



# Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

<b>NIM</b>	<b>71231050</b>
<b>Nama Lengkap</b>	<b>Samuel Natanael</b>
<b>Minggu ke / Materi</b>	<b>04 / Modular Programming</b>

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, Argument dan Parameter

Ada dua fungsi yang umum dipakai dalam Python yaitu **input()** dan **print()**

```
umur = int(input())
```

```
print(umur)
```

Program diatas adalah contoh penggunaan input dan print dimana program akan meminta input berupa angka yang kemudian akan diprint. Dua fungsi tersebut adalah fungsi yang berasal dari Python atau yang biasa disebut dengan *built-in function*. Setiap fungsi memiliki tujuan tertentu. Misalnya, fungsi `input()` berfungsi untuk membaca input yang diberikan oleh pengguna, dan fungsi `print()` menampilkan tulisan di layar.

Fungsi biasanya adalah kumpulan perintah yang digabungkan, memiliki tujuan dan fungsi tertentu dan dapat digunakan kembali. Apakah fungsi dan modular programming berhubungan? Jika Anda membuat aplikasi yang membutuhkan banyak langkah, Anda harus membagi banyak kode program menjadi bagian-bagian kecil atau blok untuk membuat program yang lebih besar. Istilah "modular" mengacu pada keadaan di mana program Anda terdiri dari sejumlah bagian modular yang masing-masing memiliki fungsi unik dan dapat digunakan kembali. Fungsi dibagi menjadi dua kategori berdasarkan asal-usulnya, yaitu

- Fungsi bawaan (*built-in function*). Berikut adalah list yang berisi fungsi bawaan Python

Built-in Functions				
<code>abs()</code>	<code>divmod()</code>	<code>input()</code>	<code>open()</code>	<code>staticmethod()</code>
<code>all()</code>	<code>enumerate()</code>	<code>int()</code>	<code>ord()</code>	<code>str()</code>
<code>any()</code>	<code>eval()</code>	<code>isinstance()</code>	<code>pow()</code>	<code>sum()</code>
<code>basestring()</code>	<code>execfile()</code>	<code>issubclass()</code>	<code>print()</code>	<code>super()</code>
<code>bin()</code>	<code>file()</code>	<code>iter()</code>	<code>property()</code>	<code>tuple()</code>
<code>bool()</code>	<code>filter()</code>	<code>len()</code>	<code>range()</code>	<code>type()</code>
<code>bytearray()</code>	<code>float()</code>	<code>list()</code>	<code>raw_input()</code>	<code>unichr()</code>
<code>callable()</code>	<code>format()</code>	<code>locals()</code>	<code>reduce()</code>	<code>unicode()</code>
<code>chr()</code>	<code>frozenset()</code>	<code>long()</code>	<code>reload()</code>	<code>vars()</code>
<code>classmethod()</code>	<code>getattr()</code>	<code>map()</code>	<code>repr()</code>	<code>xrange()</code>
<code>cmp()</code>	<code>globals()</code>	<code>max()</code>	<code>reversed()</code>	<code>zip()</code>
<code>compile()</code>	<code>hasattr()</code>	<code>memoryview()</code>	<code>round()</code>	<code>__import__()</code>
<code>complex()</code>	<code>hash()</code>	<code>min()</code>	<code>set()</code>	
<code>delattr()</code>	<code>help()</code>	<code>next()</code>	<code>setattr()</code>	
<code>dict()</code>	<code>hex()</code>	<code>object()</code>	<code>slice()</code>	
<code>dir()</code>	<code>id()</code>	<code>oct()</code>	<code>sorted()</code>	

Gambar 4.1 List Built-in Functions Python (Diambil dari

<https://stackoverflow.com/questions/8608587/finding-the-source-code-for-built-in-python-functions>)

- Fungsi yang dibuat sendiri oleh pengguna.

Contoh fungsi yang dibuat sendiri adalah fungsi dibawah ini berupa fungsi cek\_digit\_belakang untuk mengetahui digit belakang suatu bilangan :

```
def cek_digit_belakang(a):
```

```
    a = a % 10
```

```
    return a
```

Fungsi cek\_digit\_belakang ini mencakup beberapa hal yang perlu diperhatikan:

- "def" digunakan untuk membuat suatu fungsi.
- "cek\_digit\_belakang" adalah nama yang diberikan untuk memanggil fungsi tersebut.
- Isi dari fungsi harus di spasi 1 tab untuk sesuai dengan nama fungsi secara vertikal.
- Fungsi "cek\_digit\_belakang()" membutuhkan satu argument yaitu parameter a (fungsi bisa memakai lebih dari satu parameter).
- "return" digunakan untuk mengembalikan/mengeluarkan nilai fungsi yang dijalankan. Artinya setelah melakukan perhitungan ( a % 10 ) untuk mencari digit terakhir, return akan mengeluarkan nilai yang keluar dari fungsi tersebut.

