


<p>Nama: (Yehezkiel Cavaro Sitanggang)</p> <p>NIM: (064102500021)</p>	 <p>UNIVERSITAS TRISAKTI</p> <p>Praktikum Algoritma & Pemrograman</p>	<p>MODUL 7 (V.1.0)</p> <p>Nama Dosen: Dr. Binti Solihah, S.T., M.Kom</p>
<p>Hari/Tanggal: Rabu 05 11 2025</p>		<p>Nama Asisten Labratorium:</p> <ol style="list-style-type: none"> 1. Almakius Felix Bariq Hekopung - 065002300016 2. Haidar Zhafiral Haqq - 065002400027

Fungsi pada Pemrograman Python - Latihan

1. Teori Singkat

Fungsi merupakan suatu bagian dari program yang dimaksudkan untuk mengerjakan suatu tugas tertentu dan letaknya terpisah dari program yang memanggilnya. Pada pembuatan program yang kompleks dan memiliki banyak fitur, kita diharuskan menggunakan fungsi. Fungsi diperlukan agar mempermudah kita dalam membaca sebuah kode program dan mempermudah untuk merawatnya. Dengan adanya sebuah fungsi kita tidak perlu menulis kode sepanjang gerbong kereta api di program utama dan kita dapat memecah atau membaginya lalu tinggal memanggil fungsinya saja didalam program utama. Dan itu juga akan mempersingkat penulisan dari program utama tersebut nantinya

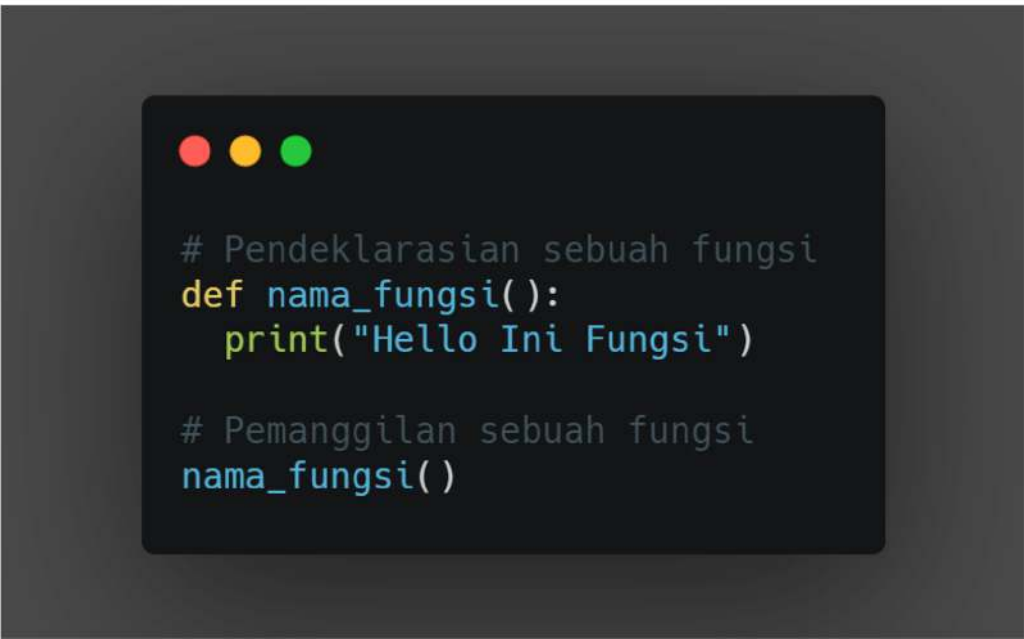


Fungsi memiliki parameter yang dapat dipergunakan untuk memasukkan atau menampung variabel kedalam sebuah fungsi. Dalam pendeklarasian parameter pada sebuah fungsi kita juga dapat memasukkan *default argument/parameter*.

Fungsi juga dapat mengembalikan nilai dengan cara menggunakan keyword *return* yang di taruh didalam blok program fungsi itu sendiri untuk mengembalikan nilai yang ingin kita kembalikan kedalam program utama.

Membuat Fungsi pada Python

Fungsi pada Python, dibuat dengan kata kunci *def* kemudian diikuti dengan nama fungsinya.



```
# Pendeklarasian sebuah fungsi
def nama_fungsi():
    print("Hello Ini Fungsi")

# Pemanggilan sebuah fungsi
nama_fungsi()
```

Fungsi dengan parameter pada Python

Memasukkan parameter kedalam sebuah fungsi



2. Alat dan Bahan

Hardware : Laptop/PC

Software : Spyder (Anaconda Python)

3. Elemen Kompetensi

Elemen Kompetensi I : Mahasiswa dapat menggunakan struktur percabangan berdasarkan data yang diberikan oleh user.

a. Latihan pertama

Buatlah sebuah program yang bisa menentukan apakah itu bilangan prima atau bukan. Gunakan 2 fungsi beserta implementasikan parameternya.

