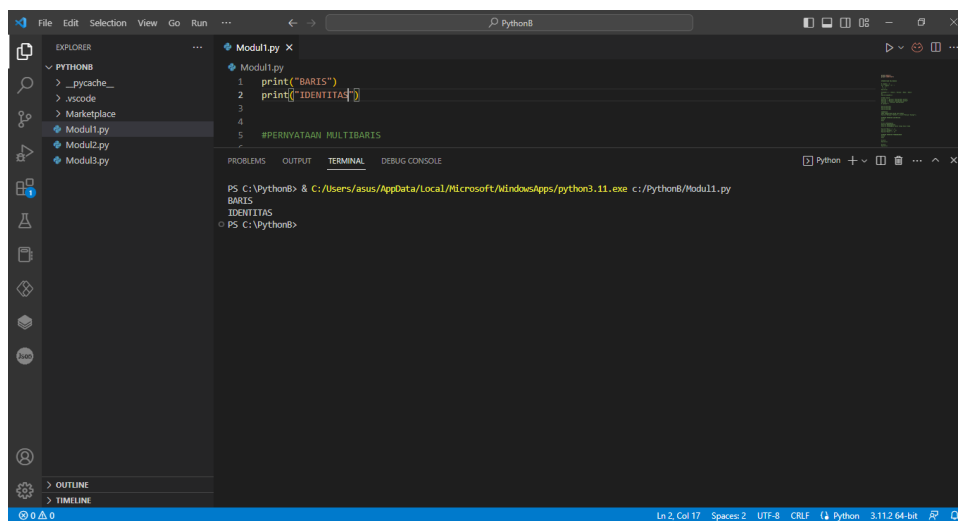
1. Identifier

Merupakan identitas atau nama yang telah diberikan kepada function, variabel, obyek, class, namespace dan lain-lain.

2. Baris dan Indentasi

Python tidak menggunakan tanda { } untuk menandai blok atau grup kode di python menggunakan tanda indentasi (spasi). Jumlah spasi untuk setiap baris yang ada dalam satu blok kode harus sama.

Contoh program baris dan identitas yang benar :



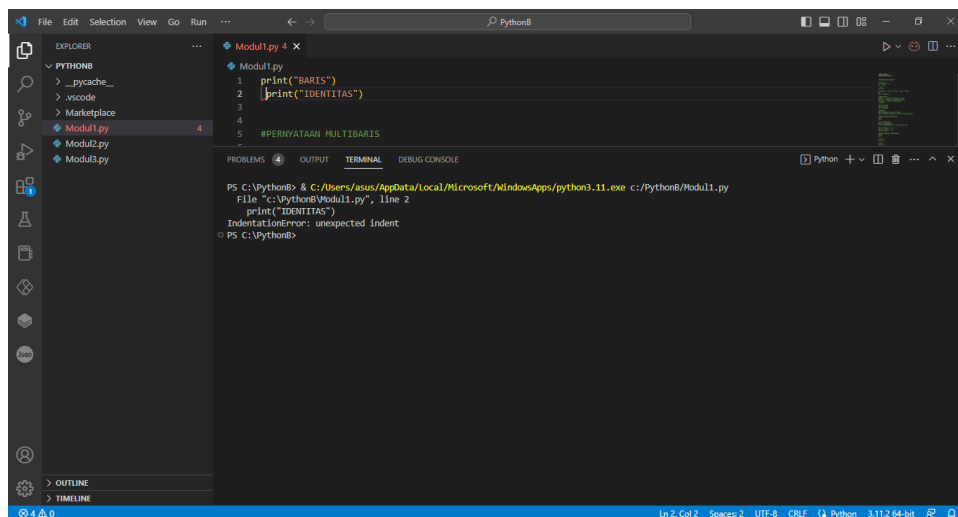
The screenshot shows the Visual Studio Code editor with a file named 'Modul1.py'. The code in the editor is as follows:

```
1 print("BARIS")
2 print("IDENTITAS")
3
4
5 #PERNYATAAN MULTIBARIS
```

The terminal output shows the execution of the script:

```
PS C:\Python> & C:/Users/asus/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Python8/Modul1.py
BARIS
IDENTITAS
PS C:\Python>
```

Contoh program salah baris dan identitas :



The screenshot shows the Visual Studio Code editor with a file named 'Modul1.py'. The code in the editor is as follows:

```
1 print("BARIS")
2 | print("IDENTITAS")
3
4
5 #PERNYATAAN MULTIBARIS
```

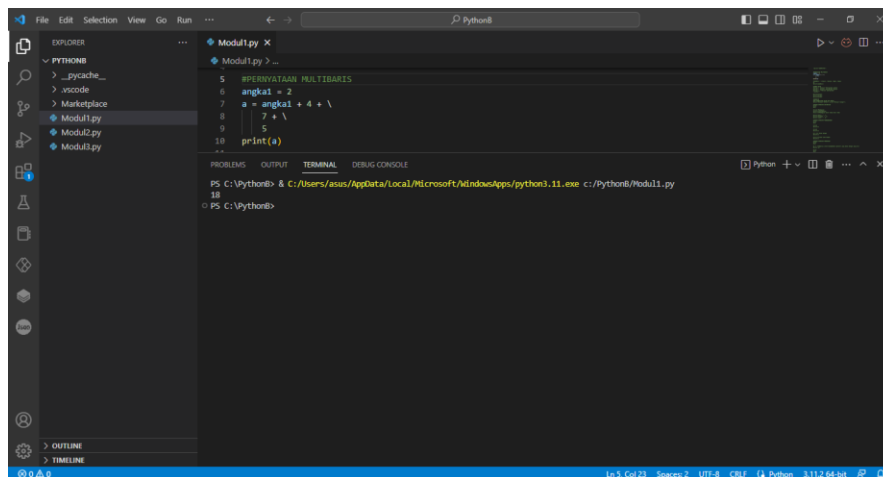
The terminal output shows an error message:

```
PS C:\Python> & C:/Users/asus/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Python8/Modul1.py
File "c:\Python8\Modul1.py", line 2
  print("IDENTITAS")
  ^
IndentationError: unexpected indent
PS C:\Python>
```

3. Pernyataan Multibaris

Digunakan untuk membuat pernyataan yang terdiri dari beberapa baris menggunakan tanda backslash (\).

Contoh program pernyataan multibaris:



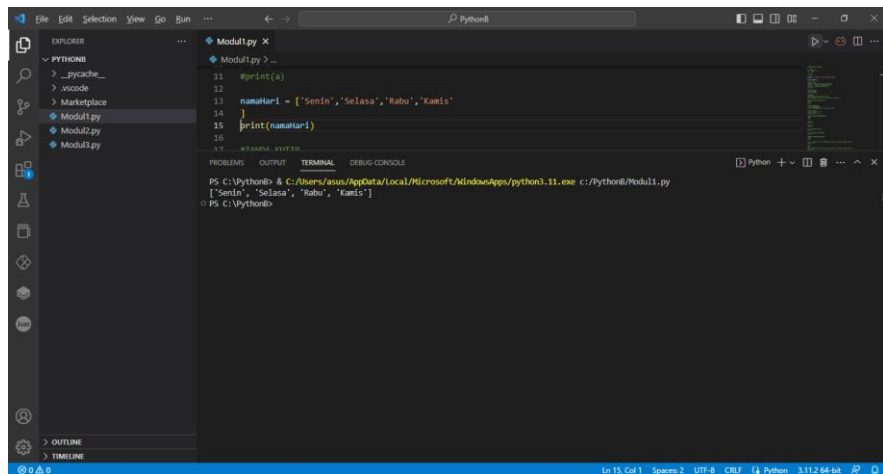
The screenshot shows a VS Code editor with a Python file named 'Modul1.py'. The code contains a multi-line statement for calculating the sum of two numbers. The code is as follows:

```
5 #PERNYATAAN MULTIBARIS
6 angka1 = 2
7 a = angka1 + 4 + \
8     7 + \
9     5
10 print(a)
```

The terminal output shows the result of the execution:

```
PS C:\Python8> & C:/Users/asus/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Python8/Modul1.py
18
PS C:\Python8>
```

Contoh program statement multibaris dalam tanda kurung :



The screenshot shows a VS Code editor with a Python file named 'Modul1.py'. The code contains a multi-line statement for printing a list of days of the week. The code is as follows:

```
11 #print(a)
12
13 namahari = ('Senin','Selasa','Rabu','Kamis')
14
15 print(namahari)
```

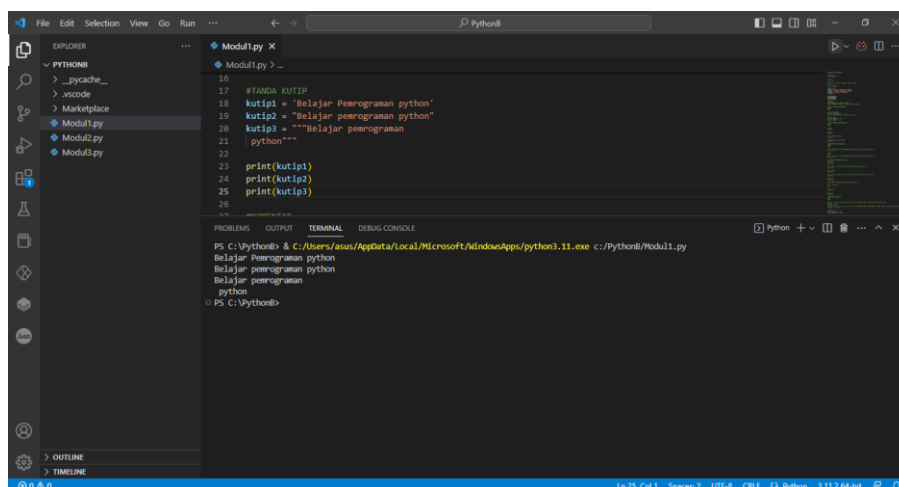
The terminal output shows the result of the execution:

```
PS C:\Python8> & C:/Users/asus/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Python8/Modul1.py
['Senin', 'Selasa', 'Rabu', 'Kamis']
PS C:\Python8>
```

4. Tanda Kutip

Digunakan untuk menandai string dimana tanda kutip tiga digunakan untuk string multibaris.

Contoh program penggunaan tanda kutip:



The screenshot shows a VS Code editor with a Python file named 'Modul1.py'. The code contains a multi-line statement for printing a string using triple quotes. The code is as follows:

```
16
17 #TANDA KUTIP
18 kutip1 = "Belajar Pemrograman python"
19 kutip2 = "Belajar pemrograman python"
20 kutip3 = """Belajar pemrograman
21 python"""
22
23 print(kutip1)
24 print(kutip2)
25 print(kutip3)
```

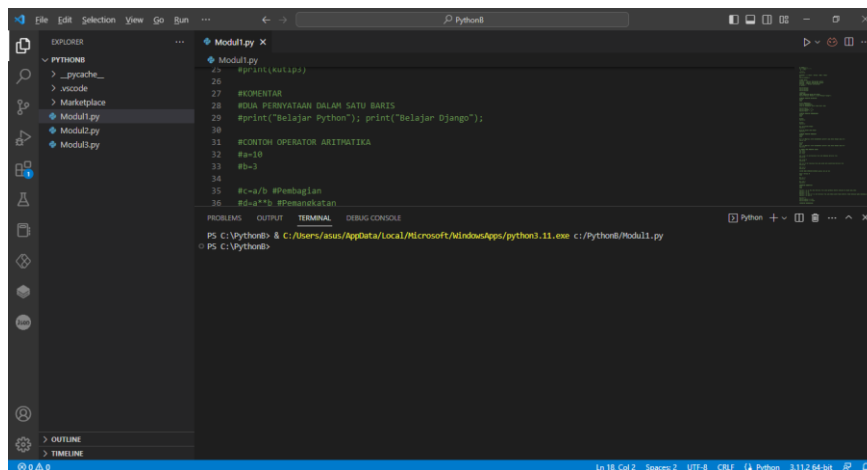
The terminal output shows the result of the execution:

```
PS C:\Python8> & C:/Users/asus/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Python8/Modul1.py
Belajar Pemrograman python
Belajar pemrograman python
Belajar pemrograman
python
PS C:\Python8>
```

5. Komentar

Tanda pagar (#) atau komentar digunakan untuk menandai komentar di python.

Komentar tidak akan diproses oleh interpreter Python.



```
Modul1.py
26 #print(kulipa)
27
28 #KOMENTAR
29 #DUA PERNYATAAN DALAM SATU BARIS
30 #print("Belajar Python"); print("Belajar Django");
31
32 #CONTOH OPERATOR ARITMATIKA
33 #a=10
34 #b=3
35 #c=a/b #Pembagian
36 #d=a**b #Pangkatatan
```

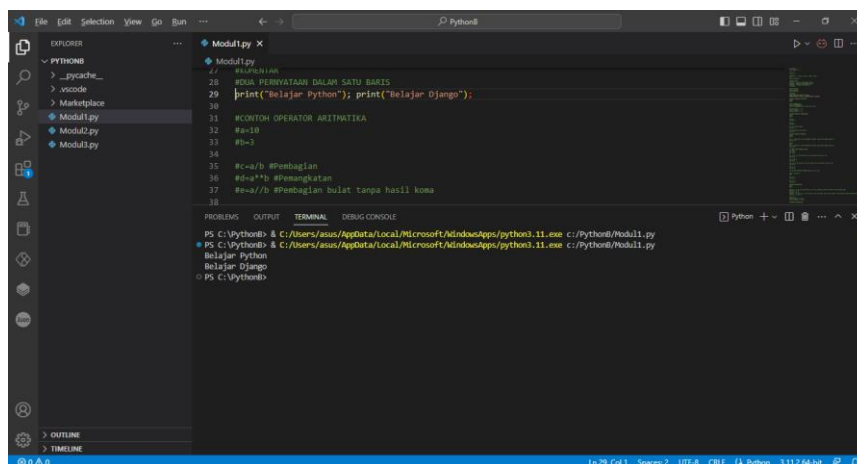
Terminal output:

```
PS C:\Python> & C:/Users/asus/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Python/Modul1.py
Belajar Python
Belajar Django
PS C:\Python>
```

6. Dua Pernyataan Dalam Satu Baris

Titik koma dapat digunakan ketika terdapat 2 pernyataan dalam 1 baris kode.

