Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	**71231023**
Nama Lengkap	**YEHEZKIEL DARREN PUTRA WARDOYO**
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%)

Fungsi, Argumen, Parameter

Kita saat membuat program pasti memerlukan barisan-barisan kode. Ketika suatu program terdiri dari banyak sekali kode, anda perlu mengelompokkan kode-kode tersebut dalam suatu tempat agar sewatku-waktu kode tersebut bisa dipanggil kembali. Tempat tersebut dinamakan fungsi. Fungsi dalam Python dibagi menjadi 2, yaitu *built-in* (fungsi bawaan Python) dan fungsi yang dibuat oleh programmer.

Fungsi *built-in*, antara lain : abs(), aiter(), all(), anext(), any(), ascii(), bin(), bool(), breakpoint(), bytearray(), bytes(), callable(), chr(), classmethod(), compile(), complex(), delattr(), dict(), dir(), divmod(), enumerate(), eval(), exec(), filter(), float(), format(), frozenset(), getattr(), globals(), hasattr(), hash(), help(), hex(), id(), input(), int(), isinstance(), issubclass(), iter(), len(), list(), locals(), map(), max(), memoryview(), min(), next(), object(), oct(), open(), ord(), pow(), print(), range(), repr(), reversed(), round(), set(), setattr(), slice(), sorted(), staticmethod(), str(), sum(), super(), property(), tuple(), type(), vars(), zip(), __import__(). Fungsi-fungsi tersebut bisa langsung dipanggil di dalam suatu program. Selanjutnya, fungsi yang dibuat oleh programmer. Sebagai contoh, saya akan membuat suatu fungsi menentukan bilangan ganjil atau genap :

```
def odd(x):
  if x%2==0:
    print("Genap")
  else:
    print("Ganjil")
```

Dari fungsi diatas, ada beberapa hal yang perlu diperhatikan:

1. Diawali dengan type ”def” untuk memulai mendefiniskan suatu fungsi baru.
2. Terdapat nama fungsi dengan diikuti tanda kurung, odd(). Diakhiri dengan titik dua (:).
3. Dalam fungsi odd() membutuhkan satu argumen atau dikenal sebagai parameter, yakni x.
4. Isi dari fungsi tersebut harus menjorok 1 tab ke kanan.
5. Suatu fungsi untuk mengeluarkan nilai, maka perlu di kembalikan atau *return*. Jika dalam fungsi tersebut terdapat fungsi print() maka fungsi tersebut tidak perlu di *return*.

Suatu fungsi bisa memiliki parameter lebih dari dua. Perlu diingat, fungsi tersebut sewaktu-waktu bisa dipanggil kembali tanpa membuat fungsi baru yang sama.

Return Value

Seperti yang saya jelaskan di bagian pertama, sebuah fungsi bisa tidak mengembalikan nilai (atau yang bisa disebut *void function*) atau mengembalikan nilai. Suatu fungsi yang mengembalikan nilai harus menggunakan return sebagai pengembali nilai dari fungsi tersebut. Apabila suatu fungsi tidak menggunakan *return* maka fungsi tersebut ketika dijalankan akan menghasilkan nilai None. Sebagai contoh, ada suatu fungsi yang bernama panggil yang digunakan untuk menampilkan suatu pesan :

```
def panggil(pesan):
   print(pesan)
   print(pesan)

print(panggil("Hello World!"))
```

ketika program diatas dijalankan, maka akan menghasilkan :

```
Hello World!
Hello World!
None
```

Akan tetapi, jika suatu fungsi yang memiliki return didalamnya maka fungsi tersebut akan menampilkan hasil yang sesuai degan nilai yang di kembalikan. Contoh:

```
def pengurangan(a,b):
   return a-b

print(pengurangan(2,1))
```

Maka akan mengeluarkan output :
1

Oleh karena itu, *return* memiliki dua fungsi :

1. Mengembalikan nilai suatu fungsi
2. Mengakhiri suatu fungsi

Optional Argument dan Named Argument

Parameter atau argument dalam suatu fungsi terdapat dua jenis, yaitu optional argument dan named argument. Optional argument/parameter bersifat opsional dan memiliki nilai bawaan(default). Contohnya :

```
def tiket_masuk(harga_masuk,orang ,diskon=0):
    harga = harga_masuk*orang
    total  = harga-((harga*diskon)/100)
    print(diskon)
    return total

print(tiket_masuk(18000,5))
print(" ")
print(tiket_masuk(15000,3,20))
print(" ")
print(tiket_masuk(20000,4,15))
print(" ")
print(tiket_masuk(10000,7,2))
```

output:

```
0
90000.0

20
36000.0

15
68000.0

2
68600.0
```

Dari output diatas, fungsi akan menggunakan value *default* dari diskon ketika tidak diberi value baru. Ketika kita menambahkan value dari diskon maka nilai bawaan dari diskon akan berubah sesuai dengan value yang dimasukkan. Perlu diingat, urutan parameter sangat penting. Dahulukan parameter yang akan selalu berubah nilainya di paling kiri karena interpreter akan memproses mulai dari kiri.

