



Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	71230977
Nama Lengkap	MICHAEL HOSEA
Minggu ke / Materi	04 / MODULAR PROGRAMMING

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

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

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

BAGIAN 1: MATERI MINGGU INI (40%)

Pada bagian ini, tuliskan kembali semua materi yang telah anda pelajari minggu ini. Sesuaikan penjelasan anda dengan urutan materi yang telah diberikan di saat praktikum. Penjelasan anda harus dilengkapi dengan contoh, gambar/ilustrasi, contoh program (source code) dan outputnya. Idealnya sekitar 5-6 halaman.

MATERI 1 : Fungsi, Argument, dan Parameter

```
nama = input("Masukkan nama Anda: ")  
print("Halo,", nama, "Selamat Datang!")
```

Code pemrograman yang berada diatas meminta pengguna untuk memasukkan nama, kemudian program akan menyapa nama yang dimasukkan oleh pengguna. Ada dua fungsi yang digunakan dalam program tersebut, yaitu input() untuk membaca input dari pengguna dan print() untuk menampilkan pesan di layar.

Fungsi pada dasarnya adalah kumpulan perintah yang memiliki tujuan tertentu dan dapat digunakan kembali. Ketika membuat program yang kompleks, kita perlu mengelompokkan kode ke dalam bagian-bagian yang lebih kecil yang disebut sebagai modul. Dengan cara ini, program kita menjadi lebih mudah dipahami dan dikelola. Ada dua jenis fungsi berdasarkan asalnya: fungsi bawaan yang sudah ada dalam Python dan fungsi yang dibuat oleh programmer sendiri.

Sebagai contoh, kita punya fungsi tambah() yang bertujuan untuk menjumlahkan dua bilangan. Fungsi ini memiliki beberapa komponen, seperti keyword def untuk mendefinisikan fungsi, nama fungsi (tambah()), dan isi fungsi yang ditulis dengan menjorok ke dalam. Fungsi tambah() membutuhkan dua argumen atau parameter, yaitu a dan b. Hasil penjumlahan dikembalikan menggunakan keyword return.

```
def tambah(a, b):  
  
    hasil = a + b  
    return hasil
```

Dalam kode program, kita bisa melihat penggunaan fungsi tersebut untuk menjumlahkan dua bilangan.

Fungsi tambah memiliki beberapa hal yang perlu diperhatikan:

- Keyword def digunakan untuk mendefinisikan sebuah fungsi.
- Nama fungsi yang digunakan adalah tambah().
- Isi dari fungsi harus ditulis dengan indentasi satu tab. Bagian tambah(a, b): merupakan penanda blok.
- Fungsi tambah() membutuhkan dua argumen, yang nantinya akan dikenali sebagai parameter a dan b.
- Fungsi tersebut menghasilkan hasil penjumlahan yang dapat disimpan di sebuah variabel. Keyword return digunakan untuk mengembalikan nilai dari fungsi.

```
def tambah(a, b):  
  
    hasil = a + b  
    return hasil  
  
c = tambah(3, 4)  
print(c)
```

Jalannya programnya seperti ini :

1. Fungsi `tambah(a, b)` didefinisikan dengan parameter `a` dan `b`.

2. Di dalam fungsi, dilakukan penjumlahan antara `a` dan `b`, hasilnya disimpan dalam variabel `hasil`.
3. Nilai `hasil` dikembalikan menggunakan keyword `return`.
4. Program memanggil fungsi `tambah(3, 4)` dengan argumen 3 dan 4, hasilnya disimpan dalam variabel `c`.
5. Nilai dari variabel `c`, yang merupakan hasil penjumlahan 3 dan 4, ditampilkan di layar menggunakan fungsi `print()`.
6. Output yang ditampilkan adalah nilai dari variabel `c`, yaitu 7.

```

Week 4 > halo.py > ...
1 def tambah(a, b):
2
3     hasil = a + b
4     return hasil
5
6 c = tambah(3, 4)
7 print(c)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
(base) maik@Michaels-MacBook-Air Praktek Algoritma (Semester 2) % python -u "/Users/maik/Documents/Praktek Algoritma (Semester 2)/Week 4/halo.py"
7
(base) maik@Michaels-MacBook-Air Praktek Algoritma (Semester 2) %