Contoh program dua pernyataan dalam satu baris :



```
Modul1.py
26 #KOMENTAR
27 #DUA PERNYATAAN DALAM SATU BARIS
28 #print("Belajar Python"); print("Belajar Django");
29
30 #CONTOH OPERATOR ARITMATIKA
31 #a=10
32 #b=3
33 #c=a/b #Pembagian
34 #d=a**b #Pangkatatan
35 #e=a//b #Pembagian bulat tanpa hasil koma
36
```

Terminal output:

```
PS C:\Python> & C:/Users/asus/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Python/Modul1.py
Belajar Python
Belajar Django
PS C:\Python>
```


Tipe Data

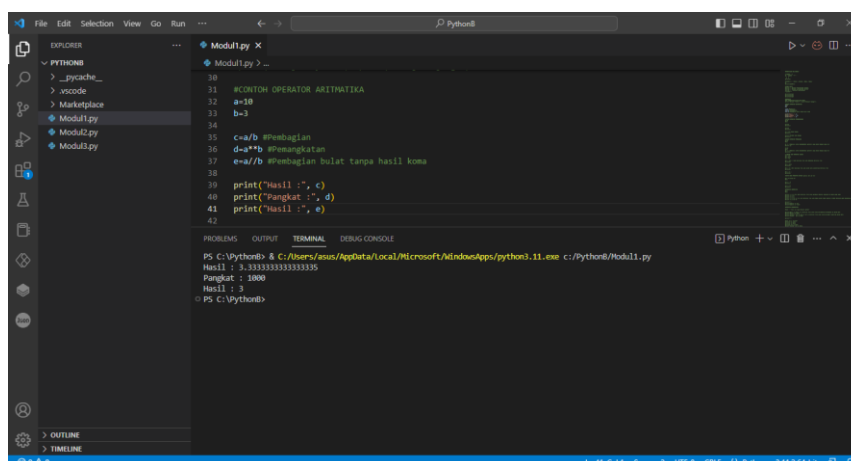
Macam tipe data antara lain:

- int, tipe data yang dapat Anda isi dengan bilangan bulat.
- float, tipe data yang dapat Anda isi dengan bilangan koma.
- string, tipe data yang dapat Anda isi dengan sebuah karakter atau kalimat.
- complex, tipe data bilangan kompleks atau bilangan imajiner, seperti 5j, 54j, 1j.
- long, tipe data yang dapat diisi dengan bilangan yang sangat besar.
- boolean, tipe data yang nilainya hanya True dan False.
- List, adalah tipe data yang berisi item yang berurut.
- Tuple, adalah jenis data lain yang mirip dengan list. Perbedaannya dengan list adalah anggotanya tidak bisa diubah (immutable). Tuple dideklarasikan dengan menggunakan tanda kurung (). dan anggotanya dipisahkan oleh tanda koma.
- Set, adalah salah satu tipe data yang tidak berurut (unordered).
- Dictionary, adalah tipe data yang tiap anggotanya terdiri dari pasangan kunci:nilai (key-value). Dictionary dideklarasikan dengan menggunakan tanda kurung kurawal { }, dimana anggotanya memiliki bentuk kunci:nilai atau key:value dan tiap anggota dipisah tanda koma. Kunci dan nilainya bisa memiliki tipe sembarang.

1. Operator aritmatika

Digunakan untuk melakukan operasi matematika, seperti penjumlahan, pengurangan, perkalian, pembagian, dan sebagainya.

Contoh Kode Operator Aritmatika



```
30
31 #CONTOH OPERATOR ARITMATIKA
32 a=10
33 b=3
34
35 c=a/b #Pembagian
36 d=a*b #perkalian
37 e=a/b #pembagian bulat tanpa hasil koma
38
39 print("Hasil :", c)
40 print("Pangkat :", d)
41 print("Hasil :", e)
42
```

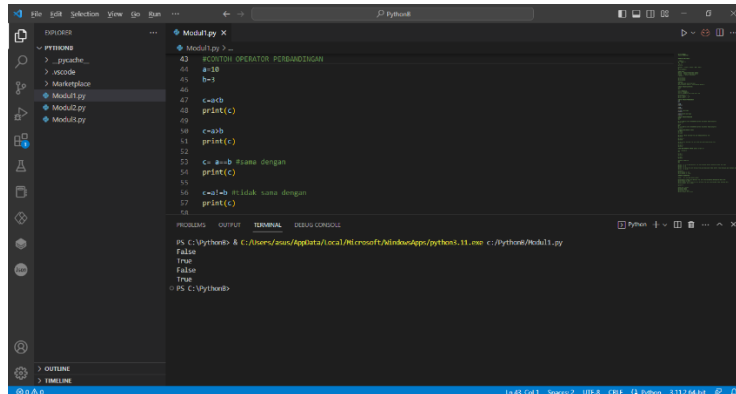
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
PS C:\Python> & C:\Users\mms/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Python/Modul1.py
Hasil : 3.3333333333333335
Pangkat : 1000
Hasil : 3
PS C:\Python>
```

2. Operator Perbandingan

Digunakan untuk membandingkan 2 buah nilai. Hasil perbandingannya antara True atau False tergantung kondisi.

Contoh program operator perbandingan :



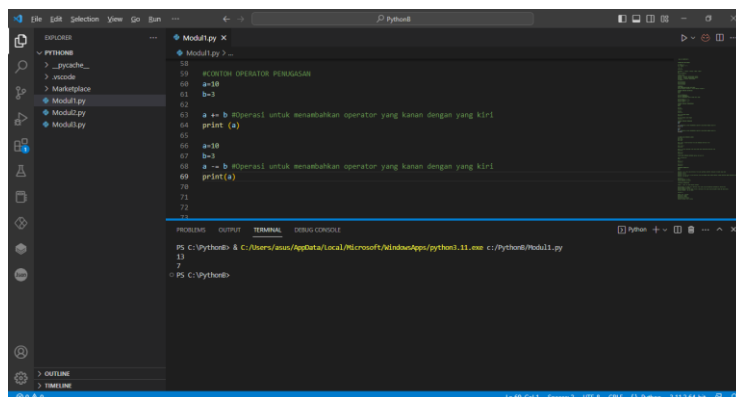
```
43 #CONTOH OPERATOR PERBANDINGAN
44 a=10
45 b=1
46
47 c=a<b
48 print(c)
49
50 c=a>b
51 print(c)
52
53 c=a==b #sama dengan
54 print(c)
55
56 c=a!=b #tidak sama dengan
57 print(c)
58
```

Terminal output:

```
PS C:\Python> python c:/Python/Modul1.py
False
True
False
True
PS C:\Python>
```

3. Operator Penugasan

Digunakan untuk memberi nilai ke variabel. Contoh program kode operator penugasan



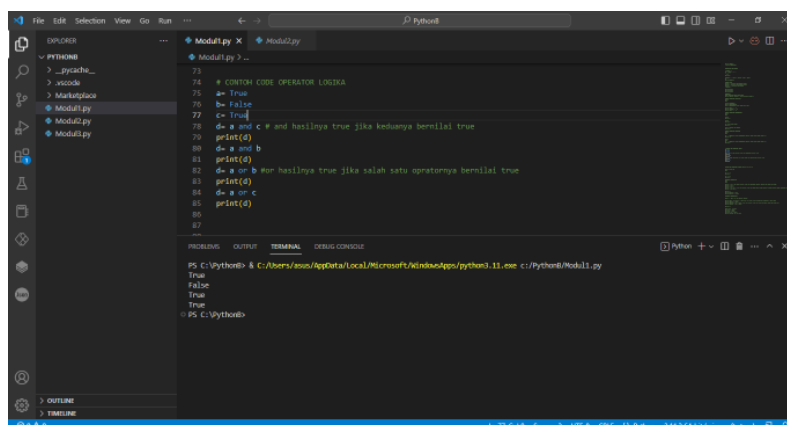
```
59 #CONTOH OPERATOR PENUGASAN
60 a=10
61 b=2
62
63 a += b #operasi untuk menambahkan operator yang kanan dengan yang kiri
64 print(a)
65
66 a=10
67 b=1
68 a -= b #operasi untuk menambahkan operator yang kanan dengan yang kiri
69 print(a)
70
71
72
73
```

Terminal output:

```
PS C:\Python> python c:/Python/Modul1.py
12
PS C:\Python>
```

4. Operator Logika

Digunakan untuk melakukan operasi logika. Contoh program kode operator logika :



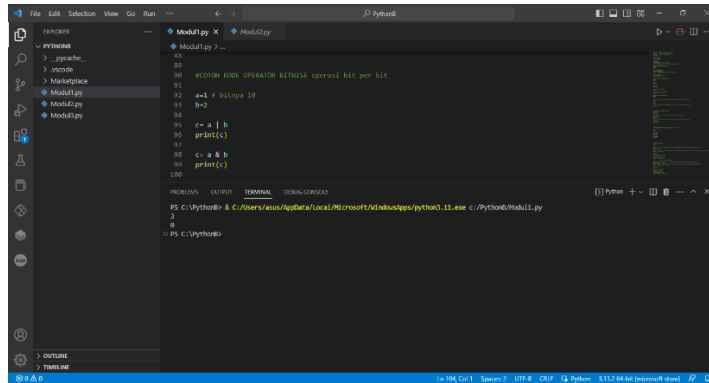
```
74 # CONTOH CODE OPERATOR LOGIKA
75 a= True
76 b= False
77 c= True
78 d= a and c # and hasilnya true jika keduanya bernilai true
79 print(d)
80 d= a and b
81 print(d)
82 d= a or b # or hasilnya true jika salah satu operatonya bernilai true
83 print(d)
84 d= a or c
85 print(d)
86
87
```

Terminal output:

```
PS C:\Python> python c:/Python/Modul1.py
True
False
True
True
True
PS C:\Python>
```

5. Operator Bitwise

Operator yang melakukan operasi bit terhadap operand, operator ini beroperasi bit per bit. Contoh program kode operator bitwise :

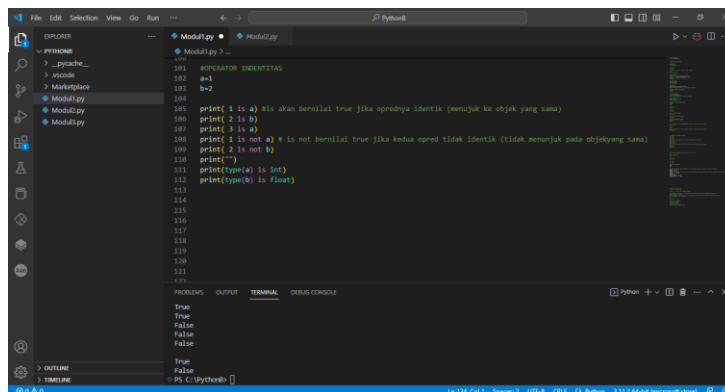


```
104
105
106 # OPERATOR KODE OPERATOR BITWISE operasi bit per bit
107
108 a=1 + binary 10
109 b=2
110
111 c= a | b
112 print(c)
113
114 c= a & b
115 print(c)
116
117
```

The screenshot shows a Python IDE with a file named 'Modul1.py'. The code defines two variables, 'a' and 'b', with binary values 10 and 2 respectively. It then performs bitwise OR ('|'), AND ('&'), and XOR ('^') operations, printing the results. The terminal output shows the results of these operations.

6. Operator Identitas

Adalah operator yang memeriksa apakah dua buah nilai (atau variabel) berada pada lokasi memori yang sama. Contoh program kode operator identitas:



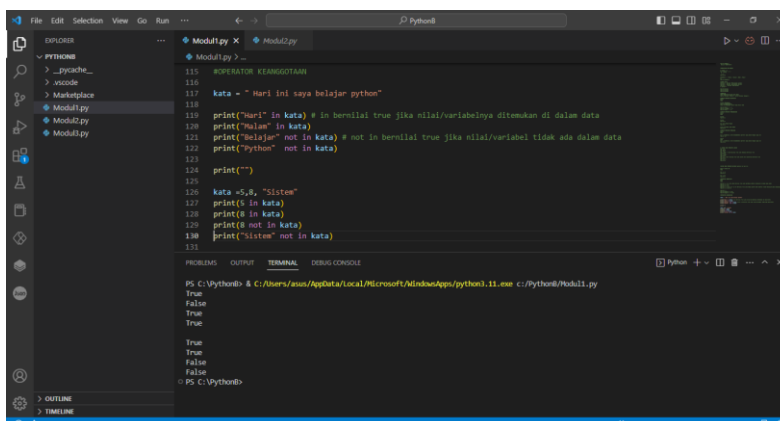
```
101 OPERATOR IDENTITAS
102 a=1
103 b=2
104
105 print(1 is a) #is akan bernilai true jika operandnya identik (menunjuk ke objek yang sama)
106 print(2 is b)
107 print(3 is a)
108 print(3 is not a) # is not bernilai true jika kedua operand tidak identik (tidak menunjuk pada objek yang sama)
109 print(2 is not b)
110 print("")
111 print(type(a) is int)
112 print(type(b) is float)
113
114
115
116
117
118
119
120
121
122
123
```

The screenshot shows a Python IDE with a file named 'Modul1.py'. The code demonstrates the 'is' and 'is not' operators by comparing variables and their types. The terminal output shows the results of these comparisons.

7. Operator Keanggotaan

Digunakan untuk memeriksa apakah suatu nilai atau variabel merupakan anggota atau ditemukan di dalam suatu data (string, list, tuple, set, dan dictionary).

Contoh program kode operator keanggotaan :