Berikut adalah contoh penggunaan fungsi yang dibuat sendiri

Source code:

```
# Meminta input a dan b
a = int(input())
b = int(input())

# Pembuatan fungsi untuk memeriksa apakah b bisa dibagi a
def cek_pembagian(a, b):
    # Rumus untuk mencari hasil bagi
    x = b % a
    # Jika bisa dibagi
    if x == 0:
        return f"{b} bisa dibagi {a}"
    # Jika tidak bisa
    else:
        return f"{b} tidak bisa dibagi {a}"
print(cek_pembagian(a, b))
```

output :

```
PS D:\Ngampus\Semester 2\PrakAlPro> & D:/Python/python.exe "d:/Ngampus/Semester 2/PrakAlPro/4,2.py"
3
6
6 bisa dibagi 3
PS D:\Ngampus\Semester 2\PrakAlPro> & D:/Python/python.exe "d:/Ngampus/Semester 2/PrakAlPro/4,2.py"
7
3
3 tidak bisa dibagi 7
PS D:\Ngampus\Semester 2\PrakAlPro> █
```

### Optional Argument and Named Argument

Fungsi bisa memiliki parameter optional, yang merupakan parameter yang bersifat opsional dan memiliki nilai bawaan, atau default, yang telah ditetapkan sebelumnya. Untuk mendefinisikan parameter optional ini, pertama-tama Anda harus mendefinisikan nilai bawaannya, seperti yang ditunjukkan dalam ilustrasi berikut.

```
def harga_bensin(liter, jenis = 1):
    total = liter * jenis * 1000
    return total
```

Fungsi `harga_bensin()` memiliki dua parameter: `liter` dan `jenis`. Secara default, parameter `jenis` memiliki nilai 1, yang berarti jenis 1. Anda dapat memanggil fungsi ini dengan beberapa cara seperti ini:

```
def harga_bensin(liter, jenis = 1):
    total = liter * jenis * 1000
    return total

print(harga_bensin(1))
print(harga_bensin(10, 2))
print(harga_bensin(15, 3))
```

output:

```
1000
20000
45000
```

Pemanggilan pertama menggunakan satu argument saja, dan argument satu lagi memakai argument default yaitu jenis pertama. Sedangkan pemanggilan kedua dan ketiga menggunakan dua argument jadi tidak perlu menggunakan argument default.

Setiap parameter fungsi memiliki nama, jadi ketika Anda memanggil fungsi, Anda juga dapat memasukkan nama parameternya, tanpa harus mengikuti urutan yang diberikan.

Berikut adalah contoh penerapannya:

```
def games(a, b, c):  
    print("game satu :", a)  
    print("game dua :", b)  
    print("game tiga :", c)  
  
games("Valo", "ML", "Palworld")  
games(b="Valo", a="ML", c="Palworld")
```

Output yang keluar adalah sebagai berikut:

```
PS D:\Ngampus\Semester 2\PrakAIPro> & D:/Python/python.exe  
game satu : Valo  
game dua : ML  
game tiga : Palworld  
PS D:\Ngampus\Semester 2\PrakAIPro> & D:/Python/python.exe  
game satu : ML  
game dua : Valo  
game tiga : Palworld  
PS D:\Ngampus\Semester 2\PrakAIPro> █
```

Pemanggilan fungsi games() pertama dilakukan dengan argument yang berurutan. Pemanggilan kedua dilakukan dengan menyebutkan nama argumennya. Dengan metode ini, urutan argument tidak harus sama dengan urutan parameter pada fungsi.

## Anonymous Function (Lambda)

Fungsi anonim adalah fungsi tanpa nama, seperti namanya. Fungsi anonim dalam Python bukan fitur utama, melainkan fitur tambahan. Berbeda dengan bahasa pemrograman fungsional seperti Haskell, Lisp, dan Erlang, keyword lambda digunakan untuk mendefinisikan fungsi anonim dalam Python.