```

Gambar 4.1 : Hasil dari pemanggilan fungsi tambah()

Anda dapat memanggil fungsi hanya setelah fungsi tersebut telah didefinisikan sebelumnya. Jika Anda mencoba memanggil fungsi sebelum definisinya, atau jika definisi fungsi tersebut berada di bawah pemanggilannya, program Anda akan mengalami kesalahan. Contohnya seperti code pada Gambar 4.2 dibawah ini.

```

Week 4 > halo.py > tambah
1 c = tambah(3, 4)
2
3 def tambah(a, b):
4
5     hasil = a + b
6     return hasil
7
8 print(c)

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS
(base) maik@Michaels-MacBook-Air Praktek Algoritma (Semester 2) % python -u "/Users/maik/Documents/Praktek Algoritma (Semester 2)/Week 4/halo.py"
Traceback (most recent call last):
  File "/Users/maik/Documents/Praktek Algoritma (Semester 2)/Week 4/halo.py", line 1, in <module>
    c = tambah(3, 4)
NameError: name 'tambah' is not defined
(base) maik@Michaels-MacBook-Air Praktek Algoritma (Semester 2) %

```

Gambar 4.2: Kesalahan karena melakukan pemanggilan sebelum didefinisikan

MATERI 2 : Return Value

Berdasarkan hasil yang dikeluarkan oleh fungsi, secara umum ada dua jenis:

1. Fungsi yang tidak mengembalikan nilai: sering disebut sebagai void function. Contohnya, fungsi `print_twice()` berikut ini:

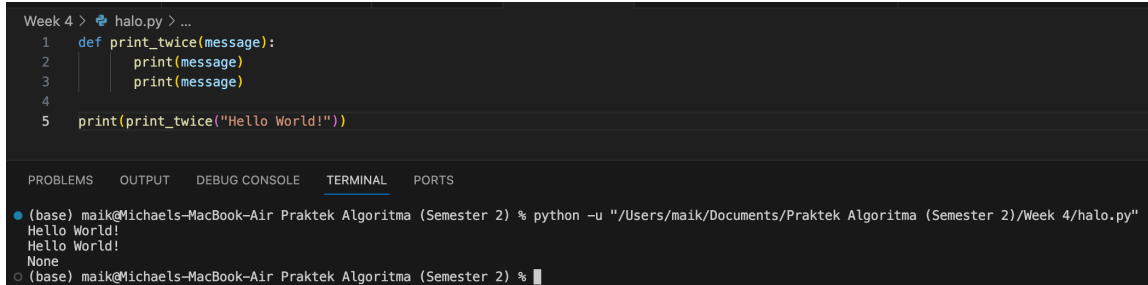
```

def print_twice(message):
    print(message)
    print(message)

```

```
print_twice("Hello World!")
```

Fungsi `print_twice()` membutuhkan satu parameter yaitu `message`. Kemudian fungsi ini akan menampilkan nilai dari variabel `message` sebanyak dua kali. Fungsi ini tidak menghasilkan nilai yang dapat digunakan untuk proses berikutnya. Jika kita mencoba memanggilnya dan mencetak hasilnya, maka akan dikeluarkan nilai `None`.



```
Week 4 > halo.py > ...
1 def print_twice(message):
2     print(message)
3     print(message)
4
5 print(print_twice("Hello World!"))

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

(base) maik@Michaels-MacBook-Air Praktek Algoritma (Semester 2) % python -u "/Users/maik/Documents/Praktek Algoritma (Semester 2)/Week 4/halo.py"
Hello World!
Hello World!
None
(base) maik@Michaels-MacBook-Air Praktek Algoritma (Semester 2) %
```

Gambar 4.3: Contoh penggunaan fungsi `print_twice` yang menghasilkan `None`

2. Fungsi yang mengembalikan nilai: Contohnya, fungsi `tambah()` berikut ini:

```
def tambah(a, b, c):
    hasil = a + b + c
    return hasil
```

Fungsi `tambah()` membutuhkan tiga parameter: `a`, `b`, dan `c`. Di dalam fungsi, dilakukan penjumlahan dari ketiga parameter tersebut dan hasilnya disimpan dalam variabel `hasil`. Kemudian variabel `hasil` tersebut dikembalikan dari fungsi menggunakan keyword `return`.