```
110 OPERATOR KEANGGOTAAN
111
112 kata = " Hari ini saya belajar python"
113
114 print("hari" in kata) # in bernilai true jika nilai/variabelnya ditemukan di dalam data
115 print("bulan" in kata)
116 print("belajar" not in kata) # not in bernilai true jika nilai/variabel tidak ada dalam data
117 print("python" not in kata)
118
119 print("")
120
121 kata =5,8, "sistem"
122 print(5 in kata)
123 print(8 in kata)
124 print(8 not in kata)
125 print("sistem" not in kata)
126
127
```

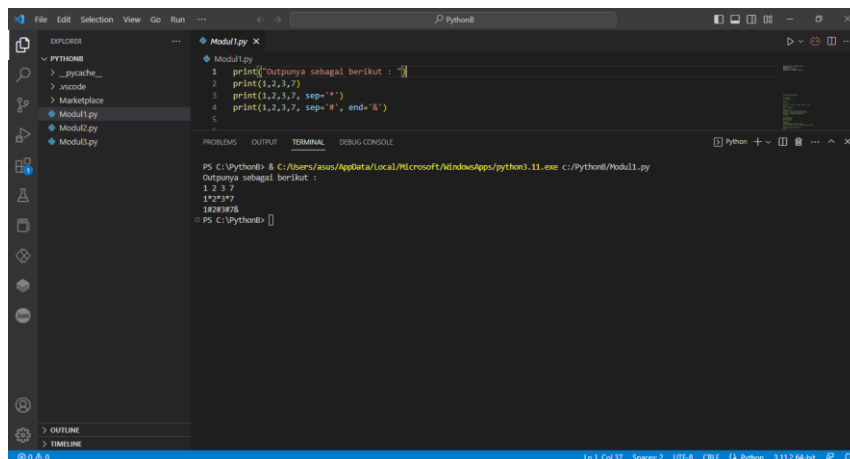
The screenshot shows a Python IDE with a file named 'Modul1.py'. The code demonstrates the 'in' and 'not in' operators by checking if a string or a value is a member of a string or a list. The terminal output shows the results of these membership checks.

MODUL 2

1. Input dan Output

a. Output

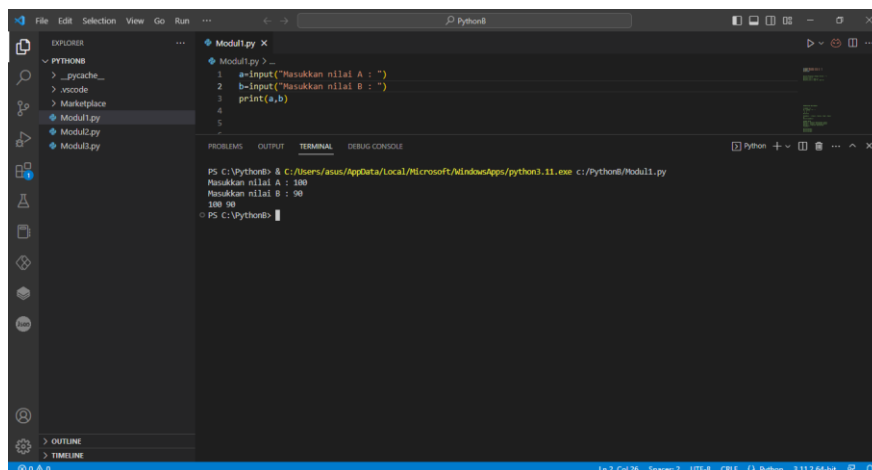
Disini terdapat fungsi output dimana fungsi yang biasa digunakan adalah `print()`. `Print` sendiri memiliki fungsi untuk menampilkan hasil atau output dari program yang kita buat. Dimana terdapat `sep` berfungsi sebagai tanda pemisah antar objek yang dicetak. `end` merupakan karakter yang akan dicetak di akhir baris. file adalah nama file kemana objek akan dicetak. Contoh program kode output :



```
File Edit Selection View Go Run ... PythonB
EXPLORER
PYTHONB
  > __pycache__
  > .vscode
  > Marketplace
  > Modul1.py
  > Modul2.py
  > Modul3.py
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
Python
PS C:\PythonB> & C:/Users/asus/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/PythonB/Modul1.py
Outputnya sebagai berikut :
1 2 3 7
1*2=3*7
1*2=3*7
PS C:\PythonB>
```

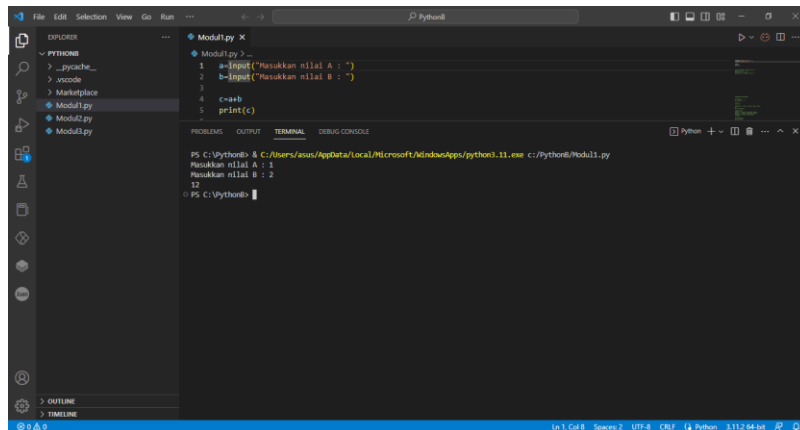
b. Input

Fungsi input ini biasanya digunakan adalah `input()` untuk menginput masukan dari program yang kita buat. Input, proses, dan output adalah inti dari semua program komputer. Contoh program kode input :



```
File Edit Selection View Go Run ... PythonB
EXPLORER
PYTHONB
  > __pycache__
  > .vscode
  > Marketplace
  > Modul1.py
  > Modul2.py
  > Modul3.py
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
Python
PS C:\PythonB> & C:/Users/asus/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/PythonB/Modul1.py
Masukkan nilai A : 100
Masukkan nilai B : 90
200 90
PS C:\PythonB>
```

Contoh program kode input tanpa menggunakan fungsi int() :

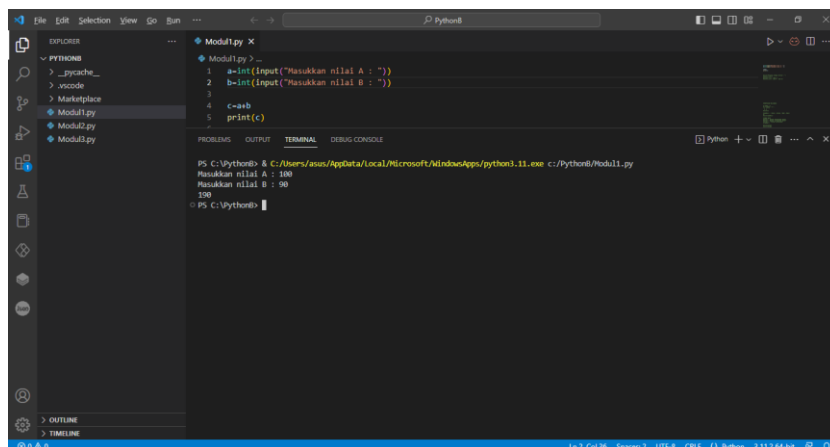


```
Modul1.py > ...
1 a=input("Masukkan nilai A : ")
2 b=input("Masukkan nilai B : ")
3
4 c=a+b
5 print(c)
```

Terminal output:

```
PS C:\Python8> & C:\Users\asus\AppData\Local\Microsoft\WindowsApps\python3.11.exe c:/Python8/Modul1.py
Masukkan nilai A : 1
Masukkan nilai B : 2
12
PS C:\Python8>
```

Contoh program kode input pertama menggunakan fungsi int() :

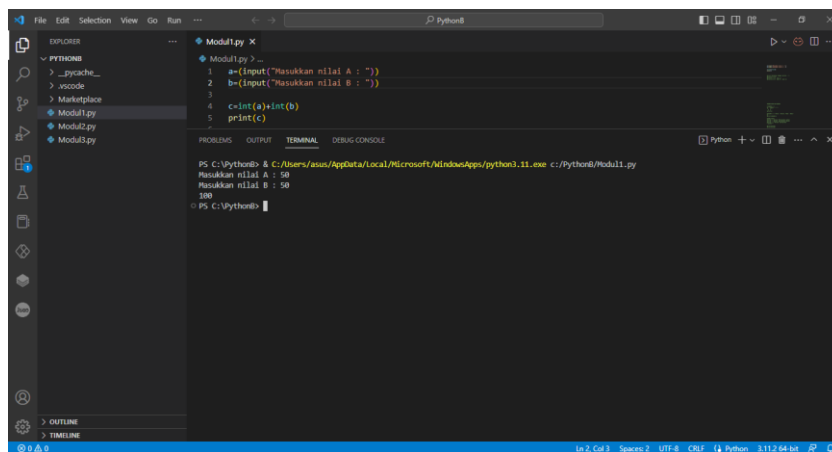


```
Modul1.py > ...
1 a=int(input("Masukkan nilai A : "))
2 b=input("Masukkan nilai B : ")
3
4 c=a+b
5 print(c)
```

Terminal output:

```
PS C:\Python8> & C:\Users\asus\AppData\Local\Microsoft\WindowsApps\python3.11.exe c:/Python8/Modul1.py
Masukkan nilai A : 100
Masukkan nilai B : 90
190
PS C:\Python8>
```

Contoh program kode input kedua menggunakan fungsi int() :

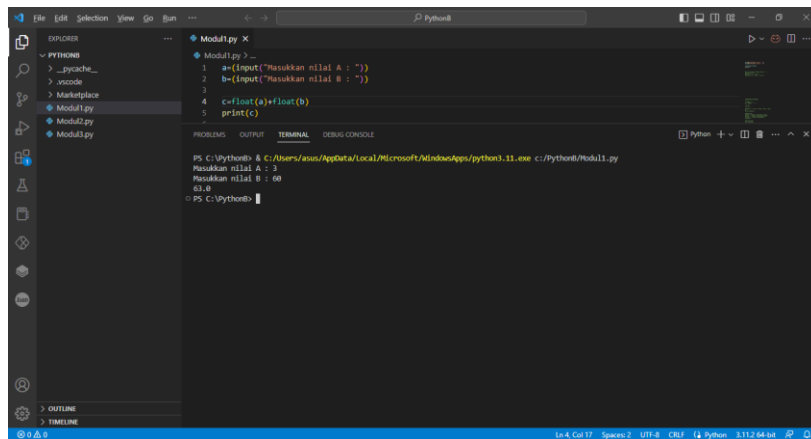


```
Modul1.py > ...
1 a=int(input("Masukkan nilai A : "))
2 b=int(input("Masukkan nilai B : "))
3
4 c=int(a)+int(b)
5 print(c)
```

Terminal output:

```
PS C:\Python8> & C:\Users\asus\AppData\Local\Microsoft\WindowsApps\python3.11.exe c:/Python8/Modul1.py
Masukkan nilai A : 50
Masukkan nilai B : 50
100
PS C:\Python8>
```

Contoh program kode input menggunakan fungsi float() untuk hasil pecahan :



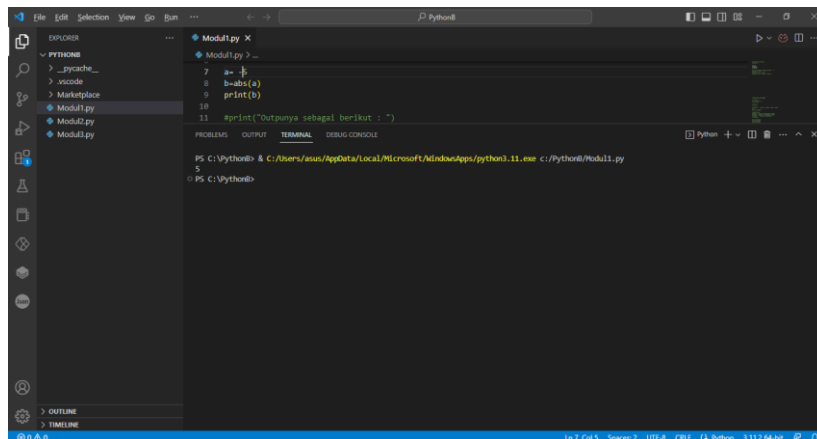
The screenshot shows a VS Code editor with a file named 'Modul1.py'. The code in the editor is as follows:

```
1 a=input("Masukkan nilai A : ")
2 b=input("Masukkan nilai B : ")
3
4 c=float(a)+float(b)
5 print(c)
```

The terminal output shows the execution of the program:

```
PS C:\Python> & C:/Users/asus/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Python/Modul1.py
Masukkan nilai A : 3
Masukkan nilai B : 60
63.0
PS C:\Python>
```

Contoh program kode input menggunakan fungsi abs() dinamis untuk menghilangkan tipe data yang ada minusnya :



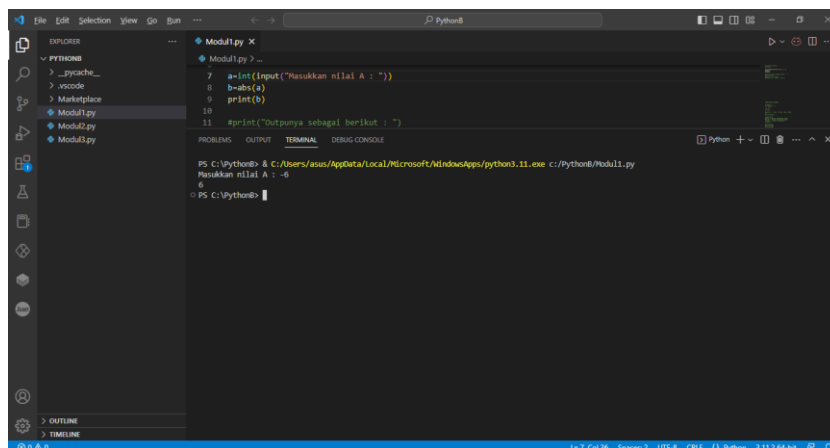
The screenshot shows a VS Code editor with a file named 'Modul1.py'. The code in the editor is as follows:

```
7 a= float(input("Masukkan nilai A : "))
8 b=abs(a)
9 print(b)
10
11 #print("Outputnya sebagai berikut : ")
```

The terminal output shows the execution of the program:

```
PS C:\Python> & C:/Users/asus/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Python/Modul1.py
Masukkan nilai A : -6
6
PS C:\Python>
```

Contoh program kode input menggunakan fungsi abs() statis untuk menghilangkan tipe data yang ada minusnya :



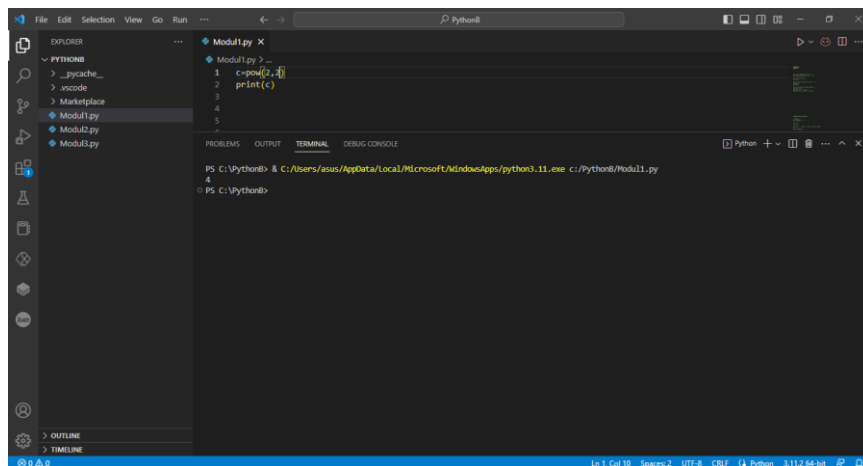
The screenshot shows a VS Code editor with a file named 'Modul1.py'. The code in the editor is as follows:

```
7 a= int(input("Masukkan nilai A : "))
8 b=abs(a)
9 print(b)
10
11 #print("Outputnya sebagai berikut : ")
```

The terminal output shows the execution of the program:

```
PS C:\Python> & C:/Users/asus/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Python/Modul1.py
Masukkan nilai A : -6
6
PS C:\Python>
```

Contoh program kode input menggunakan fungsi pow() Statis untuk menghitung pangkat :

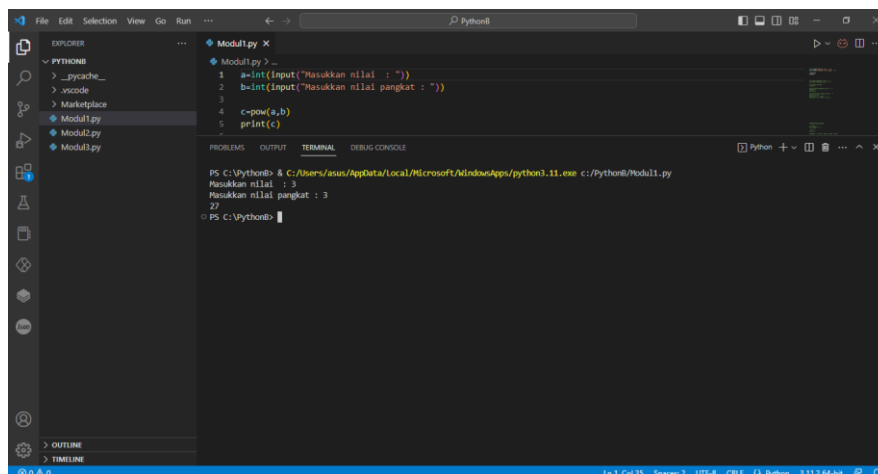


```
Modul1.py > ...
1 c=pow(2,4)
2 print(c)
3
4
5
6
```

PS C:\Python8> & C:/Users/asus/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Python8/Modul1.py

PS C:\Python8>

Contoh program kode input menggunakan fungsi pow() Statis untuk menghitung pangkat :



```
Modul1.py > ...
1 a=int(input("Masukkan nilai : "))
2 b=int(input("Masukkan nilai pangkat : "))
3
4 c=pow(a,b)
5 print(c)
6
```

PS C:\Python8> & C:/Users/asus/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Python8/Modul1.py

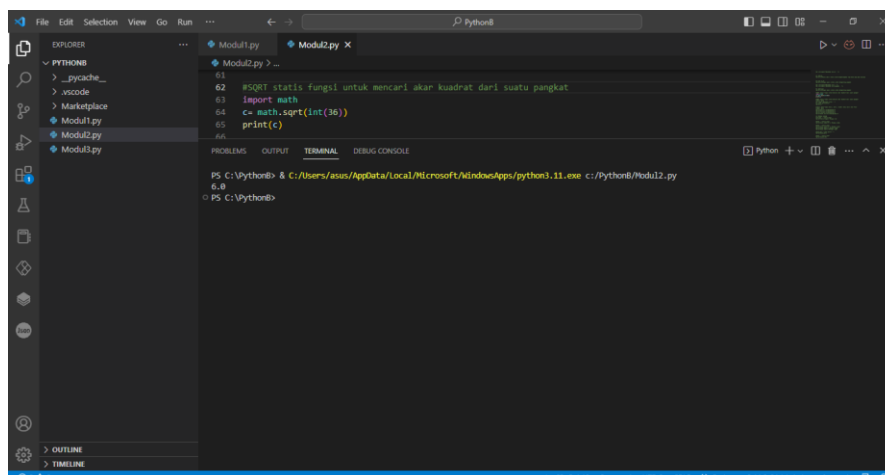
Masukkan nilai : 3

Masukkan nilai pangkat : 3

27

PS C:\Python8>

Contoh program kode input menggunakan sqrt() statis. Fungsi untuk mencari akar kuadrat dari suatu pangkat.



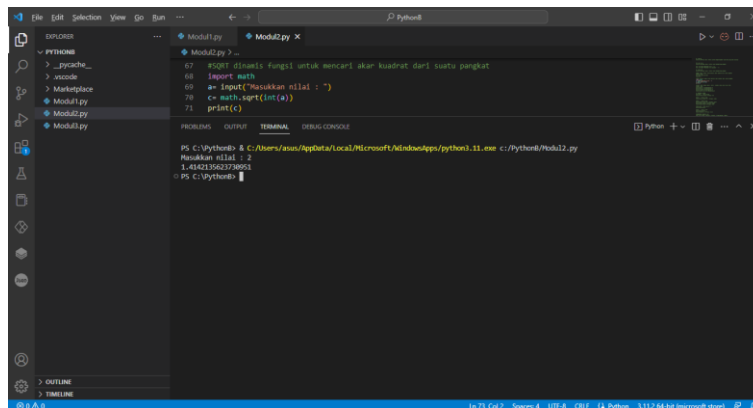
```
Modul2.py > ...
61
62 #SQRT statis fungsi untuk mencari akar kuadrat dari suatu pangkat
63
64 import math
65 c= math.sqrt(int(36))
66 print(c)
67
```

PS C:\Python8> & C:/Users/asus/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Python8/Modul2.py

6.0

PS C:\Python8>

Contoh program kode input menggunakan sqrt() dinamis. Fungsi untuk mencari akar kuadrat dari suatu pangkat.

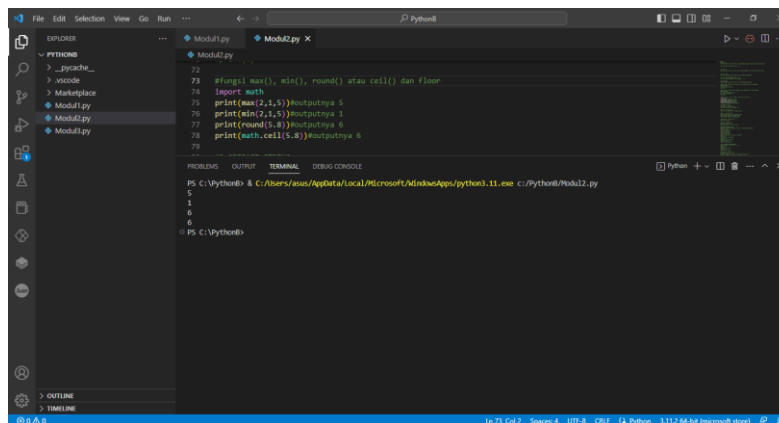


```
67 #SQRT dinamis fungsi untuk mencari akar kuadrat dari suatu pangkat
68 import math
69 a= input("Masukkan nilai : ")
70 c= math.sqrt(int(a))
71 print(c)
```

Terminal output:

```
PS C:\Python> & C:\Users\asus\AppData\Local\Microsoft\WindowsApps\python3.11.exe c:\Python\Modul2.py
Masukkan nilai : 2
1.4142135623730951
PS C:\Python>
```

Contoh lain program kode input menggunakan sqrt(). 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.



```
72
73 #fungsi max(), min(), round() atau ceil() dan floor
74 import math
75 print(max(2,1,5))#outputnya 5
76 print(min(2,1,5))#outputnya 1
77 print(round(5.8))#outputnya 6
78 print(math.ceil(5.8))#outputnya 6
79
```

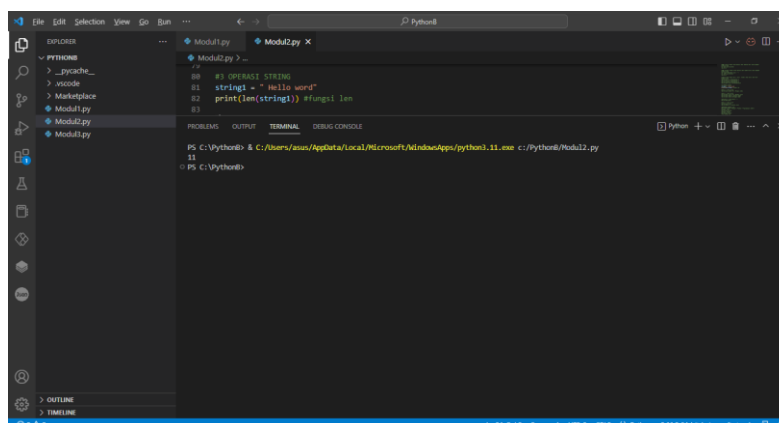
Terminal output:

```
PS C:\Python> & C:\Users\asus\AppData\Local\Microsoft\WindowsApps\python3.11.exe c:\Python\Modul2.py
5
1
6
6
PS C:\Python>
```

c. Operasi String

len() berfungsi untuk mengembalikan panjang (jumlah anggota) dari suatu objek.

Contoh program fungsi len():

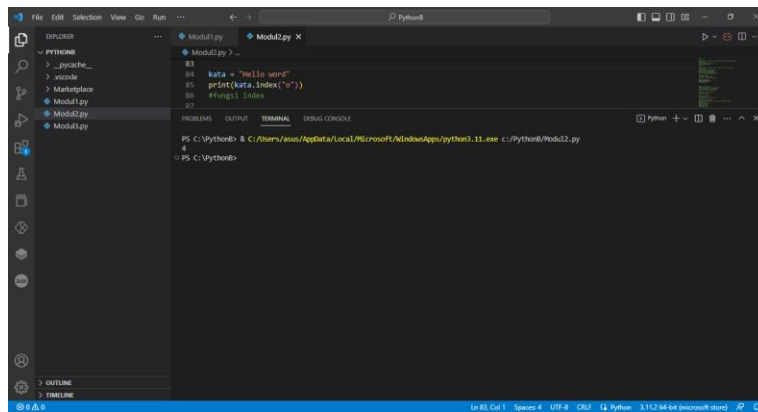


```
80 #Jumlah string
81 string1 = " Hello word"
82 print(len(string1)) #fungsi len
83
```

Terminal output:

```
PS C:\Python> & C:\Users\asus\AppData\Local\Microsoft\WindowsApps\python3.11.exe c:\Python\Modul2.py
11
PS C:\Python>
```

Fungsi index() untuk mencari posisi suatu nilai. Contoh program fungsi index():



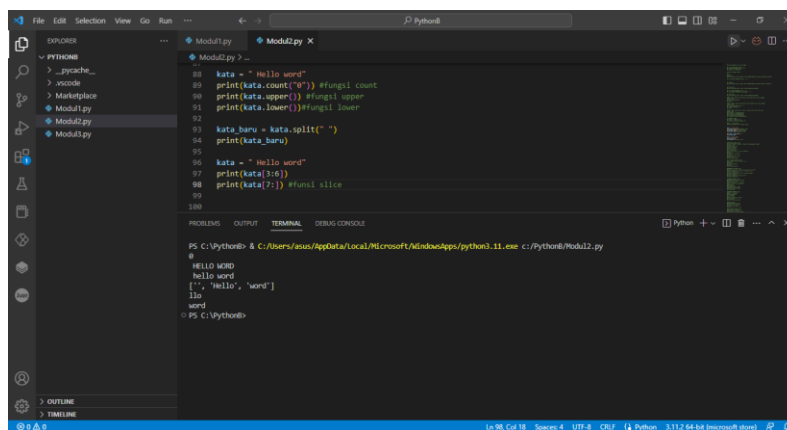
```
83 kata = "Hello word"
84 print(kata.index("o"))
85 #fungsi index
86
```

Terminal output:

```
PS C:\Python> & C:/Users/assu/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Python/Modul2.py
4
PS C:\Python>
```

Fungsi lain juga terdiri dari count() untuk menghitung kemunculan nilai tertentu. upper() untuk mengubah string menjadi huruf kapital. lower() untuk mengubah string menjadi huruf kecil. split() untuk memisah string menjadi item.

Contoh program:



```
88 kata = "Hello word"
89 print(kata.count("o")) #fungsi count
90 print(kata.upper()) #fungsi upper
91 print(kata.lower()) #fungsi lower
92
93 kata_baru = kata.split(" ")
94 print(kata_baru)
95
96 kata = "Hello word"
97 print(kata[3:6])
98 print(kata[7:]) #fungsi slice
99
100
```

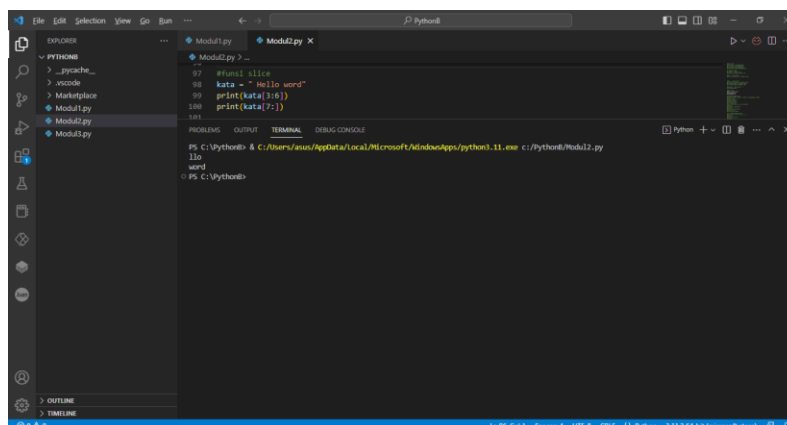
Terminal output:

```
PS C:\Python> & C:/Users/assu/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Python/Modul2.py
0
HELLO WORD
Hello word
['', 'Hello', 'word']
110
word
PS C:\Python>
```

d. Range Slice.

Untuk menampilkan range karakter dari a mendekati b (limit b), yang diformulasikan dengan.

Contoh program range slice:



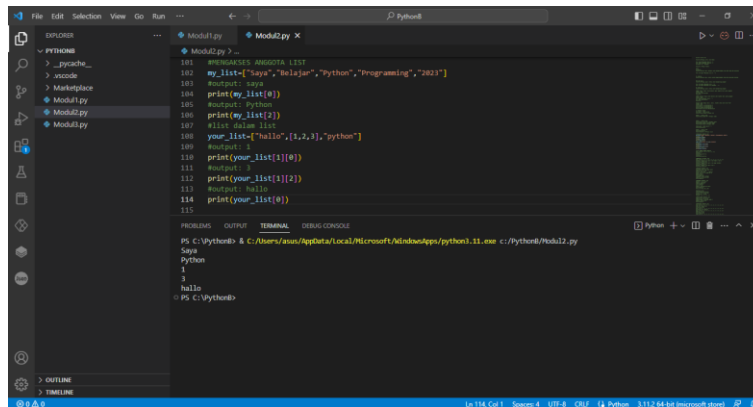
```
97 #fungsi slice
98 kata = "Hello word"
99 print(kata[3:6])
100 print(kata[7:])
101
102
```

Terminal output:

```
PS C:\Python> & C:/Users/assu/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Python/Modul2.py
110
word
PS C:\Python>
```

e. List

Salah satu tipe data built-in Python, yang dapat digunakan kapan saja tanpa harus meng-import modul terlebih dahulu. Contoh program mengakses anggota list. Kita bisa mengakses anggota list dengan menggunakan indeksnya dengan format `namalist[indeks]` :

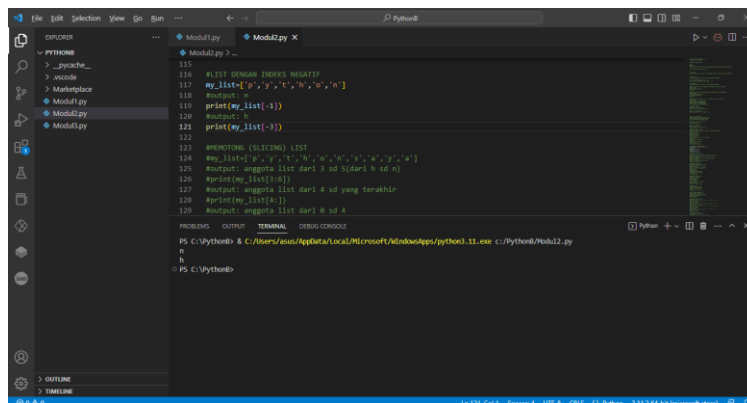


```
101 #MEMBUKSES ANGGOTA LIST
102 my_list=["Saya","Belajar","Python","Programming","2023"]
103 #Output: Saya
104 print(my_list[0])
105 #Output: Python
106 print(my_list[2])
107 #List dalam list
108 your_list=["hallo",[1,2,3],"python"]
109 #Output: 1
110 print(your_list[1][0])
111 #Output: 2
112 print(your_list[1][2])
113 #Output: hallo
114 print(your_list[0])
115
```

Terminal output:

```
PS C:\Python> & C:/Users/assu/AppData/Local/Microsoft/WindowsApps/python11.exe c:/Python/Modul2.py
Saya
Python
1
hallo
PS C:\Python>
```

List Dengan Indeks Negatif Python mendukung indeks negatif, yaitu urutan dimulai dari anggota terakhir. Indeks anggota paling belakang adalah -1, kemudian -2, dan seterusnya. Contoh list dengan index negatif :

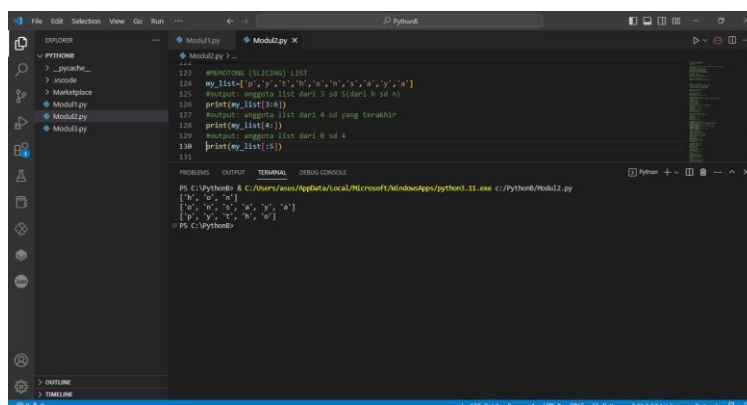


```
115
116 #LIST DENGAN INDEKS NEGATIF
117 my_list=['p','y','t','h','o','n']
118 #Output:
119 print(my_list[-1])
120 #Output: n
121 print(my_list[-3])
122
123 #MEMOTONG (SLICING) LIST
124 my_list=['p','y','t','h','o','n','a','s','a','s','a','s','a']
125 #Output: anggota list dari 3 sd 5(dari h sd n)
126 print(my_list[3:6])
127 #Output: anggota list dari 4 sd yang terakhir
128 print(my_list[4:])
129 #Output: anggota list dari 0 sd 4
```

Terminal output:

```
PS C:\Python> & C:/Users/assu/AppData/Local/Microsoft/WindowsApps/python11.exe c:/Python/Modul2.py
n
h
PS C:\Python>
```

Memotong (Slicing) List dengan mengakses anggota list dari range tertentu dengan menggunakan operator slicing titik dua (:). Contoh memotong list:

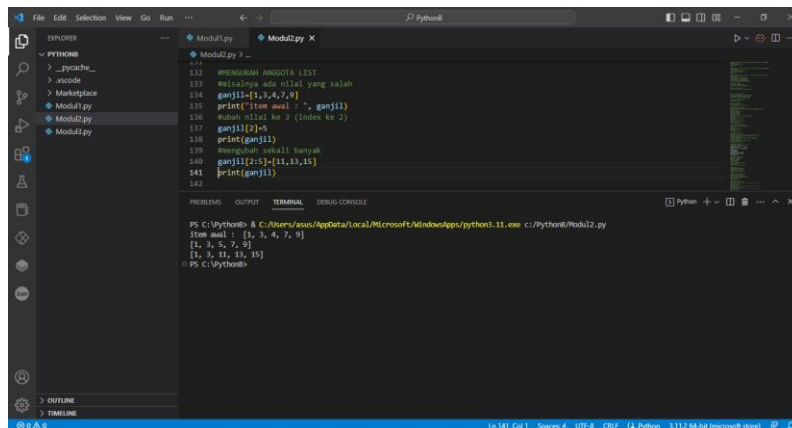


```
121
122 #MEMOTONG (SLICING) LIST
123 my_list=['p','y','t','h','o','n','a','s','a','s','a','s','a']
124 #Output: anggota list dari 3 sd 5(dari h sd n)
125 print(my_list[3:6])
126 #Output: anggota list dari 4 sd yang terakhir
127 print(my_list[4:])
128 #Output: anggota list dari 0 sd 4
129 print(my_list[:5])
130
```

Terminal output:

```
PS C:\Python> & C:/Users/assu/AppData/Local/Microsoft/WindowsApps/python11.exe c:/Python/Modul2.py
['h', 'o', 'n']
['p', 'y', 't', 'h', 'o', 'n', 'a', 's', 'a', 's', 'a', 's', 'a']
['p', 'y', 't', 'h', 'o', 'n']
PS C:\Python>
```

-Mengubah Anggota List merupakan tipe data yang bersifat mutable, artinya anggotanya bisa diubah. Contoh program mengubah anggota list :



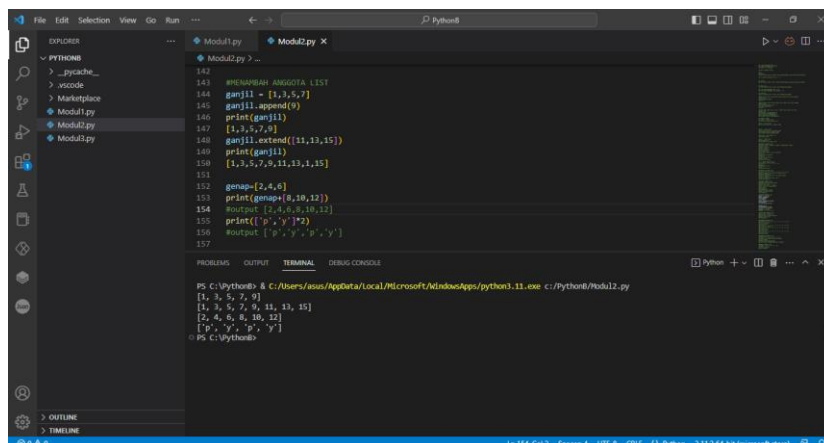
```
122
123 MENUBAH ANGOTA LIST
124 #misalnya ada nilai yang salah
125 ganjil=[1,3,4,7,9]
126 print("tipe awal : ", ganjil)
127 #ubah nilai ke 3 (index ke 2)
128 ganjil[2]=5
129 print(ganjil)
130 #mengubah keseluruhan banyak
131 ganjil[2:5]=[11,13,15]
132 print(ganjil)
```

Terminal output:

```
PS C:\Python> & C:\Users\asus\AppData\Local\Microsoft\WindowsApps\python3.11.exe c:/Python/Modul2.py
tipe awal : [1, 3, 4, 7, 9]
[1, 3, 5, 7, 9]
[1, 3, 11, 13, 15]
PS C:\Python>
```

-Menambahkan Anggota List

Fungsi `append()` berguna untuk menambahkan anggota ke dalam list. Selain itu, ada metode `extend()` untuk menambahkan anggota list ke dalam list. Kita juga bisa menggunakan operator `+` untuk menggabungkan dua list, dan operator `*` untuk melipatgandakan list. Contoh program Menambah Anggota List :

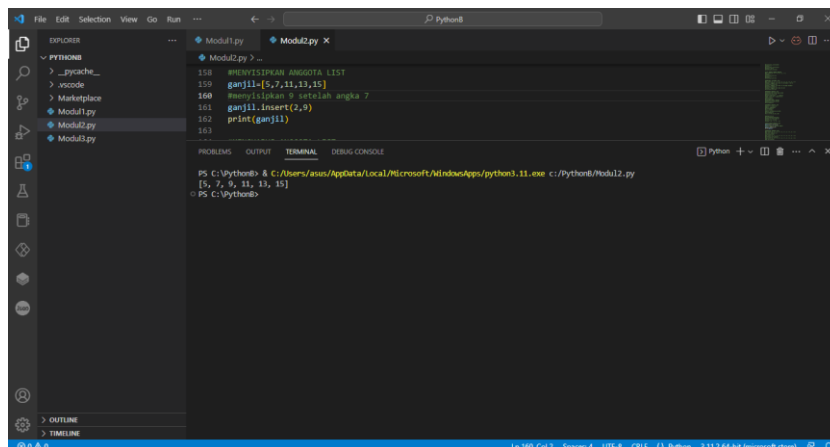


```
142
143 MENAMBAH ANGOTA LIST
144 ganjil = [1,3,5,7]
145 ganjil.append(9)
146 print(ganjil)
147 [1,3,5,7,9]
148 ganjil.extend([11,13,15])
149 print(ganjil)
150 [1,3,5,7,9,11,13,15]
151
152 genap=[2,4,6]
153 print(genap+[8,10,12])
154 #output [2,4,6,8,10,12]
155 print('p','y','y')
156 #output ['p','y','p','y']
157
```

Terminal output:

```
PS C:\Python> & C:\Users\asus\AppData\Local\Microsoft\WindowsApps\python3.11.exe c:/Python/Modul2.py
[1, 3, 5, 7, 9]
[1, 3, 5, 7, 9, 11, 13, 15]
[2, 4, 6, 8, 10, 12]
['p', 'y', 'p', 'y']
PS C:\Python>
```

-Contoh program menyisipkan anggota lis:



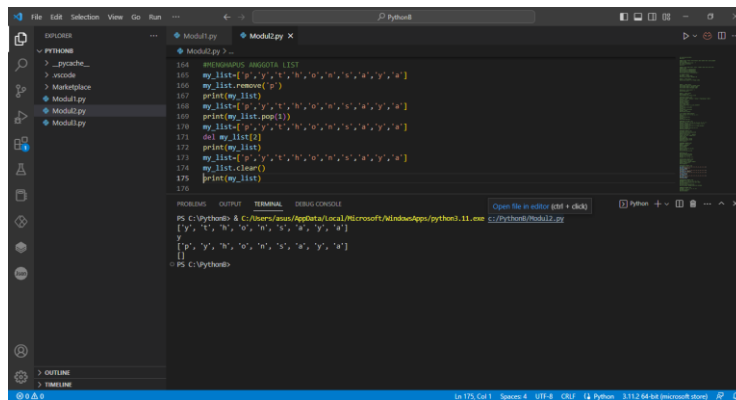
```
158
159 MENYISIPKAN ANGOTA LIST
160 ganjil=[5,7,11,13,15]
161 #menyisipkan 9 setelah angka 7
162 ganjil.insert(2,9)
163 print(ganjil)
```

Terminal output:

```
PS C:\Python> & C:\Users\asus\AppData\Local\Microsoft\WindowsApps\python3.11.exe c:/Python/Modul2.py
[5, 7, 9, 11, 13, 15]
PS C:\Python>
```

-Menghapus Anggota List

Dengan menggunakan metode `remove()`, `pop()`, atau kata kunci `del` untuk menghapus anggota list. Selain itu kita bisa menggunakan `clear()` untuk mengosongkan list. Fungsi `pop()` selain menghapus anggota list, juga mengembalikan nilai indeks anggota tersebut. Contoh program menghapus anggota list :



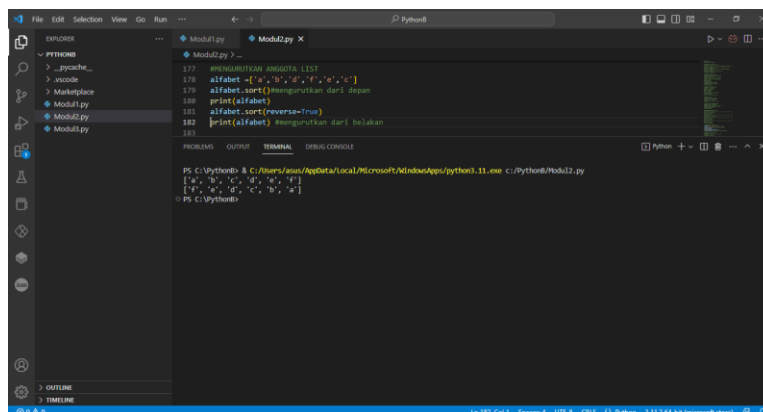
```
164 #menghapus anggota list
165 my_list=['p','y','t','h','o','n',' ','a','l','f','a','b','e','t']
166 my_list.remove('p')
167 print(my_list)
168 my_list=['p','y','t','h','o','n',' ','a','l','f','a','b','e','t']
169 print(my_list.pop())
170 my_list=['p','y','t','h','o','n',' ','a','l','f','a','b','e','t']
171 del my_list[2]
172 print(my_list)
173 my_list=['p','y','t','h','o','n',' ','a','l','f','a','b','e','t']
174 my_list.clear()
175 print(my_list)
176
```

OUTPUT:

```
['y', 't', 'h', 'o', 'n', ' ', 'a', 'l', 'f', 'a', 'b', 'e', 't']
['p', 'y', 't', 'h', 'o', 'n', ' ', 'a', 'l', 'f', 'a', 'b', 'e', 't']
['p', 'y', 't', 'h', 'o', 'n', ' ', 'a', 'l', 'f', 'a', 'b', 'e', 't']
[]
```

-Mengurutkan Anggota List

Untuk mengurutkan atau menyortir anggota list, kita bisa menggunakan metode `sort()`. Untuk membalik dengan urutan sebaliknya bisa dengan menggunakan argumen. Contoh program mengurutkan anggota list :



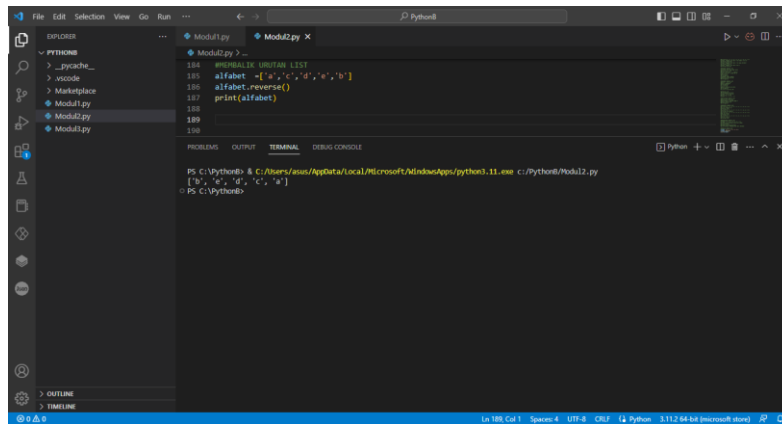
```
177 #MENGURUTKAN ANGGOTA LIST
178 alfabet = ['a','b','d','f','e','c']
179 alfabet.sort()#mengurutkan dari depan
180 print(alfabet)
181 alfabet.sort(reverse=True)
182 print(alfabet)#mengurutkan dari belakang
183
```

OUTPUT:

```
['a', 'b', 'c', 'd', 'e', 'f']
['f', 'e', 'd', 'c', 'b', 'a']
```

-Membalik Urutan List

Selain mengurutkan, kita juga bisa membalikkan urutan list dengan menggunakan metode `reverse()`. Contoh program membalik urutan list:



```
184 def membalikUrutan(L1ST):
185     alfabet = ['a','c','d','e','b']
186     alfabet.reverse()
187     print(alfabet)
188
189
190
```

Terminal Output:

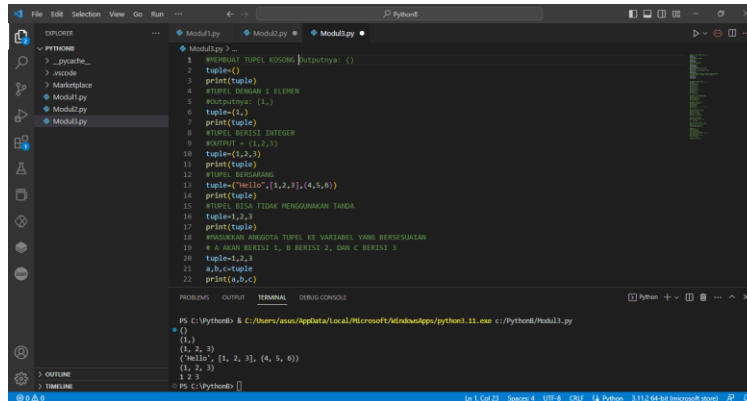
```
PS C:\Python> & C:\Users\asus\AppData\Local\Microsoft\WindowsApps\python3.11.exe c:\Python\Modul2.py
['b', 'e', 'd', 'c', 'a']
PS C:\Python>
```

MODUL 3

1. Membuat Tuple

Tuple mirip dengan list dibuat dengan meletakkan semua anggota di dalam tanda kurung (), masing-masing dipisahkan oleh tanda koma.

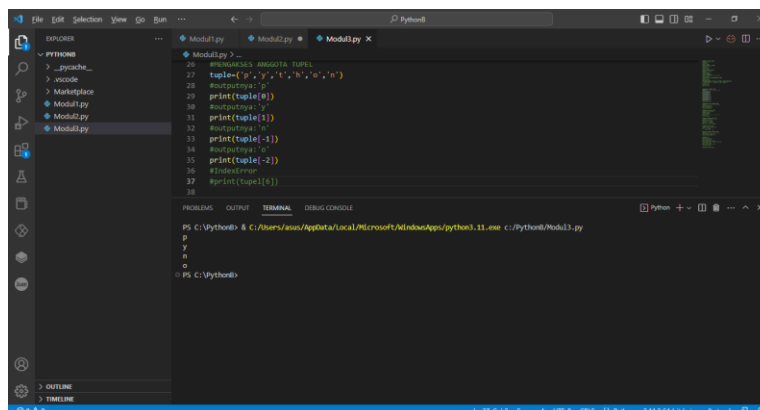
Contoh program membuat tuple :



```
1 #membuat tuple kosong
2 tuple=()
3 print(tuple)
4 #tuples dengan 1 elemen
5 #outputnya: (1,)
6 tuple=(1,)
7 print(tuple)
8 #tuple berisi integer
9 #output: (1,2,3)
10 tuple=(1,2,3)
11 print(tuple)
12 #tuple heterogen
13 tuple=("hello",[1,2,3],(4,5,6))
14 print(tuple)
15 #tuple bisa tidak menggunakan tanda
16 tuple=1,2,3
17 print(tuple)
18 #membuat anggota tuple ke variabel yang menggunakan
19 #A akan berisi 1, B berisi 2, dan C berisi 3
20 tuple=1,2,3
21 a,b,c=tuple
22 print(a,b,c)
```

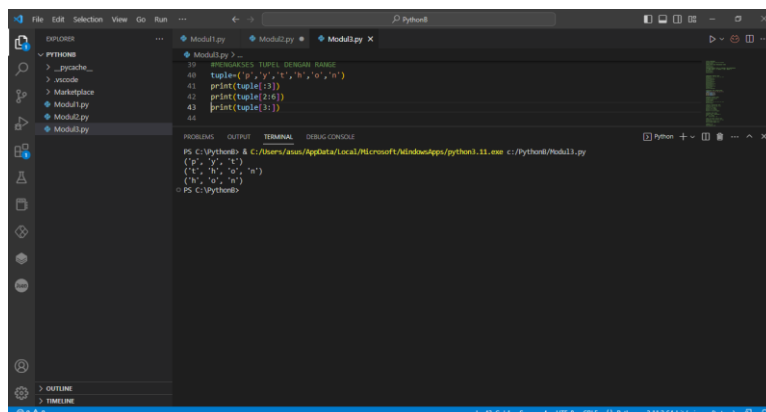
2. Mengakses Anggota Tuple

Seperti halnya list, tuple bisa mengakses anggota tuple lewat indeksnya menggunakan format `namatuple[indeks]`. Contoh program mengakses tuple :



```
1 #mengakses anggota tuple
2 tuple=('p','y','t','h','o','n')
3 #outputnya: p
4 print(tuple[0])
5 #outputnya: y
6 print(tuple[1])
7 #outputnya: n
8 print(tuple[-1])
9 #outputnya: o
10 print(tuple[-2])
11 #index error
12 #print(tuple[5])
13
```

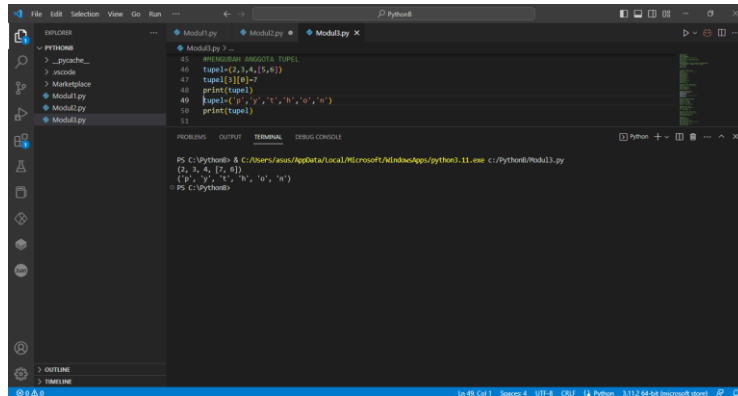
Sama seperti list, kita bisa mengakses satu range anggota tuple dengan menggunakan operator titik dua (:). Contoh program mengakses tuple dengan range :



```
1 #mengakses tuple dengan range
2 tuple=('p','y','t','h','o','n')
3 #outputnya: ('p','y','t')
4 print(tuple[0:3])
5 #outputnya: ('h','o','n')
6 print(tuple[3:])
```

3. Mengubah Anggota Tuple

Setelah tuple dibuat, maka anggota tuple tidak bisa lagi diubah atau dihapus. Akan tetapi, bila anggota tuple-nya adalah tuple bersarang dengan anggota seperti list, maka item pada list tersebut dapat diubah. Contoh program mengubah anggota tuple :



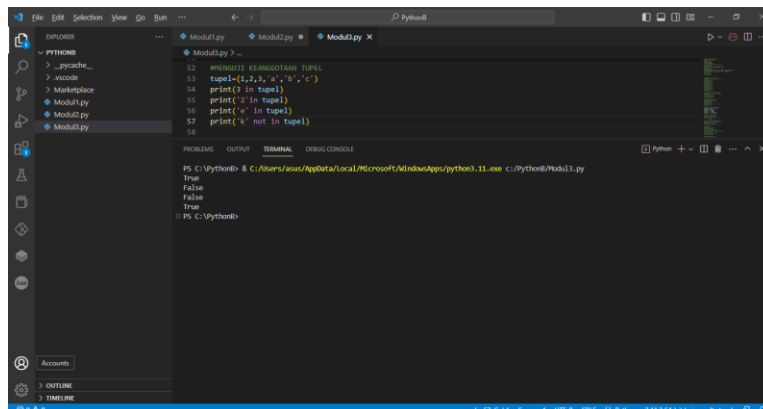
```
45 # Mendefinisikan anggota tuple
46 tuple1=(2,3,4,[5,6])
47 tuple1[3][0]=7
48 print(tuple1)
49 tuple1=( 'p','y','t','h','o','n')
50 print(tuple1)
```

Output:

```
PS C:\Python> python c:/Python/Modul3.py
(2, 3, 4, [7, 6])
('p', 'y', 't', 'h', 'o', 'n')
```

4. Menguji Keanggotaan Tuple

Seperti halnya string dan list, kita bisa menguji apakah sebuah objek adalah anggota dari tuple atau tidak, yaitu dengan menggunakan operator in atau not in untuk kebalikannya. Contoh program menguji anggota tuple :



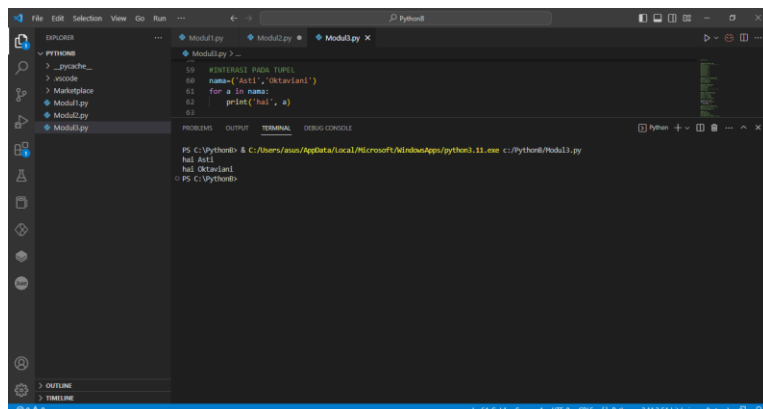
```
52 # Menguji keanggotaan tuple
53 tuple1=(1,2,3,'a','b','c')
54 print(1 in tuple1)
55 print(2 in tuple1)
56 print('a' in tuple1)
57 print('x' not in tuple1)
```

Output:

```
PS C:\Python> python c:/Python/Modul3.py
True
False
True
True
```

5. Iterasi pada Tuple

Kita bisa menggunakan for untuk melakukan iterasi pada tiap anggota dalam tuple. Contoh program iterasi pada tuple :



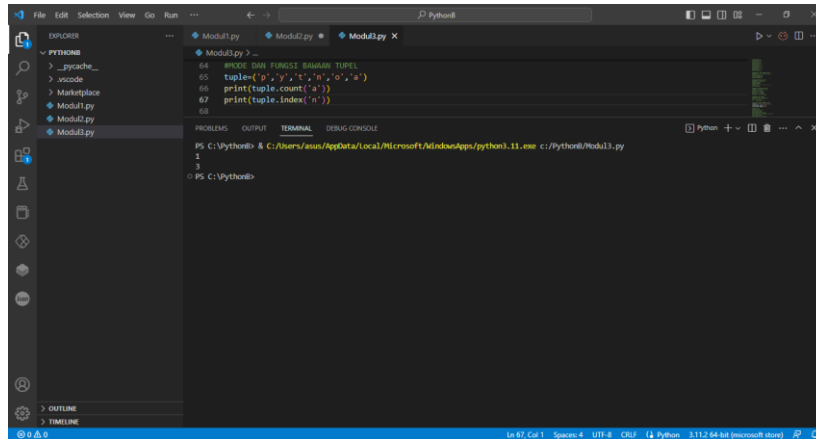
```
58 # Iterasi pada tuple
59 nama=('Asti','Octaviani')
60 for a in nama:
61     print(hai, a)
```

Output:

```
PS C:\Python> python c:/Python/Modul3.py
hai Asti
hai Octaviani
```

6. Metode dan Fungsi Bawaan Tuple

Tuple hanya memiliki dua buah metode yaitu `count()` dan `index()`. Metode `count(x)` berfungsi mengembalikan jumlah item yang sesuai dengan `x` pada tuple. Metode `index(x)` berfungsi mengembalikan indeks dari item pertama yang sama dengan `x`. Contoh program fungsi bawaan tuple :



```
44 #MENCARI DAN FUNGSI BAWAAN TUPLE
45 tuple=('p','y','t','h','o','n','a')
46 print(tuple.count('a'))
47 print(tuple.index('n'))
48
```

Terminal output:

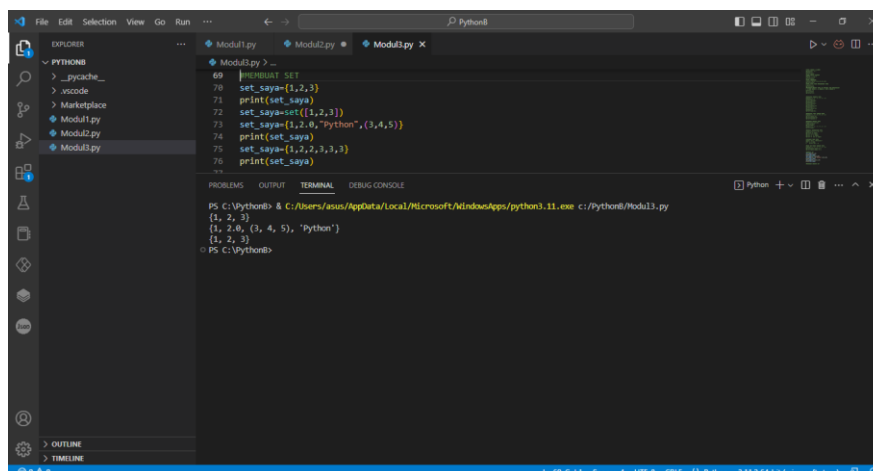
```
PS C:\Python> & C:/Users/assu/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Python/Modul3.py
1
3
PS C:\Python>
```

Tipe Data Set

Set adalah salah satu tipe data yang tidak berurut (unordered). Set memiliki anggota yang unik (tidak ada duplikasi).

1. Membuat Set

Set dibuat dengan meletakkan anggota - anggotanya di dalam tanda kurung kurawal `{ }`, dipisahkan menggunakan tanda koma. Kita juga bisa membuat set dari list dengan memasukkan list ke dalam fungsi `set()`. Set bisa berisi data campuran, baik integer, float, string, dan lain sebagainya. Akan tetapi set tidak bisa berisi list, set, dan dictionary. Contoh program membuat set :



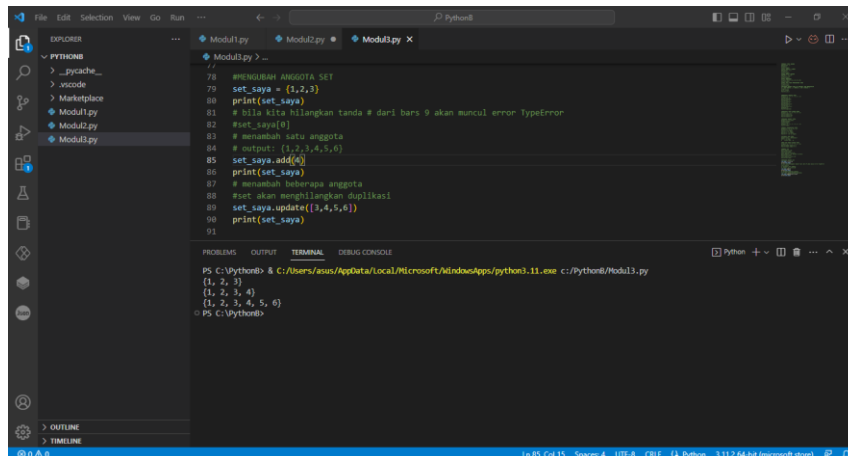
```
69 #Membuat SET
70 set_saya={1,2,3}
71 print(set_saya)
72 set_saya=set([1,2,3])
73 set_saya={1,2,0,'python',(1,4,5)}
74 print(set_saya)
75 set_saya={1,2,2,3,3,3}
76 print(set_saya)
77
```

Terminal output:

```
PS C:\Python> & C:/Users/assu/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Python/Modul3.py
{1, 2, 3}
{1, 2, 0, (1, 4, 5), 'python'}
{1, 2, 3}
```


2. Mengubah Anggota Set

Set bersifat mutable. Tapi, karena set adalah tipe data tidak berurut (unordered), maka kita tidak bisa menggunakan indeks. Set tidak mendukung indeks ataupun slicing. Untuk menambah satu anggota ke dalam set, kita bisa menggunakan fungsi `add()`, dan untuk menambahkan beberapa anggota sekaligus kita bisa menggunakan fungsi `update()`. List, tuple, maupun string bisa digunakan sebagai masukan dari fungsi `update()`. Contoh program Mengubah Anggota Set :



```
78 #MENUBAH ANGGOTA SET
79 set_saya = {1,2,3}
80 print(set_saya)
81 # bila kita hilangkan tanda # dari baris 9 akan muncul error TypeError
82 #set_saya[0]
83 # menambah satu anggota
84 # output: {1, 2, 3, 4, 5, 6}
85 set_saya.add(4)
86 print(set_saya)
87 # menambah beberapa anggota
88 #set akan menghilangkan duplikasi
89 set_saya.update([3,4,5,6])
90 print(set_saya)
91
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

PS C:\Python> & C:/Users/asus/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Python/Modul3.py

{1, 2, 3}

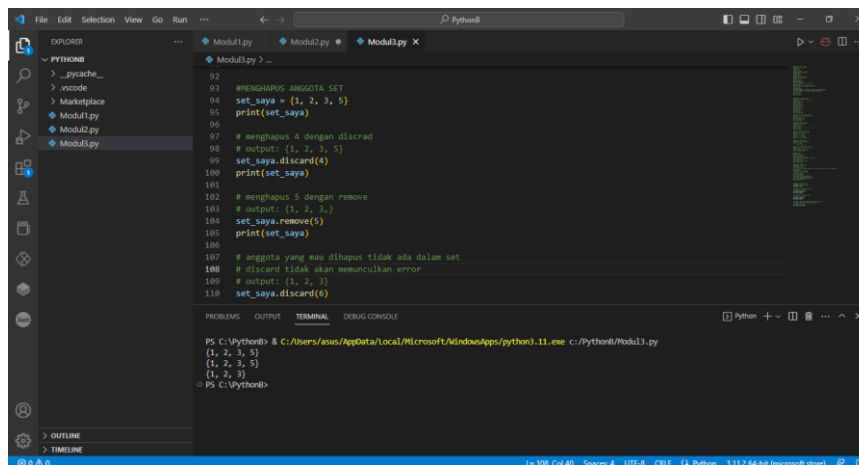
{1, 2, 3, 4}

{1, 2, 3, 4, 5, 6}

PS C:\Python>

3. Menghapus Anggota Set

Kita bisa menghapus anggota set dengan menggunakan fungsi `discard()` dan `remove()`. Contoh program Menghapus Anggota Set



```
92
93 #MENGGHAPUS ANGGOTA SET
94 set_saya = {1, 2, 3, 5}
95 print(set_saya)
96
97 # menghapus 4 dengan discard
98 # output: {1, 2, 3, 5}
99 set_saya.discard(4)
100 print(set_saya)
101
102 # menghapus 5 dengan remove
103 # output: {1, 2, 3}
104 set_saya.remove(5)
105 print(set_saya)
106
107 # anggota yang mau dihapus tidak ada dalam set
108 # discard tidak akan memunculkan error
109 # output: {1, 2, 3}
110 set_saya.discard(6)

```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

PS C:\Python> & C:/Users/asus/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Python/Modul3.py

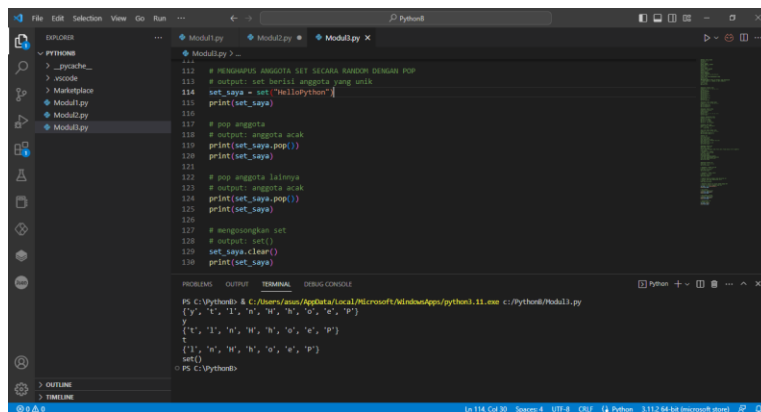
{1, 2, 3, 5}

{1, 2, 3, 5}

{1, 2, 3}

PS C:\Python>

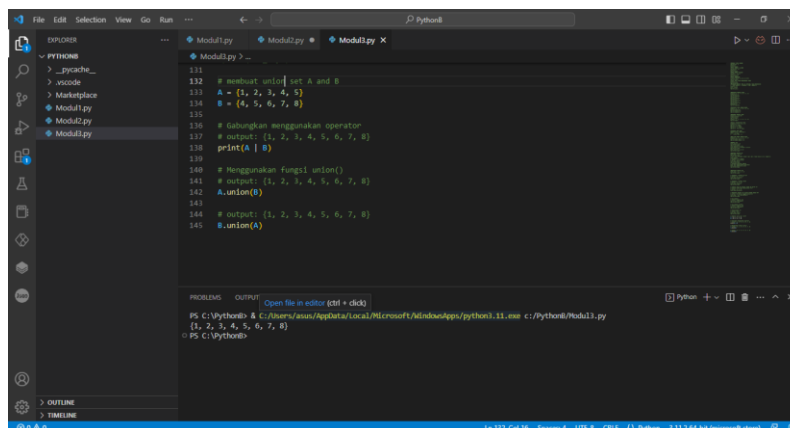
Menghapus Anggota Set Secara Random dengan pop(). Contoh program :



```
111 # Menghapus anggota set secara random dengan pop
112 # output: set berisi anggota yang unik
113 set_saya = set("HelloPython")
114 print(set_saya)
115
116 # pop anggota
117 # output: anggota acak
118 print(set_saya.pop())
119 print(set_saya)
120
121 # pop anggota lainnya
122 # output: anggota acak
123 print(set_saya.pop())
124 print(set_saya)
125
126 # menghapus set
127 # output: set()
128 set_saya.clear()
129 print(set_saya)
```

4. Operasi Gabungan (Union)

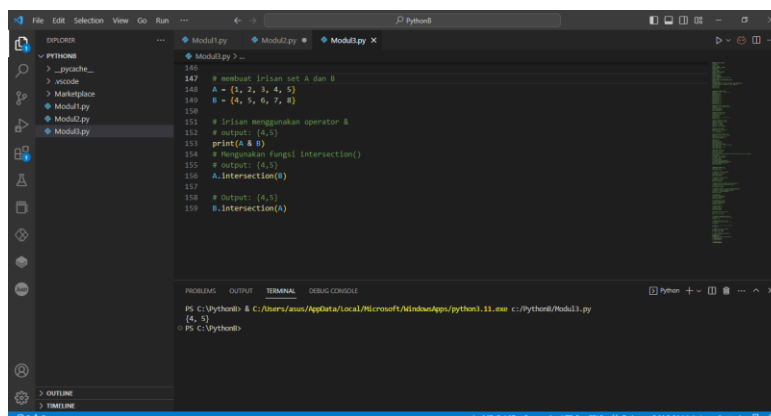
Operasi (Union) merupakan operasi gabungan dari suatu himpunan contoh himpunan A dan B. Gabungan dapat dibuat dengan menggunakan operator palang (|). Selain itu juga bisa dilakukan dengan menggunakan fungsi union(). Contoh program Operasi Gabungan (Union) dengan Set



```
111
112 # membuat union set A dan B
113 A = {1, 2, 3, 4, 5}
114 B = {4, 5, 6, 7, 8}
115
116 # Gabungan menggunakan operator
117 # output: {1, 2, 3, 4, 5, 6, 7, 8}
118 print(A | B)
119
120 # Menggunakan fungsi union()
121 # output: {1, 2, 3, 4, 5, 6, 7, 8}
122 A.union(B)
123
124 # output: {1, 2, 3, 4, 5, 6, 7, 8}
125 B.union(A)
```

5. Operasi Irisan (Intersection)

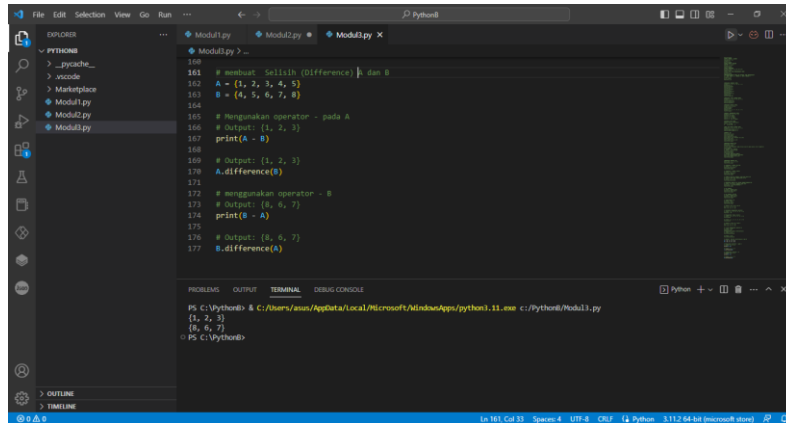
Operasi Irisan dilakukan dengan menggunakan operator jangkar (&). Irisan juga bisa dilakukan dengan menggunakan fungsi intersection(). Contoh program Operasi Irisan (Intersection) dengan Set :



```
146
147 # membuat irisan set A dan B
148 A = {1, 2, 3, 4, 5}
149 B = {4, 5, 6, 7, 8}
150
151 # irisan menggunakan operator &
152 # output: {4, 5}
153 print(A & B)
154
155 # Menggunakan fungsi intersection()
156 # output: {4, 5}
157 A.intersection(B)
158
159 # output: {4, 5}
160 B.intersection(A)
```