Selanjutnya, named parameter/argument adalah pemanggilan fungsi dengan menyebutkan nama argumentnya. Sebagai contoh, kita masih menggunakan contoh yang sama dengan contoh sebelumnya.

```
def tiket_masuk(harga_masuk,orang ,diskon=0):
    harga = harga_masuk*orang
    total  = harga-((harga*diskon)/100)
    print(diskon)
    return total

print(tiket_masuk(orang=5,harga_masuk=18000))
print(" ")
print(tiket_masuk(orang=3,harga_masuk=15000,diskon=20))
print(" ")
print(tiket_masuk(harga_masuk=20000,diskon=15,orang=4))
print(" ")
print(tiket_masuk(diskon=2, harga_masuk=10000,orang=7))
```

Output:

```
0
90000.0

20
36000.0

15
68000.0

2
68600.0
```

Keuntungan menggunakan named parameter adalah tidak perlu pusing ketika program error akibat urutan parameter yang tidak sesuai karena kita telah memanggil paramater tersebut sebelum kita memberi suatu nilai di setiap parameter.

Anonymous Function (Lambda)

Biasanya sebuah fungsi harus di definisikan dengan nama, contohnya def tiket_masuk(), kita bisa membuat sebuah fungsi singkat layaknya ternary operator pada conditional if-else. Keyword untuk fungsi lambda adalah

variable = lambda parameter : code

Sebagai contoh, saya akan membuat fungsi tiket masuk seperti dicontoh sebelumnya namun saya buat menjadi bentuk lambda function.

```
tiket_masuk = lambda harga_masuk, orang, diskon :
(harga_masuk*orang)-(((harga_masuk*orang)*diskon)/100)
```

Dalam anonymous function di Python terdiri dari beberapa bagian berikut ini :

1. Keyword: lambda
2. Bound variable : argument/parameter pada function
3. Body : isinya ekspresi atau statement yang menghasilkan suatu nilai

Dalam lambda function kita bisa menerapkan optional parameter dan named parameter.

```
tiket_masuk = lambda harga_masuk, orang, diskon=20 :
(harga_masuk*orang)-(((harga_masuk*orang)*diskon)/100)
```

Optional parameter

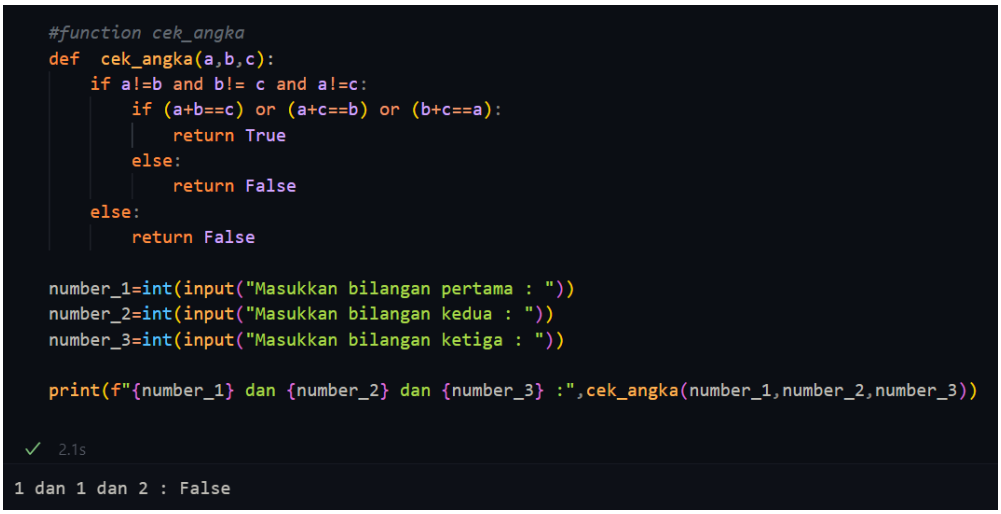
```
tiket_masuk = lambda harga_masuk, orang, diskon :
(harga_masuk*orang)-(((harga_masuk*orang)*diskon)/100)
print(tiket_masuk(harga_masuk=15000,diskon=20, orang=3))
```

Named parameter

BAGIAN 2: LATIHAN MANDIRI (60%)

Link Repo Github : https://github.com/YehezkielDarren/PrAlPro_Minggu4.git

SOAL 1



```python
#function cek_angka
def  cek_angka(a,b,c):
    if a!=b and b!= c and a!=c:
        if (a+b==c) or (a+c==b) or (b+c==a):
            return True
        else:
            return False
    else:
        return False

number_1=int(input("Masukkan bilangan pertama : "))
number_2=int(input("Masukkan bilangan kedua : "))
number_3=int(input("Masukkan bilangan ketiga : "))

print(f"{number_1} dan {number_2} dan {number_3} :",cek_angka(number_1,number_2,number_3))
```

✓ 2.1s

```
1 dan 1 dan 2 : False
```

Penjelasan :

Cara kerja program tersebut adalah dengan membandingkan 3 buah angka dengan 2 ketentuan sebagai berikut:

1. Nilai 3 parameter harus saling berbeda.
2. Dua parameter jika dijumlahkan akan menghasilkan nilai yang sama dengan parameter yang tersisa.