Contoh Output

```
Masukkan angka: 3
3 adalah bilangan Prima
> |
```

```
Masukkan angka: 4
4 bukanlah bilangan Prima
|
```

Source Code



```

Latihan praktikum algoritma 7.py > ...
1  def is_prime(number):
2      """
3      Fungsi untuk memeriksa apakah sebuah angka adalah bilangan prima.
4      Menggunakan parameter 'number'.
5      """
6      # Bilangan prima harus lebih besar dari 1
7      if number <= 1:
8          return False
9
10     # Periksa pembagian dari 2 hingga akar kuadrat dari angka
11     # Ini adalah optimasi, karena jika ada faktor yang lebih besar dari akar kuadrat,
12     # pasti ada faktor yang lebih kecil dari akar kuadrat.
13     for i in range(2, int(number**0.5) + 1):
14         if number % i == 0:
15             return False # Jika habis dibagi, itu bukan bilangan prima
16
17     return True # Jika tidak ditemukan pembagi, itu adalah bilangan prima
18
19  def check_number(num_input):
20      """
21      Fungsi untuk menampilkan hasil pemeriksaan bilangan prima.
22      Menggunakan parameter 'num_input'.
23      """
24      print(f"Masukkan angka: {num_input}")
25      if is_prime(num_input):
26          print(f"{num_input} adalah bilangan Prima")
27      else:
28          print(f"{num_input} bukanlah bilangan Prima")

```

Output

```

Masukkan angka: 3
3 adalah bilangan Prima
-----
Masukkan angka: 4
4 bukanlah bilangan Prima
-----
Masukkan angka lain untuk diperiksa: 7
Masukkan angka: 7

```

b. Latihan Kedua

Buatlah sebuah program yang bisa menampilkan output seperti berikut ini:

```

>>1 //Input
>>1st //output
>>2 //input

```



>>2nd //output.

>>13 //input

>>13rd //output

Dst. Menggunakan implementasi fungsi dengan parameter.

Contoh Output

```
Ordinal Number
ketik 0 untuk mengentikan program
masukkan angka: 1
(1, 'st')
masukkan angka: 2
(2, 'nd')
masukkan angka: 3
(3, 'rd')
masukkan angka: 0
(0, 'th')
terima kasih telah menggunakan program saya
```

Source Code



```

1  def get_ordinal_suffix(number):
2      """
3      Fungsi untuk menentukan sufiks (akhiran) bilangan ordinal (st, nd, rd, th).
4
5      Parameter:
6      number (int): Bilangan bulat yang akan dikonversi.
7
8      Mengembalikan:
9      str: Sufiks ordinal yang sesuai.
10     """
11     # Menangani angka khusus: 11, 12, 13 yang selalu berakhiran 'th'
12     if 10 <= number % 100 <= 20:
13         return 'th'
14
15     # Menentukan sufiks berdasarkan digit terakhir
16     last_digit = number % 10
17
18     if last_digit == 1:
19         return 'st'
20     elif last_digit == 2:
21         return 'nd'
22     elif last_digit == 3:
23         return 'rd'
24     else:
25         # Default untuk 0, 4, 5, 6, 7, 8, 9
26         return 'th'
27
28 def format_ordinal_number(number):
29     """

```

Output

```

Ordinal Number
ketik 0 untuk menghentikan program
masukkan angka: 10
(10, 'th')
masukkan angka: 20
(20, 'th')
masukkan angka: 0
(0, 'th')
terima kasih telah menggunakan program saya
PS D:\praktikum algoritma modul 4>

```

4. File Praktikum



Github Repository:

<https://github.com/yezzkie/Praktikum-algo-modul-7.git>



5. Soal Latihan Soal:

1. Jika sebuah deklarasi fungsi memiliki parameter a “*def nama_fungsi(a):*”, tetapi ketika dipanggil pada program utama kita mengisi fungsi tersebut dengan nama variabel b “*nama_fungsi(b)*” berbeda dengan nama parameter yang sebelumnya dideklarasikan, apakah program tetap akan berjalan dengan sesuai? dan jika sesuai mengapa demikian dan jika tidak mengapa demikian?
2. Deskripsikan serta narasikan jalannya alur source code program yang sebelumnya telah kalian buat pada Elemen Kompetensi Latihan Kedua!

