



MODUL PRAKTIKUM PEMROGRAMAN 1



**PRODI S1 BISNIS DIGITAL
FAKULTAS REKAYASA INDUSTRI DAN DESAIN
IT TELKOM PURWOKERTO
2023**

Penyusun : Muhammad Eka Purbaya, S.T.,M.Eng, dkk
Editor : Miko Ardian

TIM PENYUSUN

Muhammad Eka Purbaya, S.T.,M.Eng (NIDN : 0627069203)

Haidar Fadhila Fiqa (NIM : 20110022)

Miko Ardian (NIM : 20110011)

Moh. Lutfi Fadhila (NIM : 21110018)

Muhammad Abdul Latief (NIM : 21110002)

Raafi Alhadi (NIM : 20110025)

EDITOR

Miko Ardian (NIM : 20110011)

KATA PENGANTAR

Segala puji bagi Allah SWT yang telah melimpahkan rahmat dan karunia-Nya sehingga Modul Praktikum Pemrograman 1 ini bisa diselesaikan. Modul Praktikum ini merupakan penunjang untuk mata kuliah Pemrograman 1 yang digunakan di Program Studi S1 Bisnis Digital., FRID, IT Telkom Purwokerto. Modul ini dirancang khusus untuk memfasilitasi pembelajaran pemrograman menggunakan bahasa Python, sebuah bahasa pemrograman yang sangat relevan dan penting dalam dunia bisnis digital saat ini.

Dalam modul praktikum ini, penulis berusaha memberikan pemahaman yang komprehensif tentang konsep dasar pemrograman dan bagaimana menerapkannya dengan efektif menggunakan bahasa Python. Buku ini akan membimbing pembaca dari tingkat pemula hingga pemahaman yang lebih mendalam tentang pemrograman, sehingga pembaca dapat mengembangkan keterampilan yang diperlukan.

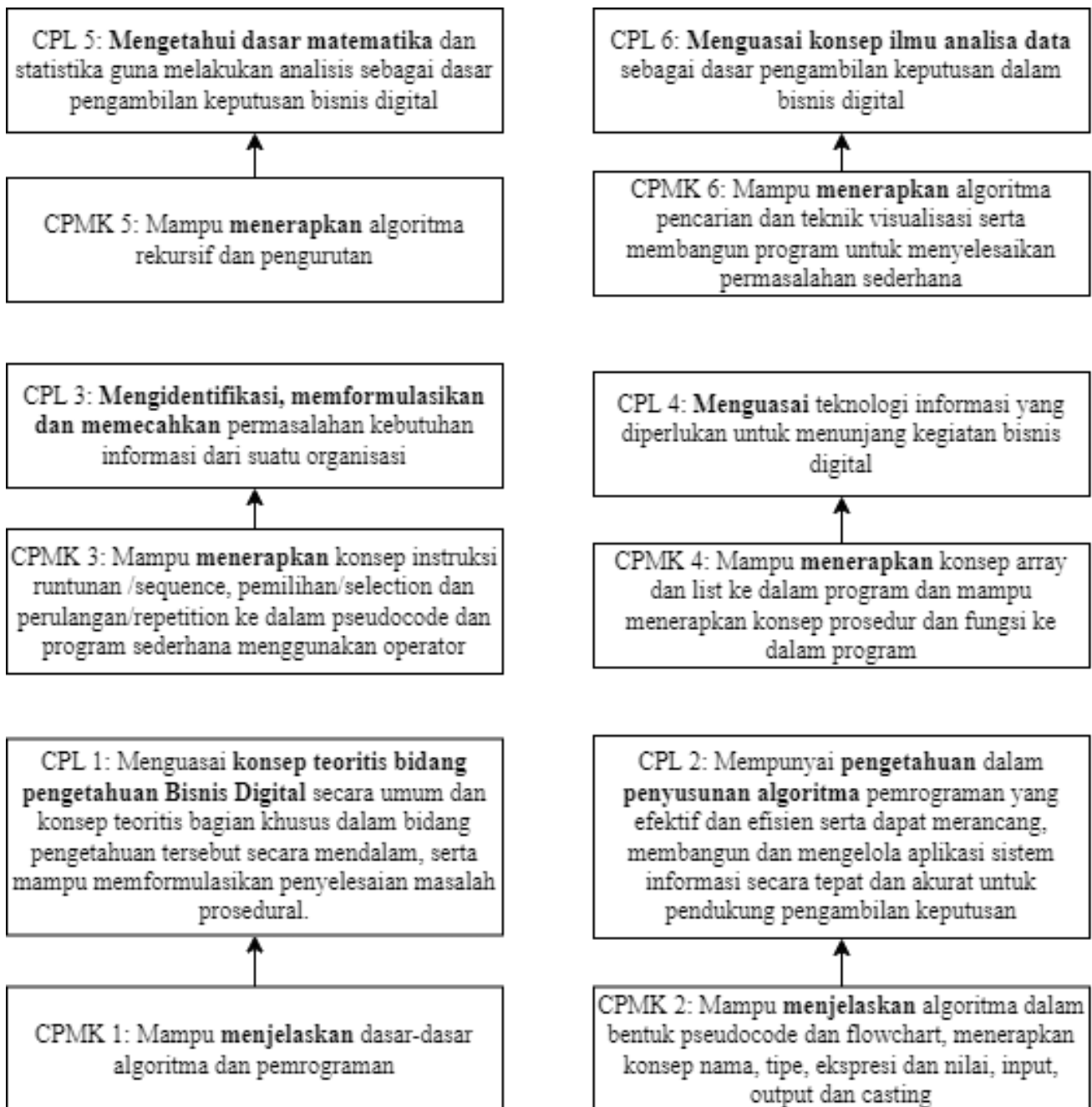
Modul Praktikum Pemrograman 1 ini terdiri dari 12 bab utama, yang dapat disesuaikan pembagiannya untuk perkuliahan selama 1 semester (14 pertemuan). Setiap bab terdiri dari tujuan pertemuan (setiap bab), dasar teori, latihan praktikum dan tugas mandiri. Kami berharap modul praktikum ini dapat menjadi panduan yang bermanfaat bagi pembaca di dunia pemrograman.