Sebagai contoh, berikut adalah contoh fungsi pangkat biasa:

```
def pangkat(a, b):
```

```
    hasil = a**b
```

```
    return hasil
```

```
print(pangkat(10,10))
```

output :

10000000000

Berikut adalah bentuk fungsi pangkat() setelah diubah menjadi lambda:

```
pangkat = lambda a, b: a**b
```

```
print(pangkat(10,10))
```

output:

10000000000

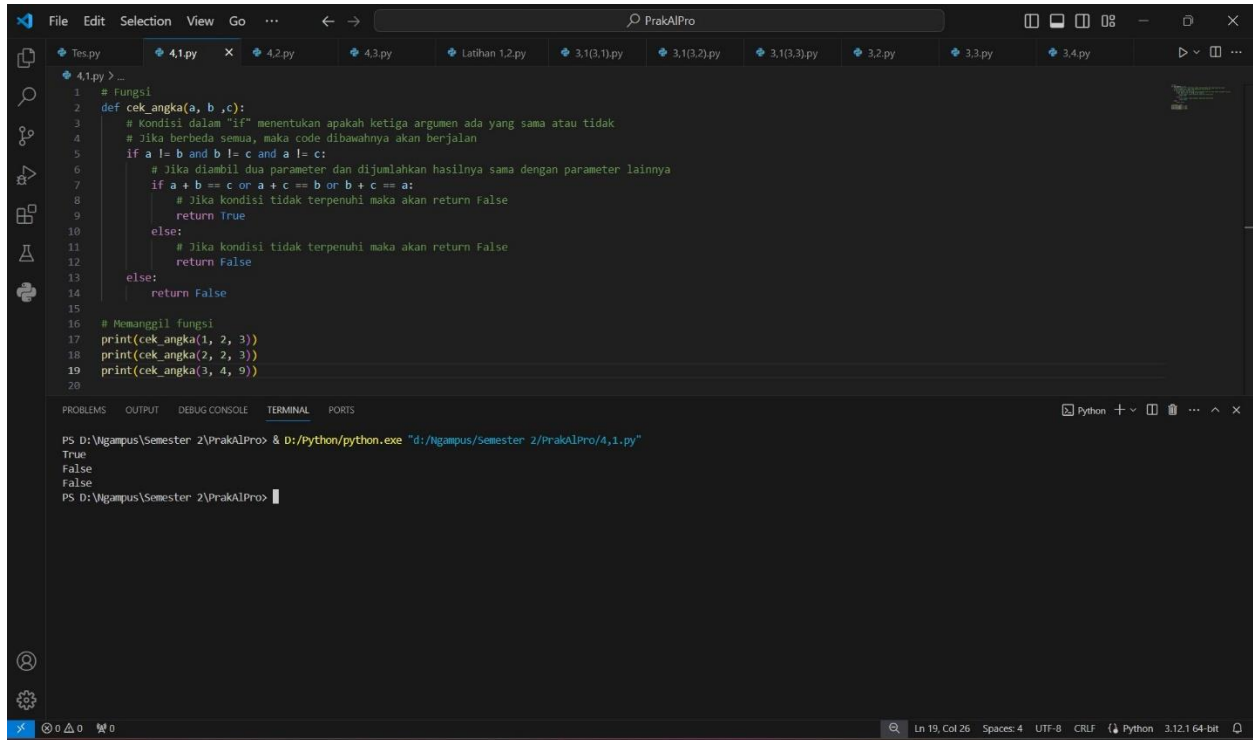
Fungsi lambda pada Python memiliki beberapa bagian seperti:

1. Keyword : Lambda
2. Bound variable: Argument pada fungsi lambda
3. Body : Bagian utama lambda, yang terdiri dari kata-kata atau pernyataan yang menghasilkan nilai.