Keyword `return` digunakan untuk mengeluarkan nilai yang merupakan hasil dari fungsi, serta mengakhiri eksekusi fungsi. Fungsi `tambah()` ini bisa digunakan untuk mencari hasil penjumlahan tiga bilangan. Hasil dari penjumlahan tersebut dapat digunakan untuk proses berikutnya, seperti dalam contoh program untuk mencari rata-rata dari tiga bilangan.

```
def tambah(a, b, c):
    hasil = a + b + c
    return hasil

nilai1 = 70
nilai2 = 85
nilai3 = 55

rata_rata = tambah(nilai1, nilai2, nilai3)/3
print(rata_rata)
```

Berikut penjelasan mengenai setiap baris dalam kode tersebut:

1. `def tambah(a, b, c):`: Mendefinisikan fungsi `tambah()` dengan tiga parameter `a`, `b`, dan `c`.
2. `hasil = a + b + c`: Di dalam fungsi, variabel `hasil` diinisialisasi dengan penjumlahan dari ketiga parameter.

3. `return hasil`: Mengembalikan nilai `hasil` sebagai hasil dari fungsi `tambah()`.
4. `nilai1 = 70`: Mendefinisikan variabel `nilai1` dengan nilai 70.
5. `nilai2 = 85`: Mendefinisikan variabel `nilai2` dengan nilai 85.
6. `nilai3 = 55`: Mendefinisikan variabel `nilai3` dengan nilai 55.
7. `rata_rata = tambah(nilai1, nilai2, nilai3)/3`: Memanggil fungsi `tambah()` dengan tiga argumen `nilai1`, `nilai2`, dan `nilai3`, kemudian hasilnya dibagi dengan 3 untuk mencari rata-rata. Hasilnya disimpan dalam variabel `rata_rata`.
8. `print(rata_rata)`: Mencetak nilai `rata_rata` ke layar.

Optional Argument dan Named Argument

Fungsi dapat memiliki parameter opsional, yang artinya parameter tersebut bersifat opsional dan memiliki nilai default yang sudah ditentukan sebelumnya. Untuk mendefinisikan parameter opsional, Anda harus menentukan nilai defaultnya dalam definisi fungsi, seperti pada contoh berikut:

```
def hitung_belanja(belanja, diskon=0):  
    bayar = belanja - (belanja * diskon)/100  
    return bayar
```

Fungsi `hitung_belanja()` memiliki dua parameter, yaitu `belanja` dan `diskon`. Parameter `diskon` secara default memiliki nilai 0, yang berarti diskon 0%.

Untuk memanggil fungsi `hitung_belanja()`, Anda dapat melakukannya dengan beberapa cara seperti yang ditunjukkan di bawah ini:

```
print(hitung_belanja(100000))  
print(hitung_belanja(100000, 10))  
print(hitung_belanja(100000, 50))
```

Output yang dihasilkan adalah sebagai berikut:

```
100000.0  
90000.0  
50000.0
```

Anda dapat memanggil fungsi `cetak()` dengan menyebutkan nama parameter atau disebut juga sebagai named argument. Dengan menggunakan cara ini, urutan argument tidak harus sama dengan urutan parameter pada fungsi. Contoh:

```
def cetak(a, b, c):  
    print("Nilai a:", a)  
    print("Nilai b:", b)  
    print("Nilai c:", c)  
  
cetak(20, 30, 40)
```

Output :

Nilai a: 20

Nilai b: 30
Nilai c: 40

Anda juga dapat memanggil fungsi `cetak()` dengan cara berikut:

```
def cetak(a, b, c):  
    print("Nilai a:", a)  
    print("Nilai b:", b)  
    print("Nilai c:", c)  
  
cetak(a=20, b=30, c=40)
```