6. Operasi Selisih (Difference)

Operasi Selisih dilakukan dengan menggunakan operator kurang (-). Bisa juga dengan menggunakan fungsi `difference()`. Contoh program Operasi Selisih (Difference) dengan Set:



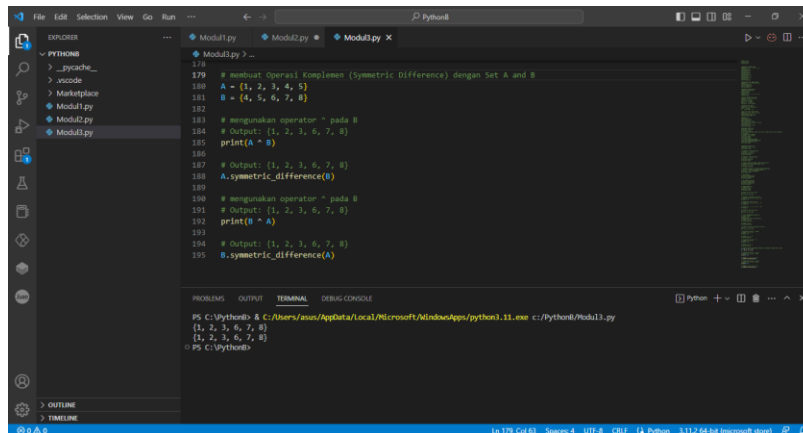
```
150 # membuat Set A dan B
151 A = {1, 2, 3, 4, 5}
152 B = {4, 5, 6, 7, 8}
153
154 # menggunakan operator - pada A
155 # Output: {1, 2, 3}
156 print(A - B)
157
158 # menggunakan operator - pada B
159 # Output: {6, 7, 8}
160 print(B - A)
161
162 # menggunakan fungsi difference()
163 # Output: {1, 2, 3}
164 A.difference(B)
165
166 # menggunakan fungsi difference()
167 # Output: {6, 7, 8}
168 B.difference(A)
```

Terminal Output:

```
PS C:\Python> python3.11.exe c:/Python/Modul3.py
{1, 2, 3}
{6, 7, 8}
```

7. Operasi Komplemen (Symmetric Difference)

Operasi komplemen dilakukan dengan menggunakan operator ^. Bisa juga dengan menggunakan fungsi `symmetric_difference()`. Contoh program Operasi Komplemen (Symmetric Difference) dengan Set



```
178 # membuat Operasi Komplemen (Symmetric Difference) dengan Set A dan B
179 A = {1, 2, 3, 4, 5}
180 B = {4, 5, 6, 7, 8}
181
182 # menggunakan operator ^ pada A
183 # Output: {1, 2, 3, 6, 7, 8}
184 print(A ^ B)
185
186 # menggunakan operator ^ pada B
187 # Output: {1, 2, 3, 6, 7, 8}
188 A.symmetric_difference(B)
189
190 # menggunakan operator ^ pada A
191 # Output: {1, 2, 3, 6, 7, 8}
192 print(B ^ A)
193
194 # menggunakan operator ^ pada B
195 B.symmetric_difference(A)
```

Terminal Output:

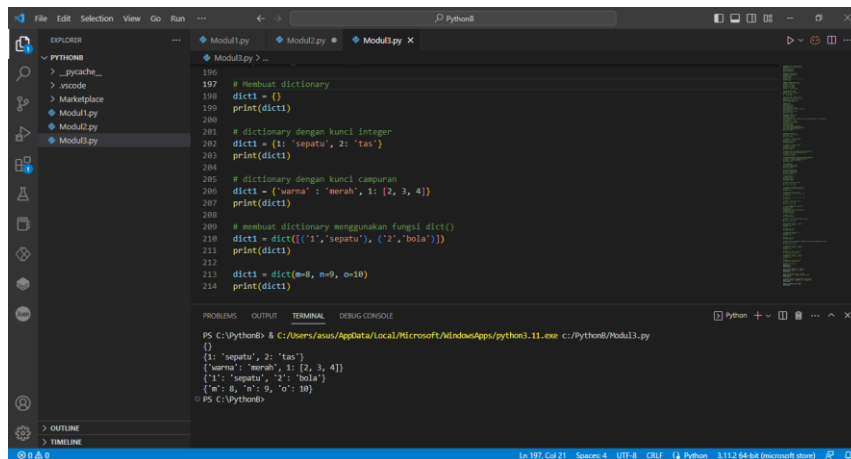
```
PS C:\Python> python3.11.exe c:/Python/Modul3.py
{1, 2, 3, 6, 7, 8}
{1, 2, 3, 6, 7, 8}
```

Dictionary

Tipe data yang anggotanya terdiri dari pasangan kunci:nilai (key:value). Dictionary bersifat tidak berurut (unordered) sehingga anggotanya tidak memiliki indeks.

1. Membuat Dictionary

Dictionary dibuat dengan menempatkan anggotanya di dalam tanda kurung kurawal {}, dipisahkan oleh tanda koma. Anggota dictionary terdiri dari pasangan kunci:nilai. Kunci harus bersifat unik, tidak boleh ada dua kunci yang sama dalam dictionary. Contoh program Membuat Dictionary :

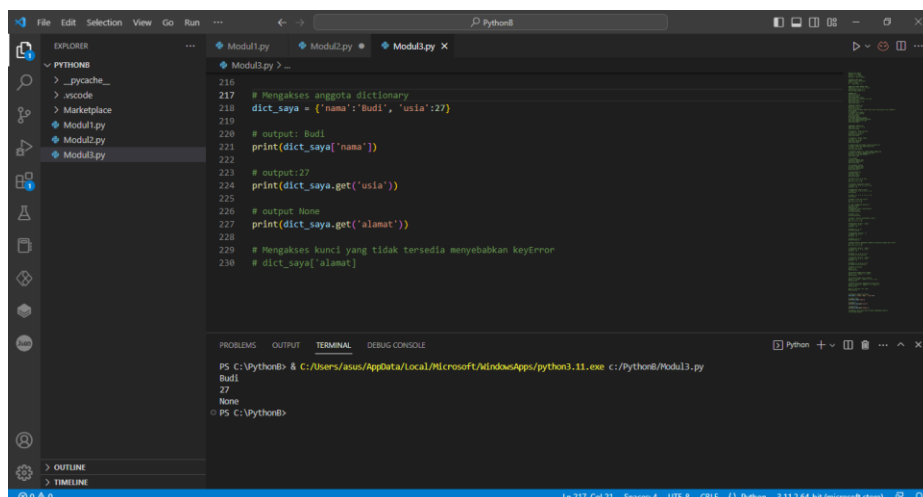


```
196
197 # Membuat dictionary
198 dict1 = {}
199 print(dict1)
200
201 # dictionary dengan kunci integer
202 dict1 = {1: 'sepatu', 2: 'tas'}
203 print(dict1)
204
205 # dictionary dengan kunci campuran
206 dict1 = {'warna': 'merah', 1: [2, 3, 4]}
207 print(dict1)
208
209 # membuat dictionary menggunakan fungsi dict()
210 dict1 = dict([('1', 'sepatu'), ('2', 'bola')])
211 print(dict1)
212
213 dict1 = dict(m=8, n=9, o=10)
214 print(dict1)
```

```
PS C:\Python> & C:/Users/asus/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Python/Modul3.py
{}
{1: 'sepatu', 2: 'tas'}
{'warna': 'merah', 1: [2, 3, 4]}
{'1': 'sepatu', '2': 'bola'}
{'m': 8, 'n': 9, 'o': 10}
```

2. Mengakses Anggota Dictionary

Dictionary tidak menggunakan indeks. Anggota dictionary diakses dengan menggunakan kuncinya. bisa juga diakses dengan menggunakan fungsi `get()`. Dengan menggunakan fungsi `get()`, bila kunci tidak ada di dalam dictionary, maka akan dikembalikan `None`. Bila tidak menggunakan fungsi `get()`, maka akan terjadi error `KeyError` bila kunci yang ingin diakses tidak ada di dalam dictionary. Contoh program Mengakses Anggota

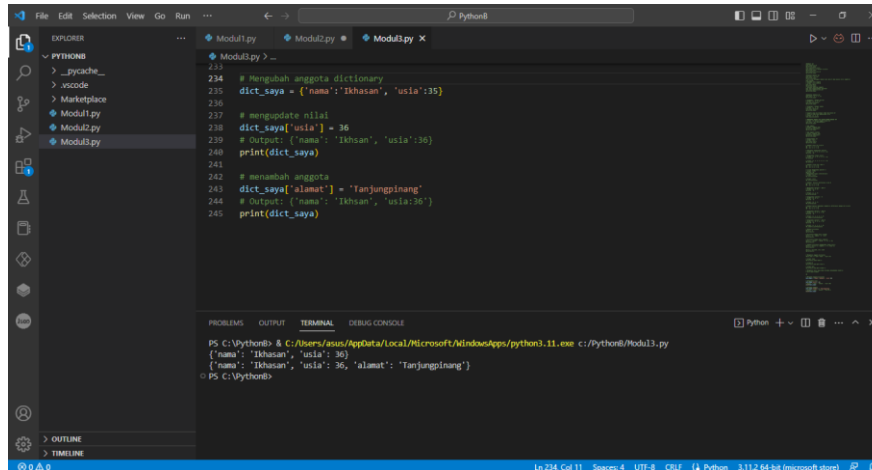


```
216
217 # Mengakses anggota dictionary
218 dict_saya = {'nama': 'Budi', 'usia': 27}
219
220 # output: Budi
221 print(dict_saya['nama'])
222
223 # output: 27
224 print(dict_saya.get('usia'))
225
226 # output: None
227 print(dict_saya.get('alamat'))
228
229 # Mengakses kunci yang tidak tersedia menyebabkan KeyError
230 # dict_saya['alamat']
```

```
PS C:\Python> & C:/Users/asus/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Python/Modul3.py
Budi
27
None
```

3. Mengubah Anggota Dictionary

Dictionary bersifat mutable. Kita bisa menambahkan atau mengubah nilai dari anggotanya menggunakan operator penugasan. Contoh program Mengubah Anggota Dictionary

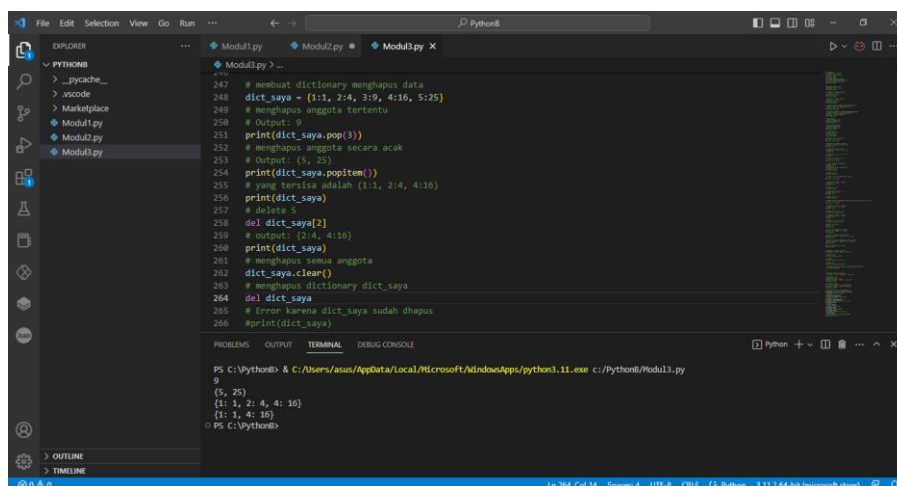


```
233
234 # Mengubah anggota dictionary
235 dict_saya = {'nama': 'Ikhwan', 'usia': 35}
236
237 # mengupdate nilai
238 dict_saya['usia'] = 36
239 # Output: {'nama': 'Ikhwan', 'usia': 36}
240 print(dict_saya)
241
242 # menambah anggota
243 dict_saya['alamat'] = 'Tanjungpinang'
244 # Output: {'nama': 'Ikhwan', 'usia': 36, 'alamat': 'Tanjungpinang'}
245 print(dict_saya)
```

PS C:\Python> & C:\Users\asus\AppData\Local\Microsoft\WindowsApps\python3.11.exe c:/Python/Modul3.py
{'nama': 'Ikhwan', 'usia': 36}
{'nama': 'Ikhwan', 'usia': 36, 'alamat': 'Tanjungpinang'}
PS C:\Python>

4. Menghapus Anggota Dictionary

Kita bisa menghapus anggota tertentu pada dictionary dengan menggunakan fungsi pop(). Fungsi ini menghapus anggota dengan mengembalikan kunci dari anggota tersebut. Fungsi lain, popitem() digunakan untuk menghapus anggota acak dari dictionary. Untuk menghapus semua anggota dictionary, bisa menggunakan fungsi clear(). Selain itu kita juga bisa menggunakan kata kunci del untuk menghapus anggota tertentu atau menghapus dictionary itu sendiri. Contoh program Menghapus Anggota Dictionary



```
247 # membuat dictionary menghapus data
248 dict_saya = {1:1, 2:4, 3:9, 4:16, 5:25}
249 # menghapus anggota tertentu
250 # Output: 9
251 print(dict_saya.pop(3))
252 # menghapus anggota secara acak
253 # Output: (5, 25)
254 print(dict_saya.popitem())
255 # yang terhapus adalah (1:1, 2:4, 4:16)
256 print(dict_saya)
257 # delete 5
258 del dict_saya[2]
259 # output: {2:4, 4:16}
260 print(dict_saya)
261 # menghapus semua anggota
262 dict_saya.clear()
263 # menghapus dictionary dict_saya
264 del dict_saya
265 # Error karena dict_saya sudah dihapus
266 #print(dict_saya)
```

PS C:\Python> & C:\Users\asus\AppData\Local\Microsoft\WindowsApps\python3.11.exe c:/Python/Modul3.py
9
(5, 25)
{1: 1, 2: 4, 4: 16}
{1: 1, 4: 16}
PS C:\Python>