## 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



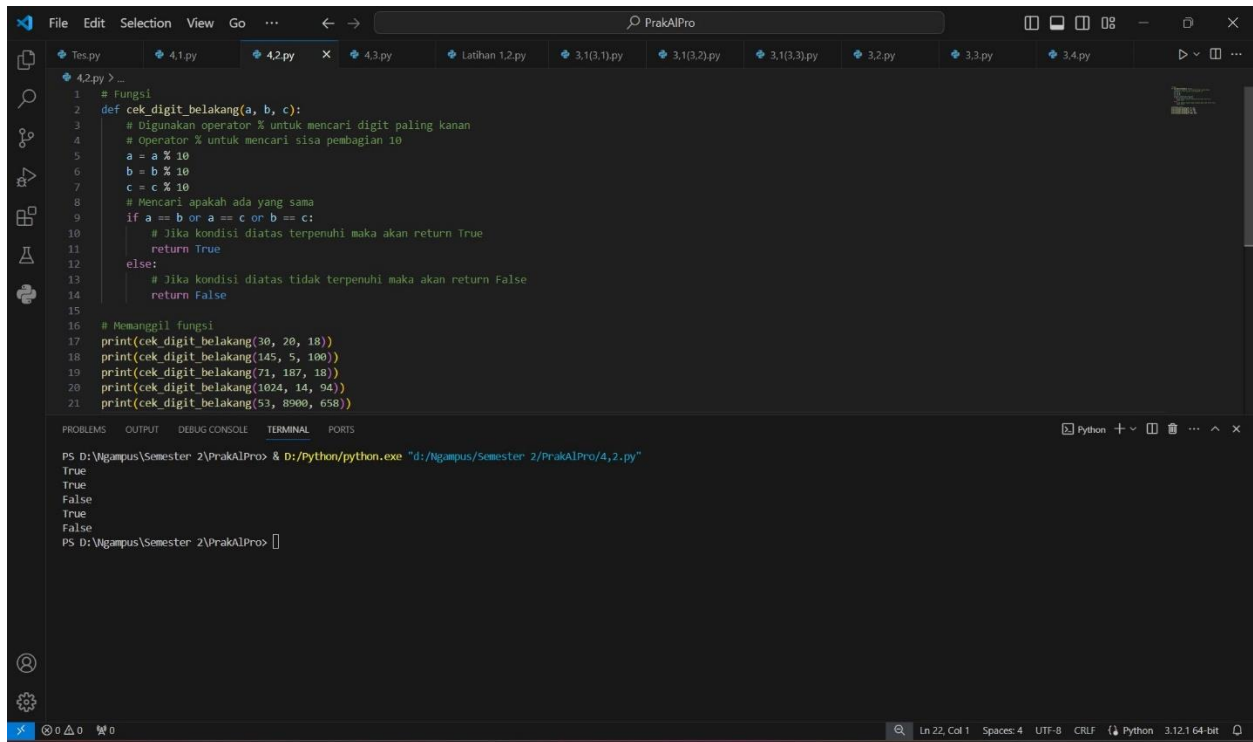
The screenshot shows a Visual Studio Code editor window with a Python file named '4,1.py'. The code defines a function 'cek\_angka(a, b, c)' that checks if three numbers are equal or if the sum of two equals the third. The function returns True if any condition is met, otherwise False. Below the function, three print statements are used to test the function with different inputs.

```
1 # Fungsi
2 def cek_angka(a, b, c):
3     # Kondisi dalam "if" menentukan apakah ketiga argumen ada yang sama atau tidak
4     # Jika berbeda semua, maka code dibawahnya akan berjalan
5     if a != b and b != c and a != c:
6         # Jika diambil dua parameter dan dijumlahkan hasilnya sama dengan parameter lainnya
7         if a + b == c or a + c == b or b + c == a:
8             # Jika kondisi tidak terpenuhi maka akan return False
9             return True
10        else:
11            # Jika kondisi tidak terpenuhi maka akan return False
12            return False
13    else:
14        return False
15
16 # Memanggil fungsi
17 print(cek_angka(1, 2, 3))
18 print(cek_angka(2, 2, 3))
19 print(cek_angka(3, 4, 9))
20
```

The terminal output shows the results of the function calls:

```
PS D:\Ngampus\Semester 2\PrakAlPro> & D:/Python/python.exe "d:/Ngampus/semester 2/PrakAlPro/4,1.py"
True
False
False
PS D:\Ngampus\Semester 2\PrakAlPro>
```