Pemanggilan fungsi `cetak()` dilakukan dengan menyebutkan nama argumentnya (named argument). Jika menggunakan cara ini, urutan argument tidak harus sama dengan urutan parameter pada fungsi. Contoh:

```
def cetak(a, b, c):  
    print("Nilai a:", a)  
    print("Nilai b:", b)  
    print("Nilai c:", c)  
  
cetak(b=30, c=40, a=20)
```

Ketiga contoh tersebut akan menghasilkan output yang sama yaitu:

Nilai a: 20
Nilai b: 30
Nilai c: 40

Materi 3: Anonymous Function (Lambda)

Anonymous function, seperti namanya, adalah fungsi tanpa nama yang didefinisikan secara langsung tanpa penggunaan kata kunci `def`. Pada Python, kita menggunakan kata kunci `lambda` untuk mendefinisikan anonymous function. Contoh penggunaan lambda function dalam kasus fungsi `tambah()` yang sudah didefinisikan sebelumnya adalah sebagai berikut:

```
def tambah(a, b):  
    hasil = a + b  
    return hasil  
  
print(tambah(10, 20))
```

Di sini, kita telah mendefinisikan fungsi `tambah()` menggunakan pendekatan konvensional dengan kata kunci `def`. Namun, jika kita ingin menggunakan lambda function, kita bisa menuliskannya tanpa perlu mendefinisikan nama fungsi seperti ini:

```
tambah = lambda a, b: a + b  
print(tambah(10, 20))
```

Dalam contoh kedua, kita mendefinisikan fungsi `tambah` menggunakan lambda function. Lambda function tersebut mengambil dua argumen, `a` dan `b`, dan langsung mengembalikan hasil penjumlahan dari kedua argumen tersebut.

Perlu diperhatikan bahwa lambda function lebih sering digunakan untuk kasus sederhana atau saat diperlukan dalam satu baris kode, sedangkan pendefinisian fungsi dengan menggunakan `def` lebih umum digunakan untuk fungsi yang lebih kompleks atau dengan beberapa baris kode.

Setiap anonymous function pada Python terdiri dari beberapa bagian berikut ini:

- Keyword: `lambda`
- Bound variable: parameter atau argumen pada lambda function
- Body: bagian utama dari lambda function, berisi ekspresi atau statement yang menghasilkan suatu nilai.

Jadi, secara umum, sebuah lambda function ditulis dalam format: `lambda <bound variable>: <body>`.

BAGIAN 2: LATIHAN MANDIRI (60%)

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

SOAL 1

Buatlah sebuah fungsi yang dapat menentukan apakah ketiga parameter memenuhi semua ketentuan berikut ini:

- Ketiga parameter tersebut nilainya berbeda semua.
- Ada kemungkinan jika diambil dua parameter dan dijumlahkan hasilnya sama dengan parameter lainnya (yang tersisa).

Fungsi tersebut akan menghasilkan nilai True jika semua ketentuan tersebut dipenuhi. Jika tidak terpenuhi maka fungsi akan menghasilkan nilai False. Fungsi anda harus diberi nama cek_angka().

