



Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	71231052
Nama Lengkap	GABRIEL SACHIO ATMADJAJA
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%)

Sumber Materi: Modul Pratikum Algoritma dan Pemrograman, kegiatan praktikum

MATERI 4

Fungsi adalah sekumpulan perintah-perintah yang dijadikan satu, memiliki suatu tujuan dan kegunaan khusus serta dapat digunakan ulang. Fungsi memiliki 2 jenis, yaitu **build-in function** (fungsi bawaan, cth: input, print, dll) dan fungsi yang dibuat sendiri (menggunakan kata kunci def diikuti nama fungsinya).

```
4 def _input(var):  
5     return var  
6  
7 a = _input("abc")  
8 print(a)
```

Gambar 1.1: Contoh fungsi

```
PS D:\Sachio\Kampus\PrakAlPro\Pert4> & C:/Users/Acer/AppData/Local/Programs/Python/Python312/python.exe d:/Sachio/Kampus/PrakAlPro/Pert4/fungsi.py  
abc
```

Gambar 1.2: Output dari kode di gambar 1.1

Perhatikan pada gambar 1.1, fungsi di deklarasi dengan kata kunci def diikuti nama fungsinya, lalu di dalam tanda kurung berisi parameter, tidak lupa tanda titik dua. Pada baris ke-5, kode perlu menjorok ke dalam dan diisi dengan hal yang ingin fungsi tersebut lakukan. Lalu kata kunci **return** digunakan untuk mengembalikan nilai, dalam kasus ini saya mengembalikan nilai dari parameter yang nanti diisi. Memanggil fungsi bisa dilakukan dengan mengetikkan nama fungsinya dan mengisi nilai parameter (dalam gambar: "abc"). Lalu karena menggunakan return, maka pemanggilan fungsi tersebut tidak akan menampilkan apapun. Untuk menampilkan outputnya perlu fungsi print() dan supaya mudah saja, pemanggilan fungsi bisa dibuat dalam variable.

Ada juga di dalam blok fungsi, tidak harus menggunakan kata kunci return. Bisa saja di fungsi tersebut langsung menggunakan fungsi print() untuk menampilkan output. Hal penting, saat fungsi dipanggil dan nilai parameternya diisi, itu disebut dengan **argument**. Perhatikan pada gambar 1.3, karena fungsi bertugas menampilkan output, untuk pemanggilan fungsi cukup dengan nama fungsi diikuti argument-nya. Jadi, tidak perlu menambahkan fungsi print.

```
10 def print_nilai(nama, a, b):  
11     print("nilai ", nama, ":", a + b)  
12  
13 print_nilai("Upin", 5, 7)  
14 print_nilai("Ipin", 13, 2)  
15 print_nilai("Apin", 4, 9)
```

Gambar 1.3: Penggunaan print dalam fungsi

```

PS D:\Sachio\Kampus\PrakAlPro\Pert4> & C:/Users/Acer/AppData/Local/Programs/Python/Python312/python.exe d:/Sachio/Kampus/PrakAlPro/Pert4/fungsi.py
abc
PS D:\Sachio\Kampus\PrakAlPro\Pert4> & C:/Users/Acer/AppData/Local/Programs/Python/Python312/python.exe d:/Sachio/Kampus/PrakAlPro/Pert4/fungsi.py
nilai Upin : 12
nilai Ipin : 15
nilai Apin : 13

```

Gambar 1.4: Output dari kode di gambar 1.3

Keterangan Gambar 1.3:

- Baris 10 -> berisi kode deklarasi fungsinya beserta parameter
- Baris 11 -> isi dari fungsi atau perintah yang ingin fungsi tersebut lakukan
- Baris 13, baris 14, dan baris 15 -> pemanggilan fungsi diikuti argumentnya
- Saat fungsi dipanggil, program akan kembali lompat ke blok kode-nya, dalam kasus saya ke baris 10. Lalu parameter tersebut akan diisi dengan argument
- Lanjut ke baris 11, program akan menjalankan perintahnya dan parameter tersebut akan diisikan argument

Berdasarkan hasil yang dikeluarkan oleh fungsi, secara umum ada dua jenis yaitu: fungsi yang tidak mengembalikan nilai dan fungsi yang mengembalikan nilai. Fungsi yang tidak mengembalikan nilai sering disebut sebagai *void function*. Contoh untuk void function bisa dilihat di gambar 1.3 (void function sebenarnya mengembalikan *None*), sedangkan contoh fungsi yang mengembalikan nilai bisa dilihat di gambar 1.1

Hal penting lainnya tentang fungsi adalah memastikan pemanggilan fungsi dibawah blok kode fungsi. Ini karena program dibaca dari atas dan lalu turun. Bila pemanggilan fungsi ditulis terlebih dahulu sebelum blok kode fungsi, maka program dapat dipastikan Error atau fungsi yang dipanggil tidak dikenali.