## SOAL 2

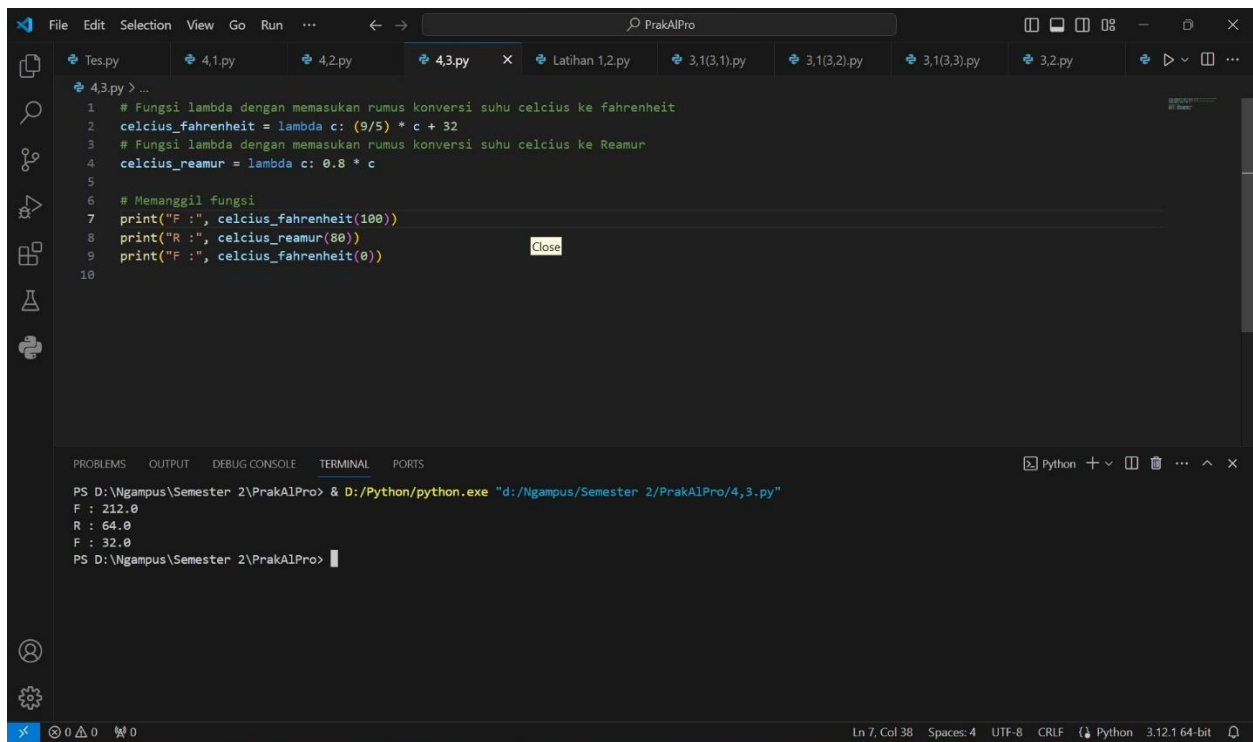


```
1 # Fungsi
2 def cek_digit_belakang(a, b, c):
3     # Digunakan operator % untuk mencari digit paling kanan
4     # Operator % untuk mencari sisa pembagian 10
5     a = a % 10
6     b = b % 10
7     c = c % 10
8     # Mencari apakah ada yang sama
9     if a == b or a == c or b == c:
10        # Jika kondisi diatas terpenuhi maka akan return True
11        return True
12    else:
13        # Jika kondisi diatas tidak terpenuhi maka akan return False
14        return False
15
16 # Memanggil fungsi
17 print(cek_digit_belakang(30, 20, 10))
18 print(cek_digit_belakang(145, 5, 100))
19 print(cek_digit_belakang(71, 187, 18))
20 print(cek_digit_belakang(1024, 44, 94))
21 print(cek_digit_belakang(53, 8900, 658))
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\Ngampus\Semester 2\PrakAIPro> & D:/Python/python.exe "d:/Ngampus/Semester 2/PrakAIPro/4,2.py"
True
True
False
True
False
PS D:\Ngampus\Semester 2\PrakAIPro>
```

## SOAL 3



```
1 # Fungsi lambda dengan memasukan rumus konversi suhu celcius ke fahrenheit
2 celcius_fahrenheit = lambda c: (9/5) * c + 32
3 # Fungsi lambda dengan memasukan rumus konversi suhu celcius ke Reamur
4 celcius_reamur = lambda c: 0.8 * c
5
6 # Memanggil fungsi
7 print("F :", celcius_fahrenheit(100))
8 print("R :", celcius_reamur(80))
9 print("F :", celcius_fahrenheit(0))
10
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\Ngampus\Semester 2\PrakAIPro> & D:/Python/python.exe "d:/Ngampus/Semester 2/PrakAIPro/4,3.py"
F : 212.0
R : 64.0
F : 32.0
PS D:\Ngampus\Semester 2\PrakAIPro>
```



Github : <https://github.com/SamuelN1508/PrakAIPro4.git>