- Source Code dan Output

```
def cek_angka():  
    a = int(input("Masukkan angka pertama: "))  
    b = int(input("Masukkan angka kedua: "))  
    c = int(input("Masukkan angka ketiga: "))  
  
    if a != b and a != c and b != c:  
        if a + b == c or a + c == b or b + c == a:  
            return True  
        return False  
  
print(cek_angka())
```



```
Week 4 > github-week4 > latihan-4.1.py > ...
1 def cek_angka():
2     a = int(input("Masukkan angka pertama: "))
3     b = int(input("Masukkan angka kedua: "))
4     c = int(input("Masukkan angka ketiga: "))
5
6     if a != b and a != c and b != c:
7         if a + b == c or a + c == b or b + c == a:
8             return True
9     return False
10
11 print(cek_angka())

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
• (base) maik@Michaels-MacBook-Air Praktek Algoritma (Semester 2) % python -u "/Users/maik/Documents/Praktek Algoritma (Semester 2)/Week 4/github-week4/latihan-4.1.py"
Masukkan angka pertama: 1
Masukkan angka kedua: 2
Masukkan angka ketiga: 3
True
• (base) maik@Michaels-MacBook-Air Praktek Algoritma (Semester 2) % python -u "/Users/maik/Documents/Praktek Algoritma (Semester 2)/Week 4/github-week4/latihan-4.1.py"
Masukkan angka pertama: 2
Masukkan angka kedua: 2
Masukkan angka ketiga: 3
True
• (base) maik@Michaels-MacBook-Air Praktek Algoritma (Semester 2) % python -u "/Users/maik/Documents/Praktek Algoritma (Semester 2)/Week 4/github-week4/latihan-4.1.py"
Masukkan angka pertama: 3
Masukkan angka kedua: 2
Masukkan angka ketiga: 1
True
• (base) maik@Michaels-MacBook-Air Praktek Algoritma (Semester 2) % python -u "/Users/maik/Documents/Praktek Algoritma (Semester 2)/Week 4/github-week4/latihan-4.1.py"
Masukkan angka pertama: 3
Masukkan angka kedua: 3
Masukkan angka ketiga: 2
True
• (base) maik@Michaels-MacBook-Air Praktek Algoritma (Semester 2) % python -u "/Users/maik/Documents/Praktek Algoritma (Semester 2)/Week 4/github-week4/latihan-4.1.py"
Masukkan angka pertama: 1
Masukkan angka kedua: 2
Masukkan angka ketiga: 3
True
• (base) maik@Michaels-MacBook-Air Praktek Algoritma (Semester 2) % python -u "/Users/maik/Documents/Praktek Algoritma (Semester 2)/Week 4/github-week4/latihan-4.1.py"
Masukkan angka pertama: 1
Masukkan angka kedua: 2
Masukkan angka ketiga: 4
False
```

Gambar 4.4: Source Code dan Output Latihan Mandiri 4.1

- Penjelasan

Berikut adalah penjelasan per baris dari kode yang diberikan:

1. `def cek_angka():`: Mendefinisikan fungsi bernama `cek_angka()` tanpa parameter.
2. `a = int(input("Masukkan angka pertama: "))`: Membaca input dari pengguna untuk angka pertama dan mengkonversi input tersebut ke tipe data integer. Nilai ini disimpan dalam variabel `a`.
3. `b = int(input("Masukkan angka kedua: "))`: Membaca input dari pengguna untuk angka kedua dan mengkonversi input tersebut ke tipe data integer. Nilai ini disimpan dalam variabel `b`.
4. `c = int(input("Masukkan angka ketiga: "))`: Membaca input dari pengguna untuk angka ketiga dan mengkonversi input tersebut ke tipe data integer. Nilai ini disimpan dalam variabel `c`.
5. `if a != b and a != c and b != c:`: Memeriksa apakah semua angka yang dimasukkan oleh pengguna tidak sama (berbeda satu sama lain). Jika kondisi ini terpenuhi, maka blok berikutnya dieksekusi.
6. `if a + b == c or a + c == b or b + c == a:`: Memeriksa apakah salah satu dari tiga kemungkinan penjumlahan dari dua angka sama dengan angka ketiga. Jika kondisi ini terpenuhi, maka fungsi mengembalikan `True`.
7. `return True`: Mengembalikan `True` jika kondisi di atas terpenuhi.
8. `return False`: Jika tidak ada satu pun dari kondisi di atas yang terpenuhi, maka fungsi mengembalikan `False`.
9. `print(cek_angka())`: Memanggil fungsi `cek_angka()` dan mencetak hasilnya.