Jawaban:

1. (Ya, program akan tetap berjalan dengan sesuai dan tidak akan ada masalah meskipun nama variabel yang digunakan saat memanggil fungsi (b) berbeda dengan nama parameter yang dideklarasikan dalam fungsi

2. Deskripsi dan Narasi Alur Program Ordinal Number

Program ini terbagi menjadi tiga bagian utama: dua fungsi penolong dan satu alur program utama.

1. Inisialisasi dan Fungsi Penolong

Narasi dimulai saat program dieksekusi:

1. Deklarasi Fungsi *get_ordinal_suffix(number)*:
 - o Tujuan: Fungsi ini dideklarasikan terlebih dahulu. Tugasnya adalah menentukan akhiran ('st', 'nd', 'rd', 'th') yang benar untuk bilangan yang diberikan.
 - o Logika Kunci: Ia memeriksa kasus khusus (angka 11, 12, 13 selalu menggunakan 'th'). Untuk kasus lainnya, ia melihat digit terakhir (*number % 10*) untuk menentukan sufiks yang tepat.
2. Deklarasi Fungsi *format_ordinal_number(number)*:
 - o Tujuan: Fungsi ini dideklarasikan setelahnya. Tugasnya adalah memformat angka dan sufiks yang didapat menjadi *string* dengan format yang diminta, yaitu (angka, 'suffix').
 - o Ketergantungan: Fungsi ini memanggil *get_ordinal_suffix(number)* untuk mendapatkan akhiran yang benar, menunjukkan implementasi parameter antar fungsi.

2. Alur Program Utama (Loop Interaktif)

Setelah fungsi siap, program beralih ke bagian utama, yang berada di dalam sebuah *loop* tak terbatas (*while True*):



1. Tampilan Awal: Program mencetak judul "Ordinal Number" dan instruksi "ketik 0 untuk menghentikan program".
 2. Menerima Input: Program masuk ke mode interaktif dan menampilkan masukkan angka: untuk menunggu input pengguna.
 3. Validasi Input (Try-Except): Input yang diterima dimasukkan ke dalam blok try untuk dicoba dikonversi menjadi bilangan bulat (int()).
 - *Jika gagal* (misalnya pengguna mengetik "lima"), blok except ValueError dijalankan, mencetak pesan kesalahan, dan *loop* berlanjut ke iterasi berikutnya.
 4. Pengecekan Kondisi Keluar: Program memeriksa nilai bilangan bulat (angka) yang baru saja dimasukkan:
 - Jika angka = 0:
 - Program memanggil format_ordinal_number(0) dan mencetak outputnya (0, 'th').
 - Program mencetak pesan penutup: "terima kasih telah menggunakan program saya".
 - Perintah break dijalankan, yang menghentikan *loop* while True. Program selesai.
 5. Pemrosesan Normal (Angka Selain 0):
 - Jika angka \neq 0:
 - Program memanggil fungsi format_ordinal_number(angka) dengan angka input sebagai parameter.
 - Fungsi ini menjalankan logika di dalamnya, mendapatkan sufiks, dan mengembalikan *string* yang diformat (contoh: untuk input 3, mengembalikan (3, 'rd')).
 - Program mencetak *string* hasil pemformatan tersebut ke layar. ■ *Loop* kembali ke awal, meminta masukan angka berikutnya.
-

6. Kesimpulan

- a. Dalam Python, ketelitian pada sintaks dan parameter fungsi print() (atau fungsi output lainnya) menentukan apakah program akan berhasil berjalan tanpa *error* dan apakah hasilnya sesuai dengan ekspektasi pengembang maupun pengguna.



7. Cek List (✓)

No	Elemen Kompetensi	Penyelesaian	
		Selesai	Tidak Selesai
1.	Menggunakan struktur percabangan berdasarkan data yang diberikan oleh user.	✓	

8. Formulir Umpan Balik

No	Elemen Kompetensi	Waktu Pengerjaan	Kriteria
1.	Menggunakan struktur percabangan berdasarkan data yang diberikan oleh user.	45 Menit	Baik

Keterangan:

1. Menarik
2. Baik
3. Cukup
4. Kurang

Menyetujui
Jakarta, 1 Juli 2024



Penanggung Jawab Praktikum (Anung



Ariwibowo, M.Kom)

**Kepala Laboratorium B.
Pemrograman**



(Anung B. Ariwibowo, M.Kom)

