



Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	71231051
Nama Lengkap	Amida A Ronsumbre
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

Penjelasan materi 1

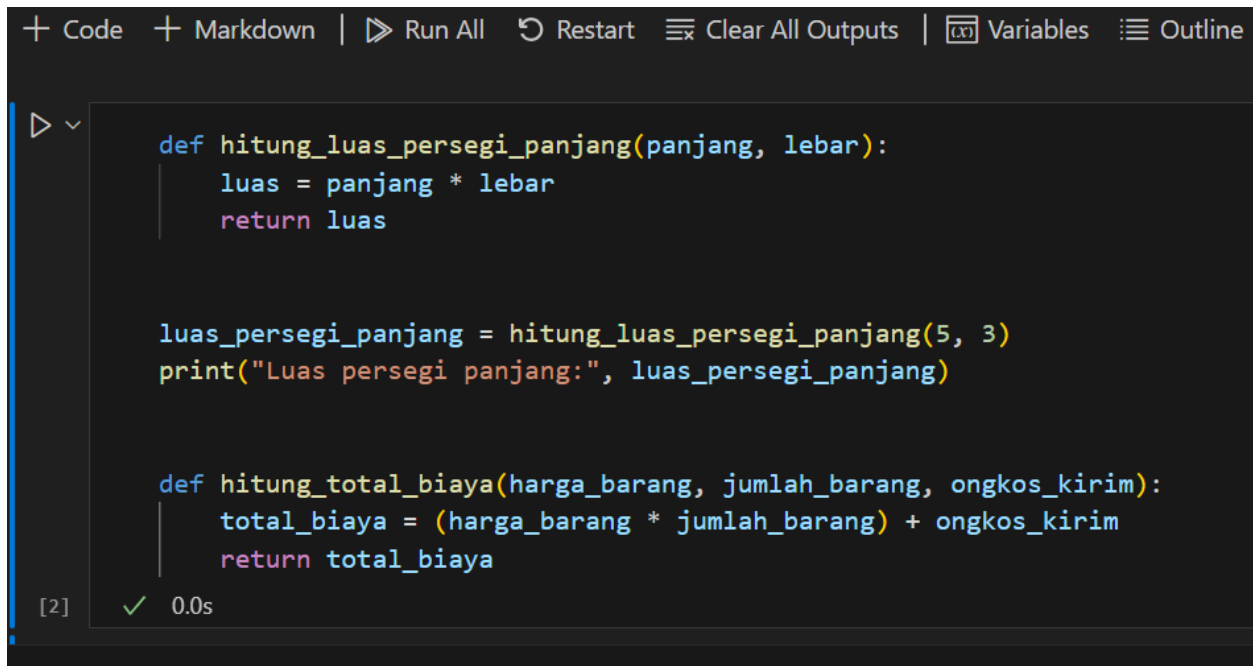
- 4.1 Tujuan Praktikum

Memahami anatomi dan struktur fungsi dan penggunaan parameter dan hasil fungsi dan mendefinisikan fungsi dan parameter untuk memecahkan masalah.

Tentu, berikut adalah penjelasan singkat dan ringkas untuk setiap pernyataan:

1. Memahami anatomi dan struktur fungsi: Ini berarti pemahaman tentang bagaimana sebuah fungsi dalam pemrograman terstruktur, terdiri dari nama fungsi, parameter, dan body fungsi.
2. Memahami penggunaan parameter dan hasil fungsi: Ini merujuk pada kemampuan menggunakan parameter sebagai input untuk fungsi dan memahami bagaimana hasil dari fungsi tersebut dihasilkan berdasarkan parameter yang diberikan.
3. Mendefinisikan fungsi dan parameter untuk memecahkan masalah: Ini melibatkan kemampuan merancang fungsi dengan mempertimbangkan parameter yang diperlukan untuk memecahkan masalah tertentu, sehingga fungsi tersebut dapat digunakan untuk menyelesaikan masalah yang diberikan.