SOAL 2

Buatlah sebuah fungsi yang dapat menentukan apakah minimal dua dari tiga parameter yang diberikan memiliki digit paling kanan yang sama. Fungsi tersebut menghasilkan nilai True jika memenuhi dan False jika tidak memenuhi. Gunakan fungsi tersebut untuk mengecek beberapa test-case berikut ini:

- Input = 30, 20, 18. Output yang diharapkan = True
- Input = 145, 5, 100. Output yang diharapkan = True
- Input = 71, 187, 18. Output yang diharapkan = False
- Input = 1024, 14, 94. Output yang diharapkan = True
- Input = 53, 8900, 658. Output yang diharapkan = False

Ketiga bilangan tersebut diinputkan oleh pengguna, sehingga anda perlu membaca input dari pengguna. Fungsi anda harus diberi nama `cek_digit_belakang()`.

- Source Code dan Output

```
def cek_digit_belakang(a, b, c):
    if str(a)[-1] == str(b)[-1] or str(a)[-1] == str(c)[-1] or str(b)[-1] == str(c)[-1]:
        return True
    else:
        return False

# Test-case
print(cek_digit_belakang(30, 20, 18))
print(cek_digit_belakang(145, 5, 100))
print(cek_digit_belakang(71, 187, 18))
print(cek_digit_belakang(1024, 14, 94))
print(cek_digit_belakang(53, 8900, 658))
```

```
Week 4 > github-week4 > latihan-4.2.py > ...
1 def cek_digit_belakang(a, b, c):
2     if str(a)[-1] == str(b)[-1] or str(a)[-1] == str(c)[-1] or str(b)[-1] == str(c)[-1]:
3         return True
4     else:
5         return False
6
7 # Test-case
8 print(cek_digit_belakang(30, 20, 18))
9 print(cek_digit_belakang(145, 5, 100))
10 print(cek_digit_belakang(71, 187, 18))
11 print(cek_digit_belakang(1024, 14, 94))
12 print(cek_digit_belakang(53, 8900, 658))

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
• (base) maik@Michaels-MacBook-Air Praktek Algoritma (Semester 2) % python -u "/Users/maik/Documents/Praktek Algoritma (Semester 2)/Week 4/github-week4/latihan-4.2.py"
True
True
False
True
False
(base) maik@Michaels-MacBook-Air Praktek Algoritma (Semester 2) %
```

Gambar 4.5: Source Code dan Output Latihan Mandiri 4.2

- Penjelasan

Berikut adalah penjelasan per baris dari kode yang diberikan:

1. ``def cek_digit_belakang(a, b, c):``: Mendefinisikan fungsi ``cek_digit_belakang()`` dengan tiga parameter: ``a``, ``b``, dan ``c``.
2. ``if str(a)[-1] == str(b)[-1] or str(a)[-1] == str(c)[-1] or str(b)[-1] == str(c)[-1]:``: Memeriksa apakah digit terakhir dari setiap angka sama satu sama lain. Angka-angka tersebut dikonversi menjadi string menggunakan fungsi ``str()`` untuk memungkinkan pengambilan digit terakhir dengan indeks ``-1``.
3. ``return True``: Mengembalikan ``True`` jika ada setidaknya dua digit terakhir yang sama.
4. ``else:``: Jika tidak ada dua digit terakhir yang sama,
5. ``return False``: Mengembalikan ``False``.
6. ``print(cek_digit_belakang(30, 20, 18))``: Memanggil fungsi ``cek_digit_belakang()`` dengan argumen (30, 20, 18) dan mencetak hasilnya.
7. Test-case: Berikutnya, kode menguji fungsi ``cek_digit_belakang()`` dengan beberapa kasus uji, yaitu:
 - (30, 20, 18)
 - (145, 5, 100)
 - (71, 187, 18)
 - (1024, 14, 94)
 - (53, 8900, 658)
8. Setiap hasil dari kasus uji dicetak menggunakan pernyataan ``print()``.

SOAL 3

Buatlah fungsi-fungsi konversi suhu menggunakan lambda function. Fungsi-fungsi yang harus anda implementasikan:

- Celcius to Fahrenheit. $F = (9/5) * C + 32$
- Celcius to Reamur. $R = 0.8 * C$

Berikan contoh penggunaannya untuk test-case berikut ini:

- Input C = 100. Output F = 212.
- Input C = 80. Output R = 64.
- Input = 0. Output F = 32.