Kami juga mengharapkan saran, masukan dan kritik mengenai modul praktikum ini, sehingga dapat terus diperbaiki dan terus disesuaikan dengan kebutuhan penyelenggaraan mata kuliah Praktikum Pemrograman 1.

Purwokerto, 10 Oktober 2023

Tim Penulis

CAPAIAN PEMBELAJARAN MATA KULIAH PRAKTIKUM PEMROGRAMAN 1



DAFTAR ISI

| | |
|--|-----|
| KATA PENGANTAR | iii |
| CAPAIAN PEMBELAJARAN MATA KULIAH | iv |
| DAFTAR ISI | v |
| DAFTAR GAMBAR | vi |
| DAFTAR TABEL..... | xi |
| BAB I PENGANTAR PEMROGRAMAN KOMPUTER | 1 |
| BAB II TIPE DATA DAN VARIABEL | 6 |
| BAB III OPERASI DASAR INPUT DAN OUTPUT | 12 |
| BAB IV OPERATOR | 22 |
| BAB V SELECTION..... | 32 |
| BAB VI REPETITION | 37 |
| BAB VII ARRAY DAN LIST | 45 |
| BAB VIII FUNCTION DAN MODULES..... | 50 |
| BAB IX REKURSIF..... | 57 |
| BAB X SORTING | 66 |
| BAB XI SEARCHING | 73 |
| BAB XII VISUALIZATION | 78 |

