Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	**71230982**
Nama Lengkap	**Rainie Fanita Chrisabel Hadisantoso**
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

Modular Programming

Tujuan dari pertemuan ke-4 ini adalah mahasiswa dapat memahami anatomi dan struktur dari fungsi, dapat menggunakan parameter dan mengidentifikasi hasil dari suatu fungsi, dan mendefinisikan fungsi serta menggunakannya dalam sebuah algoritma untuk memecahkan suatu masalah. Modular programming adalah

Fungsi, Argument dan Parameter

Fungsi adalah fitur bawaan dari bahasa pemrograman yang digunakan. Setiap fungsi memiliki manfaatnya masing-masing. Beberapa contoh fungsi dari Python yang sudah sering kita gunakan adalah fungsi input() dan fungsi print().

belajar python 2 > alpro4.1.py > ...

```
1   # fungsi input() dan fungsi print()
2   # penulisan
3   print("Hello, world!")
4   a = input()
5   print(a)
```

Gambar 1. Fungsi input & Fungsi print

Fungsi input() digunakan untuk membaca data yang dimasukkan oleh pengguna. Sedangkan fungsi print() digunakan untuk menampilkan tulisan maupun hasil dari program di layar.

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\asus\belajar-git> python -u "c:\Users\asus\belajar-git\belajar python 2\
Hello, world!
a
a
```

Gambar 2. Hasil fungsi input & fungsi print

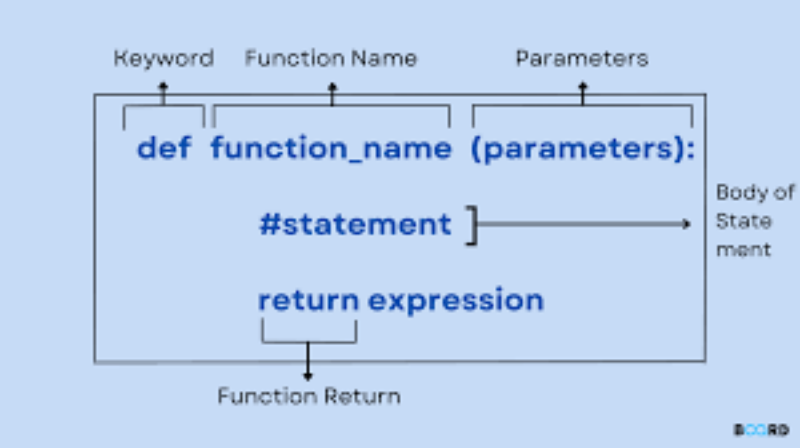
Secara garis besar, fungsi berupa kumpulan perintah yang dijadikan satu sehingga memiliki manfaat dan dapat mencapai sebuah tujuan.

Terdapat dua jenis fungsi:

- Fungsi bawaan dari bahasa pemrograman (Built-in function)
- Fungsi yang dibuat oleh programmer sendiri.

Terdapat beberapa peraturan untuk membuat fungsi sendiri:

- Menggunakan keyword def untuk mendefinisikan (memberi nama) fungsi.
- Memberi nama fungsi memiliki aturan yang sama dengan nama variabel.
- Isi atau kode dalam fungsi harus menjorok 1 tab ke dalam.
- Menjabarkan dua variabel atau data yang diperlukan dalam fungsi
- Untuk menampilkan hasil dapat dilakukan return, print(), maupun memanggil dan print fungsi.



Gambar 3. Struktur Fungsi Buatan

Berikut contoh beberapa program fungsi buatan:

```
7   def hitung(harga, diskon):
8       bayar = harga - (harga*diskon/100)
9       return bayar
10
11  print(hitung(100000, 20))
```

Gambar 4. Contoh Fungsi Menghitung Diskon

Return Value

Ada dua jenis hasil yang dikeluarkan oleh fungsi: Fungsi yang tidak mengembalikan nilai dan fungsi yang mengembalikan nilai. Berikut perbedaan kedua jenis fungsi berdasarkan jenis hasil yang dikeluarkan tersebut:

1. Fungsi yang tidak mengembalikan nilai fungsi disebut juga void function.

```
14  def nilai(nama,a,b):
15      print(f"Nilai {nama} adalah {a + b}")
```

```
PROBLEMS 3   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

PS C:\Users\asus\belajar-git> python -u "c:\Users\asus\belajar-git\belajar python 2\

None
```

Gambar 5. Contoh Fungsi Tidak Mengembalikan Nilai (Void Function)

Hasil dari fungsi tidak dengan otomatis akan ditampilkan atau tidak ada)none), karena tidak ada perintah mengembalikan (return) dalam fungsi tersebut. Karena itu untuk menggunakan void function perlu ditambahkan fungsi print() untuk memanggil hasil fungsi tersebut.