Parameter dalam fungsi dapat ditentukan nilai default-nya terlebih dahulu, disebut dengan *optional parameter*. Contoh:

```

22
23 def hitung_belanja(hargaBarang, diskon = 0.1):
24     bayar = hargaBarang - (hargaBarang * diskon)/100
25     return bayar
26
27 print(hitung_belanja(100000))

```

Gambar 1.5: Contoh optional parameter

Perhatikan gambar 1.5, terdapat parameter hargaBarang dan diskon. Pada parameter diskon terdapat nilai defaultnya yaitu 0,1. Nilai default ini nantinya akan dijalankan semisal saat pemanggilan fungsi, parameter diskon tidak diisikan nilai.

Berikut output-nya:

```

PS D:\Sachio\Kampus\PrakAlPro\Pert4> & C:/Users/Acer/AppData/Local/Programs/Python/Python312/python.exe d:/Sachio/Kampus/PrakAlPro/Pert4/fungsi.py
99900.0

```

Gambar 1.6: Output

Lalu ada yang disebut *Named Argument* yaitu pemanggilan fungsi dengan menyebutkan nama argument-nya. Jika menggunakan cara ini, urutan argument tidak akan berpengaruh. Contoh:

```
32 def lingkaran(jari_jari, pi = 3.14):
33     luas = 0.5 * pi * jari_jari ** 2
34     return luas
35
36 print(lingkaran(pi = 3.14, jari_jari = 5))
```

Gambar 1.7: Named Argument

Berikut output-nya:

```
PS D:\Sachio\Kampus\PrakAlPro\Pert4> & C:/Users/Acer/AppData/Local/Programs/Python/Python312/python.exe d:/Sachio/Kampus/PrakAlPro/Pert4/fungsi.py
39,25
```

Gambar 1.8: Output

Dalam Python, fungsi bisa dibuat dalam satu baris dan tanpa nama disebut sebagai *anonymous function*. Kata kunci lambda digunakan untuk mendefinisikan anonymous function. Contoh:

```
lambda.py > ...
1 # BENTUK LAMBDA FUNGSI (ANONYMOUS FUNCTION)
2 hasil = lambda sisi: (sisi ** 2)
3 print(hasil(20))
```

Gambar 1.9: Lambda untuk membuat anonymous function

Keterangan gambar 1.9:

- Pertama, membuat variable
- Isikan variable dengan kata kunci lambda diikuti bound variable (cth: sisi) dan body (sisi ** 2)

```
10 # BENTUK FUNGSI YG NORMAL
11 def luas_persegi(sisi):
12     luas = sisi ** 2
13     return luas
14
15 print(luas_persegi(20))
16
```

Gambar 1.10: Bentuk fungsi normal dari lambda function di gambar 1.9

Kode pada gambar 1.9 maupun gambar 1.10 akan menghasilkan output yang sama.

```
PS D:\Sachio\Kampus\PrakAlPro\Pert4> & C:/Users/Acer/AppData/Local/Programs/Python/Python312/python.exe d:/Sachio/Kampus/PrakAlPro/Pert4/lambda.py
400
```

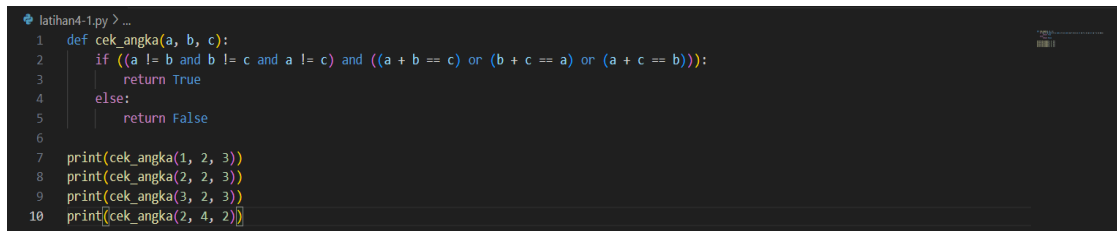
Gambar 1.11: Output

BAGIAN 2: LATIHAN MANDIRI (60%)

Source Code: <https://github.com/gabrielsachioa/PrakAIPro4.git>

SOAL 1

Berikut kode programnya:



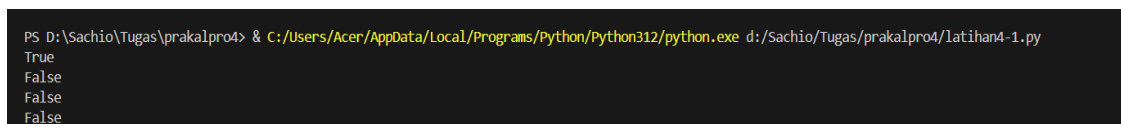
```
1 def cek_angka(a, b, c):
2     if ((a != b and b != c and a != c) and ((a + b == c) or (b + c == a) or (a + c == b))):
3         return True
4     else:
5         return False
6
7 print(cek_angka(1, 2, 3))
8 print(cek_angka(2, 2, 3))
9 print(cek_angka(3, 2, 3))
10 print(cek_angka(2, 4, 2))
```

Gambar 2.1: Kode Program Cek Angka

Keterangan gambar 2.1:

- Baris 1, 2, 3, 4, 5 -> definisi fungsi cek_angka dan belum ada yang dijalankan
- karena ingin outputnya ada 2 (True atau False) maka membuat percabangan if dan else. Pada percabangan if ada 2 kondisi yang harus terpenuhi supaya mengembalikan nilai True, yaitu nilai ketiga parameter berbeda dan ada kemungkinan jika diambil dua parameter serta dijumlahkan hasilnya sama dengan parameter lainnya. Maka diantara 2 kondisi akan menggunakan operator logika **and**. Kondisi pertama (nilai ketiga parameter berbeda) bisa dibuat dengan menggunakan operator **!=** (artinya tidak sama dengan) dan operator **and**. Lalu kondisi kedua, dibuat dengan menjumlahkan 2 parameter ($a + b$) dan dibandingkan nilainya ($==$) dengan parameter yang tersisa. Lalu untuk percabangan else, saya mengembalikan nilai False.
- Baris 7, 8, 9, 10 -> memanggil fungsi dan memasukkan argument, lalu menampilkan outputnya
- Setelah fungsi dipanggil dan dimasukkan argument, program akan kembali ke baris 1-5 dengan mengganti parameternya dengan argument, lalu mengecek kondisi pada percabangan. Jika kondisi di salah satu percabangan terpenuhi, maka akan mengembalikan nilai sesuai dengan percabangannya. Lalu kembali ke baris 7-10, dan menampilkan outputnya

Berikut outputnya:

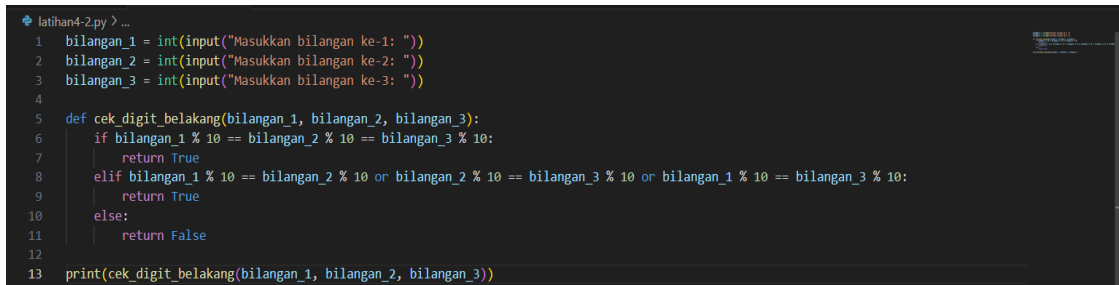


```
PS D:\Sachio\Tugas\prakalpro4> & C:/Users/Acer/AppData/Local/Programs/Python/Python312/python.exe d:/Sachio/Tugas/prakalpro4/latihan4-1.py
True
False
False
False
```

Gambar 2.2: Output fungsi cek_angka

SOAL 2

Berikut kode programnya:



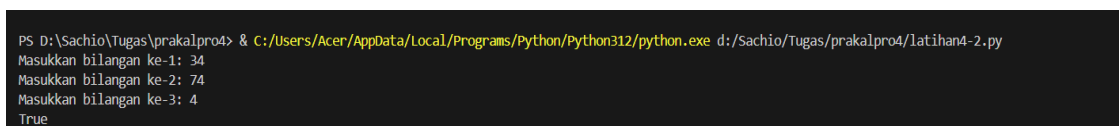
```
1  bilangan_1 = int(input("Masukkan bilangan ke-1: "))
2  bilangan_2 = int(input("Masukkan bilangan ke-2: "))
3  bilangan_3 = int(input("Masukkan bilangan ke-3: "))
4
5  def cek_digit_belakang(bilangan_1, bilangan_2, bilangan_3):
6      if bilangan_1 % 10 == bilangan_2 % 10 == bilangan_3 % 10:
7          return True
8      elif bilangan_1 % 10 == bilangan_2 % 10 or bilangan_2 % 10 == bilangan_3 % 10 or bilangan_1 % 10 == bilangan_3 % 10:
9          return True
10     else:
11         return False
12
13  print(cek_digit_belakang(bilangan_1, bilangan_2, bilangan_3))
```