Dibawah ini adalah codingan dari (Anatomi fungsi, penggunaan fungsi, dan parameter fungsi):



```
+ Code + Markdown | ▶ Run All ↺ Restart ☰ Clear All Outputs | 📄 Variables ☰ Outline

▶ ▼

def hitung_luas_persegi_panjang(panjang, lebar):
    luas = panjang * lebar
    return luas

luas_persegi_panjang = hitung_luas_persegi_panjang(5, 3)
print("Luas persegi panjang:", luas_persegi_panjang)

def hitung_total_biaya(harga_barang, jumlah_barang, ongkos_kirim):
    total_biaya = (harga_barang * jumlah_barang) + ongkos_kirim
    return total_biaya

[2] ✓ 0.0s
```

- 4.2 Alat dan Bahan

Pada praktikum ini, diperlukan perangkat komputer yang memiliki spesifikasi minimum dan perangkat lunak berikut:

- ❖ Spesifikasi Perangkat Keras:

1. Terkoneksi ke Internet.
2. Mampu menjalankan Windows 10 atau Ubuntu Linux.

- ❖ Perangkat Lunak yang Diperlukan:

1. Python 3.7 atau 3.8, terpasang melalui Anaconda atau instalasi Python lainnya.
2. Web Browser: Mozilla Firefox, Microsoft Edge, atau Google Chrome.
3. Command Prompt (untuk Windows) atau Terminal (untuk Linux).
4. Editor Python: Visual Studio Code, PyCharm, Spyder, atau editor lain yang mendukung Python.

Dengan memenuhi spesifikasi tersebut, praktikum dapat dilaksanakan dengan baik.

- 4.3.1 Fungsi, Argument dan Parameter

Pernyataan tersebut menjelaskan tentang fungsi dalam pemrograman Python dan kaitannya dengan modular programming. Fungsi adalah kumpulan perintah yang memiliki tujuan khusus dan dapat

digunakan ulang. Ada dua jenis fungsi: fungsi bawaan (built-in function) dan fungsi yang dibuat sendiri oleh programmer. Fungsi dibuat dengan menggunakan keyword ``def``, diikuti dengan nama fungsi dan parameter jika ada, serta diakhiri dengan blok kode yang menjelaskan tindakan yang dilakukan oleh fungsi tersebut. Penggunaan fungsi ini memungkinkan pemrogram untuk memecah program menjadi bagian-bagian yang lebih kecil dan dapat dikelola dengan lebih mudah.

Contoh konkretnya adalah dengan mendefinisikan fungsi ``tambah()`` yang memiliki dua parameter, kemudian fungsi tersebut dipanggil dengan memberikan argument ke dalamnya. Jalannya program akan mengikuti urutan yang ditentukan oleh pemanggilan fungsi dan eksekusi kode di dalamnya. Jika pemanggilan fungsi dilakukan sebelum definisi fungsi, atau fungsi didefinisikan di bawah pemanggilan, akan menyebabkan kesalahan dalam program.

- 4.3.2 Return Value

Berdasarkan hasil yang dikeluarkan oleh fungsi, secara umum ada dua jenis: fungsi yang tidak mengembalikan nilai (void function) dan fungsi yang mengembalikan nilai.

Contoh fungsi yang tidak mengembalikan nilai adalah ``print_twice()``, yang menampilkan pesan yang diberikan ke layar dua kali.

Contoh fungsi yang mengembalikan nilai adalah ``tambah()``, yang melakukan penjumlahan tiga bilangan dan mengembalikan hasilnya.

Fungsi ``print_twice()`` tidak menghasilkan nilai yang dapat digunakan untuk proses berikutnya, sedangkan fungsi ``tambah()`` mengembalikan hasil penjumlahan yang dapat digunakan dalam proses selanjutnya, seperti mencari rata-rata tiga bilangan.

- 4.3.3 Optional Argument dan Named Argument

Fungsi dapat memiliki parameter opsional, yang memiliki nilai bawaan yang telah ditentukan sebelumnya. Untuk mendefinisikan parameter opsional, nilai defaultnya diberikan di dalam tanda kurung saat definisi fungsi.

Contoh fungsi ``hitung_belanja()`` memiliki dua parameter: ``belanja`` dan ``diskon``, di mana ``diskon`` memiliki nilai default 0 (artinya 0%). Fungsi ini menghitung jumlah yang harus dibayar dengan mengurangi diskon dari total belanja. Pemanggilan fungsi ``hitung_belanja()`` dapat dilakukan dengan satu atau dua argumen. Jika hanya satu argumen yang diberikan, nilai diskon akan dianggap 0. Contoh pemanggilan:

```
print(hitung_belanja(100000)) # Output: 100000.0
print(hitung_belanja(100000, 10)) # Output: 90000.0
print(hitung_belanja(100000, 50)) # Output: 50000.0
```

Setiap

parameter pada fungsi memiliki nama. Oleh karena itu, saat memanggil fungsi, Anda dapat menyertakan nama parameter tanpa harus memperhatikan urutan. Ini disebut sebagai argumen yang dinamai (named arguments).