```python
def nilai(nama,a,b):
    print(f"Nilai {nama} adalah {a + b}")

print(nilai("upin",4,5))
```

```
PROBLEMS 3  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\asus\belajar-git> python -u "c:\Users\asus\belajar-git\belajar python 2\

Nilai upin adalah 9
```

Gambar 6. Contoh Fungsi Tidak Mengembalikan Nilai (Void Function) 2

2. Fungsi yang tidak mengembalikan nilai dapat diubah menjadi fungsi mengembalikan nilai. Untuk mengubahnya hanya perlu menambahkan perintah untuk mengembalikan hasil (return). Keyword return digunakan untuk mengeluarkan nilai hasil dari fungsi sekaligus mengakhiri fungsi.

```python
def hitung(harga, diskon):
    bayar = harga - (harga*diskon/100)
    return bayar

print(hitung(100000, 20))
```

```
PROBLEMS 3  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\asus\belajar-git> python -u "c:\Users\asus\belajar-git\belajar python 2\

80000.0
```

Gambar 7. Contoh Fungsi Mengembalikan Nilai

Dengan adanya perintah return, kita tidak perlu menggunakan fungsi print() untuk menampilkan hasil dari fungsi tersebut dari dalam fungsi. Tetapi kita tetap perlu menggunakan operator print() untuk memanggil dan menampilkan hasil fungsi dari luar fungsi.

Optional Argument dan Named Argument

1. Optional Argument

Optional Argument adalah nilai parameter default yang dapat diberikan pada parameter dalam sebuah fungsi. Untuk memberikan optional argument dapat ditambahkan seperti: nama parameter = nilai default dalam parameter. Apabila tidak ada nilai default parameter, dan pengguna tidak memasukkan parameter yang diminta dengan lengkap, maka program akan menampilkan hasil error.

belajar python 2 > alpro4.1.py > ...

```python
def lingkaran(jari, pi ):
    luas= (jari**2)*pi*1/2
    return luas

print(lingkaran(20,3.14))
print(lingkaran(20,))
```

```
PROBLEMS 3   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

PS C:\Users\asus\belajar-git> python -u "c:\Users\asus\belajar-git\belajar python 2\al
628.0
Traceback (most recent call last):
  File "c:\Users\asus\belajar-git\belajar python 2\alpro4.1.py", line 32, in <module>
    print(lingkaran(20,))
          ^^^^^^^^^^^^^
TypeError: lingkaran() missing 1 required positional argument: 'pi'
```

Gambar 8. Contoh Tanpa Optional Atgument

Dengan optional argument, ketika pengguna tidak memberikan input nilai, komputer akan menggunakan nilai default yang telah diberikan. Sehingga program akan tetap berjalan sesuai dengan nilai default yang diberikan.

```
belajar python 2 > alpro4.1.py > lingkaran > pi
27  def lingkaran(jari, pi= 3.14):
28      luas= (jari**2)*pi*1/2
29      return luas
30
31  print(lingkaran(20,3.14))
32  print(lingkaran(20,))

PROBLEMS 3   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

PS C:\Users\asus\belajar-git> python -u "c:\Users\asus\belajar-git\belajar python 2\
628.0
628.0
```

Gambar 9. Contoh Dengan Optional Argument

2. Named Argument

Named Argument adalah memasukkan nilai-nilai parameter yang dibutuhkan sebuah fungsi dengan menggunakan nama parameter-parameternya. Biasanya untuk memasukkan nilai parameter kita hanya perlu memanggil fungsi dan memasukkan nilai-nilai yang dibutuhkan dengan menggunakan tanda koma (,) sebagai pemisah.

```
27  def lingkaran(jari, pi= 3.14):
28      luas= (jari**2)*pi*1/2
29      return luas
30
31
32  print(lingkaran(20,3.14))
33
34  print(lingkaran(3.14, 20))
```

PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\asus\belajar-git> python -u "c:\Users\asus\belajar-git\belajar python 2\
628.0
98.596
```

Gambar 10. Contoh Tanpa Named Argument

Dalam hal ini, komputer akan membaca nilai-nilai yang kita masukkan secara urut dan menyimpannya dalam tiap-tiap variabel parameter sesuai urutannya. Sehingga ketika kita memasukkan nilai yang dibutuhkan parameter terbalik seperti contoh di atas, hasil dari fungsi tidak sesuai dengan apa yang diinginkan.