DAFTAR GAMBAR

| | |
|--|----|
| Gambar 1. Ilustrasi algoritma secara sederhana | 2 |
| Gambar 2. Website python.org | 2 |
| Gambar 3. Download python 3.10.2 | 3 |
| Gambar 4. Print “hello world” | 3 |
| Gambar 5. Hasil coding | 3 |
| Gambar 6. Coding range angka ganjil | 4 |
| Gambar 7. Hasil coding | 4 |
| Gambar 8. Coding operasi matematika sederhana | 4 |
| Gambar 9. Hasil coding | 5 |
| Gambar 10. Code tipe data | 9 |
| Gambar 11. Code mencetak identitas | 9 |
| Gambar 12. Komentar dengan banyak baris | 9 |
| Gambar 13. Code untuk perintah for | 10 |
| Gambar 14. Indentasi pada python | 10 |
| Gambar 15. Code untuk konsep menukar isi gelas | 10 |
| Gambar 16. Print angka 5 | 14 |
| Gambar 17. Hasil coding | 14 |
| Gambar 18. Print angka dengan indentasi | 14 |
| Gambar 19. Hasil coding menunjukkan error | 14 |
| Gambar 20. Komentar dengan banyak baris | 15 |
| Gambar 21. Komentar dengan berbagai jenis simbol | 15 |
| Gambar 22. Membuat coding input | 15 |
| Gambar 23. Hasil coding | 16 |
| Gambar 24. Membuat coding input dengan separator | 16 |
| Gambar 25. Hasil coding | 16 |
| Gambar 26. Membuat new line | 17 |
| Gambar 27. Hasil coding | 17 |
| Gambar 28. Membuat input secara otomatis | 18 |
| Gambar 29. Hasil coding | 18 |
| Gambar 30. Membuat input untuk alamat | 18 |

| | |
|---|----|
| Gambar 31. Hasil coding..... | 18 |
| Gambar 32. Membuat tipe data | 19 |
| Gambar 33. Hasil coding..... | 19 |
| Gambar 34. Membuat input untuk operasi matematika | 19 |
| Gambar 35. Hasil coding..... | 20 |
| Gambar 36. Input untuk operasi matematika | 20 |
| Gambar 37. Membuat variabel “hasil” | 20 |
| Gambar 38. Hasil coding..... | 20 |
| Gambar 39. Coding operator aritmatika..... | 28 |
| Gambar 40. Coding operator perbandingan | 28 |
| Gambar 41. Coding operator penugasan 1 | 29 |
| Gambar 42. Coding operator penugasan 2 | 29 |
| Gambar 43. Coding operator logika..... | 30 |
| Gambar 44. Coding operator keanggotaan..... | 30 |
| Gambar 45. Coding operator identitas | 31 |
| Gambar 46. Coding if statement..... | 33 |
| Gambar 47. Coding if else statement | 33 |
| Gambar 48. Coding if elif else statement | 34 |
| Gambar 49. Coding if elif else statement | 34 |
| Gambar 50. Coding if nested statement | 34 |
| Gambar 51. Coding if elif else statement dengan input | 35 |
| Gambar 52. Coding if else statement dengan operator | 35 |
| Gambar 53. Flowchart tugas praktikum..... | 36 |
| Gambar 54. Coding repetition for..... | 38 |
| Gambar 55. Coding repetition for untuk range 1 | 38 |
| Gambar 56. Coding repetition for untuk range 2..... | 39 |
| Gambar 57. Coding repetition while..... | 39 |
| Gambar 58. Coding repetition while dengan if statement..... | 40 |
| Gambar 59. Coding repetition while dengan if statement dan operator | 40 |
| Gambar 60. Coding break statement..... | 40 |
| Gambar 61. Coding continue statement | 41 |
| Gambar 62. Coding pass statement..... | 41 |

| | |
|--|----|
| Gambar 63. Coding repetition for dengan list | 41 |
| Gambar 64. Coding repetition for dengan list dan if else statement..... | 42 |
| Gambar 65. Coding repetition while dengan if dan break statement..... | 42 |
| Gambar 66. Coding repetition while dan for..... | 43 |
| Gambar 67. Piramida dari hasil repetition | 43 |
| Gambar 68. Ilustrasi array | 45 |
| Gambar 69. Coding operasi pembagian pada array | 47 |
| Gambar 70. Coding operasi pembagian pada list | 47 |
| Gambar 71. Beberapa modifikasi pada list 1 | 47 |
| Gambar 72. Beberapa modifikasi pada list 2 | 48 |
| Gambar 73. Beberapa modifikasi pada list 3 | 48 |
| Gambar 74. Kombinasi list dengan repetition for serta statement if dan break | 48 |
| Gambar 75. Coding fungsi sederhana | 52 |
| Gambar 76. Coding fungsi yang berisi operasi aritmatika..... | 52 |
| Gambar 77. Coding fungsi yang memiliki argumen..... | 53 |
| Gambar 78. Coding fungsi lambda | 53 |
| Gambar 79. Coding fungsi yang berisi beberapa operator aritmatika | 54 |
| Gambar 80. Import modul..... | 54 |
| Gambar 81. Operator aritmatika menggunakan import modul | 55 |
| Gambar 82. Ilustrasi rekursif melalui cermin..... | 57 |
| Gambar 83. Ilustrasi rekursif melalui sampul buku..... | 58 |
| Gambar 84. Coding rekursif pada suatu fungsi | 58 |
| Gambar 85. Error pada coding yang dijalankan | 59 |
| Gambar 86. Coding repetition for..... | 59 |
| Gambar 87. Coding dasar repetition for..... | 59 |
| Gambar 88. Coding repetition for dengan batasan | 60 |
| Gambar 89. Coding repetition for dengan rekursif..... | 60 |
| Gambar 90. Coding rekursif untuk faktorial | 61 |
| Gambar 91. Coding rekursif untuk bilangan berpangkat | 61 |
| Gambar 92. Coding fungsi rekursif dengan list..... | 62 |
| Gambar 93. Coding fungsi fibonacci | 63 |
| Gambar 94. Coding persiapan variable untuk fungsi fibonacci | 63 |