- Source Code dan Output

```
# Fungsi konversi Celcius ke Fahrenheit
celcius_to_fahrenheit = lambda C: (9/5) * C + 32

# Fungsi konversi Celcius ke Reamur
celcius_to_reamur = lambda C: 0.8 * C

# Test-case 1
C = 100.0
F = celcius_to_fahrenheit(C)
print("Input C =", C, ". Output F =", F)

# Test-case 2
C = 80.0
```

```

R = celcius_to_reamur(C)
print("Input C =", C, ". Output R =", R)

# Test-case 3
C = 0.0
F = celcius_to_fahrenheit(C)
print("Input C =", C, ". Output F =", F)

```

The screenshot shows a code editor with a dark theme. The top part displays the source code for a Python script named 'latihan-4.3.py'. The code defines two lambda functions: 'celcius_to_fahrenheit' which takes a Celsius value 'C' and returns its equivalent in Fahrenheit using the formula $F = \frac{9}{5}C + 32$, and 'celcius_to_reamur' which takes a Celsius value 'C' and returns its equivalent in Reamur using the formula $R = 0.8C$. Below the function definitions, there are three test cases. Test-case 1 sets C=100.0 and prints the Fahrenheit result (212.0). Test-case 2 sets C=80.0 and prints the Reamur result (64.0). Test-case 3 sets C=0.0 and prints the Fahrenheit result (32.0). The bottom part of the screenshot shows the terminal output of the program, which matches the print statements in the code: 'Input C = 100.0 . Output F = 212.0', 'Input C = 80.0 . Output R = 64.0', and 'Input C = 0.0 . Output F = 32.0'.

```

Week 4 > github-week4 > latihan-4.3.py > ...
1 # Fungsi konversi Celcius ke Fahrenheit
2 celcius_to_fahrenheit = lambda C: (9/5) * C + 32
3
4 # Fungsi konversi Celcius ke Reamur
5 celcius_to_reamur = lambda C: 0.8 * C
6
7 # Test-case 1
8 C = 100.0
9 F = celcius_to_fahrenheit(C)
10 print("Input C =", C, ". Output F =", F)
11
12 # Test-case 2
13 C = 80.0
14 R = celcius_to_reamur(C)
15 print("Input C =", C, ". Output R =", R)
16
17 # Test-case 3
18 C = 0.0
19 F = celcius_to_fahrenheit(C)
20 print("Input C =", C, ". Output F =", F)

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

(base) maik@Michaels-MacBook-Air Praktek Algoritma (Semester 2) % python -u "/Users/maik/Documents/Praktek Algoritma (Semester 2)/Week 4/github-week4/latihan-4.3.py"
Input C = 100.0 . Output F = 212.0
Input C = 80.0 . Output R = 64.0
Input C = 0.0 . Output F = 32.0
(base) maik@Michaels-MacBook-Air Praktek Algoritma (Semester 2) %

```

Gambar 4.6: Source Code dan Output Latihan Mandiri 4.3

- Penjelasan
Kode di atas mendefinisikan dua fungsi konversi: 'celcius_to_fahrenheit' untuk mengubah suhu dari Celcius ke Fahrenheit, dan 'celcius_to_reamur' untuk mengubah suhu dari Celcius ke Reamur.
- Fungsi 'celcius_to_fahrenheit' mengambil satu parameter, 'C', yang merupakan suhu dalam Celcius, dan mengembalikan hasil konversi ke Fahrenheit.
- Fungsi 'celcius_to_reamur' juga mengambil satu parameter, 'C', yang merupakan suhu dalam Celcius, dan mengembalikan hasil konversi ke Reamur.

Setelah mendefinisikan kedua fungsi, beberapa kasus uji dijalankan:

- Kasus uji pertama mengkonversi suhu 100 derajat Celcius ke Fahrenheit.
- Kasus uji kedua mengkonversi suhu 80 derajat Celcius ke Reamur.
- Kasus uji ketiga mengkonversi suhu 0 derajat Celcius ke Fahrenheit.

Hasil dari setiap kasus uji dicetak ke layar untuk diperiksa.