Contoh, dalam fungsi `cetak()`, argumen dapat disebutkan dengan namanya:

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

cetak(20, 30, 40)
```

Anda juga dapat memanggil fungsi `cetak()` dengan menyebutkan nama argumennya dalam urutan yang berbeda. Ini akan menghasilkan output yang sama seperti sebelumnya, meskipun urutan argumennya berbeda.

```
cetak(20, 30, 40)
cetak(a=20, b=30, c=40)
```

• 4.3.4 Anonymous Function (Lambda)

Ya, Anda telah memberikan penjelasan yang baik tentang anonymous function (fungsi tanpa nama) di Python menggunakan kata kunci `lambda`. Ini adalah cara singkat untuk membuat fungsi sederhana dalam baris kode yang tidak memerlukan definisi formal menggunakan kata kunci "def".

Penjelasan Anda tentang bagian-bagian dari sebuah anonymous function di Python juga tepat:

1. Keyword: "lambda".
2. Bound variable: Argumen-argumen fungsi.
3. Body: Ekspresi atau pernyataan yang menghasilkan nilai yang akan dikembalikan oleh fungsi.

Anda juga memberikan contoh yang baik dengan membandingkan definisi fungsi `tambah` menggunakan `def` dan menggunakan `lambda`. Ini membantu memperjelas bagaimana `lambda` digunakan untuk membuat anonymous function dengan cara yang lebih ringkas.

- 4.4.1 Mendefinisikan Fungsi

Fungsi yang Anda definisikan, `tagihan_listrik`, cukup jelas dan sesuai dengan kebutuhan yang dijelaskan dalam permasalahan. Berikut adalah penjelasan singkat tentang fungsi tersebut:

1. `tagihan_listrik`: Ini adalah nama fungsi yang Anda definisikan.
2. `pemakaian`: Parameter pertama yang mewakili jumlah pemakaian listrik dalam kWh.
3. `golongan`: Parameter opsional kedua yang menunjukkan golongan tarif. Jika tidak diberikan, maka akan menggunakan nilai default yaitu golongan 3.

Anda telah menggunakan logika yang tepat untuk menghitung tagihan listrik berdasarkan jumlah pemakaian dan golongan tarif yang diberikan. Penggunaan `if-elif-else` dengan baik memungkinkan Anda untuk memilih tarif yang sesuai berdasarkan golongan yang diberikan. Contoh penggunaan fungsi ini dalam kode program memberikan hasil yang sesuai dengan yang diharapkan. Ini menunjukkan bahwa fungsi Anda telah diimplementasikan dengan benar. Jika ada pertanyaan lebih lanjut atau perbaikan yang diperlukan, jangan ragu untuk bertanya!

4.4.2 Argument dan Parameter

Output dari kode program tersebut adalah:

Penjelasan:

1. Saat fungsi `abc` dipanggil dengan parameter `nilai1`, `nilai2`, dan `nilai3`, nilai-nilai tersebut disalin ke dalam parameter `a`, `b`, dan `c` di dalam fungsi.
2. Di dalam fungsi, nilai-nilai `a`, `b`, dan `c` diubah berdasarkan operasi aritmatika yang diberikan.
3. Namun, perubahan nilai tersebut hanya berlaku di dalam lingkup fungsi. Variabel-variabel `nilai1`, `nilai2`, dan `nilai3` di luar fungsi tidak terpengaruh oleh perubahan ini karena pada Python, argumen yang diberikan kepada fungsi adalah objek dan parameter yang menerima argumen merupakan referensi ke objek tersebut.
4. Karena objek yang diteruskan adalah immutable (integer), maka perubahan nilai di dalam fungsi tidak memengaruhi nilai variabel asli di luar fungsi.

Sehingga, meskipun di dalam fungsi `abc` nilai-nilai `a`, `b`, dan `c` telah diubah, nilai-nilai `nilai1`, `nilai2`, dan `nilai3` tetap tidak berubah dan tetap sesuai dengan nilai awalnya saat fungsi dipanggil.

- 4.4.3 Anonymous/Lambda Function

Anda telah berhasil mengubah fungsi `kelipatan_sembilan` menjadi bentuk lambda function dengan benar. Berikut adalah bentuk lambda function-nya:

Dalam lambda function ini:

```
kelipatan_sembilan = lambda angka: angka % 9 == 0

kelipatan_sembilan(2000) # Output: False
```

- Bound variable: `angka` - Body: Pengecekan apakah `angka` habis dibagi 9 atau tidak (`angka % 9 == 0`).

Contoh penggunaan lambda function tersebut dapat dilihat di bawah ini:

Hasil cetak menunjukkan bahwa lambda function ini berfungsi seperti yang diharapkan, yaitu mengembalikan `True` jika angka yang diberikan adalah kelipatan 9 dan `False` jika tidak.

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

Latihan 4.1:

Berikut adalah implementasi fungsi `cek_angka()` sesuai dengan deskripsi yang Anda berikan:

```
def cek_angka(a, b, c):
    # Memeriksa apakah ketiga parameter nilainya berbeda
    if a != b and a != c and b != c:
        # Memeriksa apakah ada kemungkinan jumlah dua parameter sama dengan parameter lainnya
        if a + b == c or a + c == b or b + c == a:
            return True
        return False

    # Contoh penggunaan fungsi
    print(cek_angka(1, 2, 3)) # True, karena semua ketentuan terpenuhi
    print(cek_angka(2, 3, 5)) # False, karena tidak semua nilai berbeda
    print(cek_angka(3, 6, 9)) # True, karena semua ketentuan terpenuhi (3 + 6 = 9)
    print(cek_angka(4, 4, 8)) # False, karena tidak semua nilai berbeda
```

Fungsi ini bekerja dengan cara sebagai berikut:

1. Memeriksa apakah ketiga parameter memiliki nilai yang berbeda.
2. Jika semua nilai berbeda, maka fungsi akan memeriksa apakah ada pasangan dari dua parameter yang jumlahnya sama dengan parameter ketiga.

3. Jika kedua kondisi terpenuhi, maka fungsi akan mengembalikan `True`, jika tidak, akan mengembalikan `False`.

Contoh penggunaan fungsi menunjukkan output yang sesuai dengan yang diharapkan.

SOAL 2

Latihan 4.2:

Berikut adalah implementasi fungsi `cek_digit_belakang()` sesuai dengan deskripsi yang Anda berikan:

```
input_b = int(input("Masukkan bilangan kedua: "))
input_c = int(input("Masukkan bilangan ketiga: "))

# Memanggil fungsi cek_digit_belakang() dengan input dari pengguna
hasil = cek_digit_belakang(input_a, input_b, input_c)

# Menampilkan hasil
print("Output:", hasil)
```

... True
True
True
False

Fungsi ini bekerja dengan cara sebagai berikut:

```
def cek_digit_belakang(a, b, c):
    # Mengekstrak digit paling kanan dari setiap parameter
    digit_a = a % 10
    digit_b = b % 10
    digit_c = c % 10

    # Memeriksa apakah minimal dua dari tiga digit paling kanan sama
    if digit_a == digit_b or digit_a == digit_c or digit_b == digit_c:
        return True
    else:
        return False

# Meminta input dari pengguna
input_a = int(input("Masukkan bilangan pertama: "))
input_b = int(input("Masukkan bilangan kedua: "))
input_c = int(input("Masukkan bilangan ketiga: "))
```

1. Mengambil digit paling kanan dari setiap parameter dengan menggunakan operasi modulo (`% 10`).

2. Memeriksa apakah minimal dua dari tiga digit paling kanan sama.
3. Jika setidaknya dua digit paling kanan sama, fungsi mengembalikan `True`, jika tidak, mengembalikan `False`.

Anda dapat menggunakan kode di atas untuk memeriksa test case yang Anda sebutkan. Program akan meminta input dari pengguna dan kemudian menampilkan hasilnya sesuai dengan deskripsi yang diberikan.

SOAL 3

Latihan 4.3:

Ini merupakan pendekatan yang lebih langsung dan mudah dipahami. Kedua fungsi didefinisikan secara eksplisit menggunakan "Def", dan setiap fungsi memiliki satu pernyataan return yang menghitung hasil konversi sesuai rumus yang diberikan.

```
> # Fungsi konversi Celcius ke Fahrenheit
def celcius_to_fahrenheit(c):
    return (9/5) * c + 32

# Fungsi konversi Celcius ke Reamur
def celcius_to_reamur(c):
    return 0.8 * c

# Contoh penggunaan fungsi konversi suhu
input_c1 = 100
output_f1 = celcius_to_fahrenheit(input_c1)
print("Input C =", input_c1, "Output F =", output_f1)

input_c2 = 80
output_r2 = celcius_to_reamur(input_c2)
print("Input C =", input_c2, "Output R =", output_r2)

input_c3 = 0
output_f3 = celcius_to_fahrenheit(input_c3)
print("Input C =", input_c3, "Output F =", output_f3)
```

13] ✓ 0.0s

```
.. Input C = 100 Output F = 212.0
   Input C = 80 Output R = 64.0
   Input C = 0 Output F = 32.0
```