Bentuk program, sebagai berikut:

1. Kita buat 3 parameter, yaitu a,b, dan c.
2. Selanjutnya, gunakan kondisi pertama yaitu semua parameter tidak bernilai sama. Kondisi kedua; semua parameter bernilai sama.
3. Ketiga, jika kondisi pertama pada langkah 2 terpenuhi maka kita buat 2 kondisi lagi di dalam kondisi tersebut. Kondisi 1; jumlahkan 2 parameter yang diambil secara acak (bisa : a dan b, a dan c, b dan c), nilai penjumlahan harus sama dengan nilai parameter sisa (contoh : a+b==c, a+c==b, b+c==a). Catatan : menggunakan ”or” karena kondisi yang diminta adalah ”ada”. Kondisi 2; nilai penjumlahan 2 parameter yang diambil tidak sama dengan nilai parameter sisa.
4. Terakhir, jika kondisi 1 pada langkah 3 terpenuhi maka menghasilkan True. Akan tetapi, jika kondisi 2 pada langkah 3 yang terpenuhi maka menghasilkan False.

SOAL 2

```python
#function cek_digit_belakang()
def cek_digit_belakang(x,y,z):
    a= x%10
    b= y%10
    c=z%10
    if a==b==c:
        return True
    else:
        if a==b or b==c or a==c:
            return True
        else:
            return False

#stdin
firstNum=int(input("Masukkan bilangan pertama : "))
secNum=int(input("Masukkan bilangan kedua : "))
thirdNum=int(input("Masukkan bilangan ketiga : "))

#Test-case
print("30 dan 20 dan 18\t: ",cek_digit_belakang(30,20,18)) #output True
print("145 dan 5 dan 100\t: ",cek_digit_belakang(145,5,100)) #output True
print("71 dan 187 dan 18\t: ", cek_digit_belakang(71,187, 18)) #output False
print("1024 dan 14 dan 94\t: ",cek_digit_belakang(1024, 14,94)) #output True
print("53 dan 8900 dan 658\t: ",cek_digit_belakang(53, 8900, 658)) #output False
print("==============================")
#Stdout
print(f"{firstNum} dan {secNum} dan {thirdNum}\t: ",cek_digit_belakang(firstNum,secNum,thirdNum))
```

Output:

```
30 dan 20 dan 18          :  True
145 dan 5 dan 100         :  True
71 dan 187 dan 18         :  False
1024 dan 14 dan 94        :  True
53 dan 8900 dan 658       :  False
==============================
378495 dan 2143764 dan 2146587230174  :  True
```

Penjelasan:

1. Buat 3 parameter dalam fungsi tersebut; a, b, dan c
2. Kedua, ambil digit terakhir dari masing-masing parameter dengan menggunakan modulus 10.
3. Ketiga, bandingkanlah ketiga parameter dan buat 2 kondisi utama. Kondisi 1; ketiga parameter sama besar. Jika kondisi 1 terpenuhi maka menghasilkan nilai True.
4. Keempat, kondisi 2 ketika tidak terpenuhi semua nilai 3 parameter sama besar.jika kondisi 2 yang terpenuhi maka di buat 2 kondisi lagi, yaitu kondisi 1 ,ketika salah dua dari tiga parameter memiliki digit akhir yang sama, dan kondisi 2, ketika semua parameter memiliki nilai yang berbeda.
5. Jika kondisi 1 pada langkah 4 terpenuhi maka fungsi mengreturn True. Akan tetapi, jika kondisi 2 pada langkah 4 yang terpenuhi maka fungsi akan mengreturn False.

SOAL 3

```python
#Lambda function
suhu_reamur = lambda a=0 : 0.8*a
suhu_fahrenheit= lambda a=0 : (9/5)*a +32

#stdin
celci=float(input("Masukkan suhu dalam Celcius : "))

#testcase
print("Input C = 100. Output F = ",suhu_fahrenheit(100))
print("Input C = 80. Output F = ",suhu_reamur(80))
print("Input C = 0. Output F = ",suhu_fahrenheit())
print("="*10)
#stdout
print(f"Input C = {celci}. Output F = {suhu_fahrenheit(celci)}")
print(f"Input C = {celci}. Output R = {suhu_reamur(celci)}")
```

✓ 0.0s

```
Input C = 100. Output F =  212.0
Input C = 80. Output F =  64.0
Input C = 0. Output F =  32.0
==========
Input C = 65.3. Output F = 149.54
Input C = 65.3. Output R = 52.24
```

Penjelasan :

1. Kita membuat dua buah lambda function untuk menghitung suhu reamur dan fahrenheit dari suhu celcius.
2. Lambda function suhu_reamur berisi rumus konversi suhu celcius ke reamur. Sedangkan lambda function suhu_fahrenheit berisi rumus konversi suhu celcius ke fahrenheit. Saya ingin membuat default parameter dari setiap lambda sama dengan 0 agar saat ingin menampilkan suhu konversi dari 0 derajat celcius kita tidak perlu memasukkan nilai di parameter tersebut, cukup memanggil fungsinya saja.