Gambar 2.3: Fungsi cek digit belakang

Keterangan gambar 2.3:

- Baris 1, 2, 3 -> membuat 3 variable yang meminta input bilangan. Untuk meminta input, saya gunakan fungsi input dan dibungkus dengan keyword int
- Baris 5, 6, 7, 8, 9, 10 -> definisi fungsi dan belum dijalankan
- Baris 5 -> nama fungsinya cek_digit_belakang dan parameternya 3 variable yang dibuat di baris 1,2, dan 3
- Program akan mengembalikan 2 nilai yaitu True atau else, maka dibuatlah percabangan. Pada percabangan if, kondisinya adalah jika digit paling kanan ketiga bilangan sama maka kembalikan nilai True. Untuk mengambil digit paling kanan, caranya adalah dengan menggunakan % (modulo) 10. Artinya adalah setiap bilangan yang dimodulo 10 akan mengembalikan satuan. Lalu setelah itu tinggal dibandingkan dengan parameter lainnya menggunakan ==
- Pada percabangan elif, kondisinya adalah jika terdapat 2 variable dengan digit paling kanan sama, maka kembalikan nilai True. Mirip dengan kondisi di percabangan if, hanya saja membandingkan 2 variable saja dengan setiap kemungkinannya. Misalnya kemungkinan bilangan 1 dan 2 sama atau bilangan 2 dan 3 yang sama. Untuk itu saya gunakan operator logika untuk menghubungkan kemungkinan-kemungkinan itu.
- Pada percabangan else, saya mengembalikan nilai False
- Baris 13 -> memanggil fungsi dan menampilkan output-nya dengan perintah print. Alasan kenapa saat memanggil fungsi parameternya tidak diisi argument/nilai adalah karena programnya ingin mengambil nilai tersebut dari input yang diberikan.

Berikut outputnya:



```
PS D:\Sachio\Tugas\prakalpro4> & C:/Users/Acer/AppData/Local/Programs/Python/Python312/python.exe d:/Sachio/Tugas/prakalpro4/latihan4-2.py
Masukkan bilangan ke-1: 34
Masukkan bilangan ke-2: 74
Masukkan bilangan ke-3: 4
True
```

Gambar 2.4: Output bila 3 digit paling kanan sama

```
PS D:\Sachio\Tugas\prakalpro4> & C:/Users/Acer/AppData/Local/Programs/Python/Python312/python.exe d:/Sachio/Tugas/prakalpro4/latihan4-2.py
Masukkan bilangan ke-1: 17
Masukkan bilangan ke-2: 23
Masukkan bilangan ke-3: 67
True
```

Gambar 2.5: Output bila 2 digit paling kanan sama

```
PS D:\Sachio\Tugas\prakalpro4> & C:/Users/Acer/AppData/Local/Programs/Python/Python312/python.exe d:/Sachio/Tugas/prakalpro4/latihan4-2.py
Masukkan bilangan ke-1: 11
Masukkan bilangan ke-2: 12
Masukkan bilangan ke-3: 13
False
```

Gambar 2.6: Output bila ketiga digit paling kanan bilangan tidak sama

SOAL 3

Berikut kode programnya:

```
latihan4-3.py > ...
1 celcius_fahrenheit = lambda C: (9/5) * C + 32
2 celcius_reamur = lambda C: 0.8 * C
3
4 print(celcius_fahrenheit(100))
5 print(celcius_reamur(80))
6 print(celcius_fahrenheit(0))
```

Gambar 2.7: Lambda Function Konversi Suhu

Keterangan gambar 2.5:

- Baris 1 -> untuk membuat lambda function konversi celcius to fahrenheit, awalnya saya membuat variable celcius_fahrenheit lalu diisi dengan kata kunci lambda (untuk membuat lambda function), lalu bound variabelnya C, dan di body saya isi dengan rumus konversinya yaitu $(9/5) * C$ (celcius) + 32
- Baris 2 -> untuk membuat lambda function konversi celcius to reamur, awalnya saya membuat variable celcius_reamur lalu diikuti kata kunci lambda. Lalu bound variabelnya C, dan di body saya isi dengan rumus konversinya yaitu $0,8 * C$
- Baris ke 4, 5, 6 -> memanggil lambda function dan mengisi nilai bound variabelnya, lalu menjalankan kode di body. Setelah itu di tampilkan dengan perintah print

Berikut outputnya:

```
PS D:\Sachio\Tugas\prakalpro4> & C:/Users/Acer/AppData/Local/Programs/Python/Python312/python.exe d:/Sachio/Tugas/prakalpro4/latihan4-3.py
212.0
64.0
32.0
```

Gambar 2.8: Output dari lambda function konversi suhu