Namun dengan menggunakan named argument atau menamai nilai parameter, kita dapat memasukkan nilai yang dibutuhkan dengan urutan bebas, dengan catatan nilai-nilai harus diberi nama sesuai parameter-parameter yang ada.

```
27  def lingkaran(jari, pi= 3.14):
28      luas= (jari**2)*pi*1/2
29      return luas
30
31
32  print(lingkaran(20,3.14))
33
34  print(lingkaran(3.14, 20))
35  print(lingkaran(pi = 3.14, jari= 20))
```

PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\asus\belajar-git> python -u "c:\Users\asus\belajar-git\belajar python 2\
628.0
98.596
628.0
```

Gambar 11. Contoh Dengan Named Argument

Dengan demikian komputer akan membaca nilai dan memasukkannya sesuai nama variabel yang kita pilihkan.

Anonymous Function (Lambda)

Lambda adalah anonymous function atau fungsi tanpa nama, karena lambda tidak perlu memiliki nama fungsi untuk dapat dijalankan, hanya seperti nama variabel. Lambda merupakan fitur tambahan pada Python. Anonymous function atau lambda memiliki struktur program seperti berikut:

1. Keyword: lambda
2. Bound variable: variabel berisikan fungsi lambda
3. Body: berisikan ekspresi atau statement untuk menjalankan fungsi lambda
4. Input: untuk membaca nilai-nilai dari pengguna (jika diperlukan)
5. Print: untuk menampilkan hasil dari fungsi lambda

Berikut contoh sebuah fungsi kalkulator dan bentuk fungsi tersebut jika menggunakan lambda:

Fungsi biasa:

belajar python 2 > alpro4.py > ...

```
38  def calculator(x, y, op):
39      if op == "+":
40          hasil = x + y
41      elif op == "-":
42          hasil = x - y
43      elif op == "/":
44          hasil = x / y
45      else:
46          hasil = x * y
47
48      return hasil
49
50  x = float(input("Masukkan x= "))
51  y = float(input("Masukkan y= "))
52  op = input("Masukkan operator (+,-,/,*)= ")
53
54  print(calculator(x,y,op))
```

PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\asus\belajar-git> python -u "c:\Users\asus\belajar-git\belajar python 2\
Masukkan x= 2
Masukkan y= 3
Masukkan operator (+,-,/,*)= +
5.0
```

Gambar 12. Fungsi Def Kalkulator

Fungsi lambda:

```
belajar python 2 > alpro4.py > calculator
56  calculator = {
57      "+":lambda x,y : x + y,
58      "-":lambda x,y : x - y,
59      "/":lambda x,y : x / y,
60      "*":lambda x,y : x * y
61  }
62
63  x = float(input("Masukkan x= "))
64  y = float(input("Masukkan y= "))
65  op = input("Masukkan operator (+,-,/,*)= ")
66
67  print(calculator[op](x,y))
```

```
PROBLEMS 3  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\asus\belajar-git> python -u "c:\Users\asus\belajar-git\belajar python 2\
```

```
Masukkan x= 2
Masukkan y= 3
Masukkan operator (+,-,/,*)= +
5.0
```

Gambar 13. Fungsi Lambda Kalkulator

Kalkulator versi fungsi lambda menggunakan array untuk menyimpan rumus-rumus kalkulator. Dengan menggunakan fungsi lambda tidak perlu menggunakan percabangan, hanya perlu definisi operator diikuti lambda, nilai-nilai, dan rumus. Kemudian fungsi input agar pengguna dapat memasukkan nilai-nilai dan operator yang hendak digunakan. Terakhir fungsi dipanggil dengan print berisi nama fungsi lambda dan parameter-parameternya.

LATIHAN MANDIRI

SOAL 4.1

Membuat fungsi dengan nama cek_angka berisi tiga parameter: a, b, dan c. Ketiga parameter harus memenuhi dua kondisi untuk menghasilkan True, ketika tidak terpenuhi keduanya maka False:

1. Ketiganya memiliki nilai yang berbeda.
2. Ketika dua parameter dijumlah hasilnya parameter lainnya.

a. Source Code

```
[2]  def cek_angka(a,b,c):
       if a != b != c  and (a + b == c or a + c == b or b + c == a):
         return True
       else:
         return False

     print(cek_angka(1,2,3))
     print(cek_angka(2,4,10))
     print(cek_angka(2,2,4))
```

b. Hasil Output

```
print(cek_angka(1,2,3))
print(cek_angka(2,4,10))
print(cek_angka(2,2,4))
```

```
True
False
False
```

c. Penjelasan

Menggunakan def fungsi dengan tiga parameter: a, b dan c, untuk mencari tahu apakah angka-angka memenuhi kedua kriteria:

1. Semua nilai bilangan berbeda: Menggunakan operator tidak sama dengan (!=). Jadi a tidak sama dengan b tidak sama dengan c :

   a != b != c

2. Dua dari parameter dijumlahkan hasilnya parameter sisa: Menggunakan operator or untuk mengidentifikasi ketika hanya salah satu yang memenuhi:

   a + b == c or b + c == a or c + a == b

Kedua kondisi digunakan dalam sebuah if dan digabung dengan operator and: kondisi sama semua and dua ditambah = bilangan sisa. Sehingga if hanya akan mengembalikan True ketika kedua kondisi terpenuhi dan menuju else ketika kriteria tidak terpenuhi. Ketika kondisi tidak terpenuhi else akan mengembalikan False. Diberikan beberapa test case dengan cara memanggil fungsi dan print.

SOAL 4.2

Membuat sebuah fungsi dengan nama cek_digit_belakang dengab tiga parameter input dari pengguna. Fungi menghasilkan True ketika minimal dua bilangan memiliki digit terakhir yang sama. Kemudian jalankan program dengan input beberapa test case berikut.:

- 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

a. Source Code

```
def cek_digit_belakang(a,b,c):
  digit_a = a % 10
  digit_b = b % 10
  digit_c = c % 10

  if digit_a == digit_b  or digit_a == digit_c and digit_b == digit_c:
    return True
  else:
    return False

a=int(input("Masukkan a= "))
b=int(input("Masukkan b= "))
c=int(input("Masukkan c= "))

print(cek_digit_belakang(a,b,c))
```

b. Hasil Output

```
Masukkan a= 1024
Masukkan b= 14
Masukkan c= 94
True
```

c. Penjelasan

Untuk mencari digit terakhir atau satuan dari ketiga bilangan digunakan operator modulo 10. Hasil dari modulo 10 masing-masing bilangan kemudian ditampung dalam variabel digit_a, digit_b dan digit_c. Karena minimal dua bilangan yang memiliki dua digit sama, maka kita dapat menggunakan kondisi ketika digit pertama sama dengan digit kedua: digit_a == digit_b or digit_b == digit_c or digit_c == digit_a.

Ketiga kondisi dihubungkan dengan operator or agar terpenuhi ketika salah satu kondisi betul. Kita tidak menambahkan kondisi ketika ketiga digit bilangan sama karena ketika dua digit terpenuhi otomatis jika ketiganya juga terpenuhi. Ketika kondisi tidak terpenuhi maka program akan mengeluarkan False. Setelah fungsi dibuat, tambahkan fungsi input dan print untuk pengguna memasukkan masing-masing bilangan a, b, dan c. Kemudian tambahkan fungsi print untuk memanggil fungsi cek_digit_belakang.

SOAL 4.3

Fungsi untuk mengonversi suhu menggunakan lambda function. Fungsi mengonversi suhu dari satuan Celcius menjadi Reamur atau Fahrenheit.

- $F = \left(\frac{9}{5} \times C\right) + 212$
- $R = 0.8 \times C$

Lakukan tes pada program menggunakan beberapa test case berikut:

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

a. Source Code

```
hitung = {
    "F" : lambda C: ((9/5)* C )+ 32,
    "R" : lambda C: (0.8*C)
}

C = int(input("Masukkan suhu dalam celcius= "))
konversi = input("Pilih konversi suhu (F atau R)= ")

print(round(hitung[konversi](C)))
```

b. Hasil Output

```
Masukkan suhu dalam celcius= 100
Pilih konversi suhu (F atau R)= F
212
```

c. Penjelasan

Fungsi lambda diberi nama hitung. Digunakan array untuk memasukkan kedua rumus konversi berdasarkan pilihan suhu pengguna. Pilihan suhu diwakilkan dengan “F: untuk Fahrenheit dan “R” untuk Reamur. Kemudian variabel C digunakan untuk menyimpan input suhu Celcius dari pengguna. Digunakan satu variabel lagi, konversi, untuk menyimpan input pilihan tipe suhu yang ingin dikonversi. Terakhir, fungsi lambda hitung dipanggil dan ditampilkan menggunakan print.

Link Github Repository:

https://github.com/rainiefch/Praktikum-Algoritma-Pemrograman_71230982.git