| | |
|--|----|
| Gambar 95. Hasil coding fungsi fibonacci..... | 64 |
| Gambar 96. Coding fungsi inti dari Fibonacci | 65 |
| Gambar 97. Ilustrasi dari bubble sorting..... | 67 |
| Gambar 98. Ilustrasi dari selection sort..... | 67 |
| Gambar 99. Ilustrasi insertion sort..... | 68 |
| Gambar 100. Coding fungsi bubble sort | 68 |
| Gambar 101. List yang akan disorting | 68 |
| Gambar 102. Hasil coding..... | 69 |
| Gambar 103. Coding fungsi selection sort | 69 |
| Gambar 104. List yang akan disorting | 69 |
| Gambar 105. Hasil coding..... | 70 |
| Gambar 106. Coding fungsi insertion sort | 70 |
| Gambar 107. List yang akan disorting | 71 |
| Gambar 108. Hasil coding..... | 71 |
| Gambar 109. Ilustrasi linear searching | 74 |
| Gambar 110. Ilustrasi binary searching | 75 |
| Gambar 111. Coding linear searching tanpa fungsi..... | 76 |
| Gambar 112. Coding linear searching menggunakan fungsi | 76 |
| Gambar 113. Coding binary searching tanpa fungsi..... | 76 |
| Gambar 114. Coding binary searching menggunakan fungsi | 77 |
| Gambar 115. Contoh syntax untuk matplotlib | 79 |
| Gambar 116. Contoh sintac untuk seaborn | 79 |
| Gambar 117. Linea chart..... | 80 |
| Gambar 118. Scatter plot..... | 81 |
| Gambar 119. Bar plot..... | 81 |
| Gambar 120. Box plot..... | 82 |
| Gambar 121. Histogram | 82 |
| Gambar 122. Library yang diperlukan | 83 |
| Gambar 123. Visualisasi satu array | 83 |
| Gambar 124. Visualisasi dua array | 83 |
| Gambar 125. Menambahkan label pada visualisasi | 84 |
| Gambar 126. Menggunakan dataset yang berbeda pada satu visualisasi | 84 |

| | |
|---|----|
| Gambar 127. Library yang diperlukan dan membuat visualisasi | 84 |
| Gambar 128. Library yang diperlukan | 85 |
| Gambar 129. Persiapan dataset yang akan divisualisasikan..... | 85 |
| Gambar 130. Penyesuaian visualisasi..... | 85 |
| Gambar 131. Library yang diperlukan | 85 |
| Gambar 132. Persiapan dataset yang akan divisualisasikan..... | 85 |
| Gambar 133. Menampilkan hasil visualisasi..... | 85 |
| Gambar 134. Library yang diperlukan | 86 |
| Gambar 135. Persiapan dataset yang akan divisualisasikan..... | 86 |
| Gambar 136. Menampilkan hasil visualisasi..... | 86 |
| Gambar 137. Menampilkan hasil visualisasi..... | 86 |

DAFTAR TABEL

| | |
|--|----|
| Tabel 1. Beberapa tipe data | 6 |
| Tabel 2. Keyword yang sudah ada di python | 8 |
| Tabel 3. Fungsi-fungsi casting dan kegunaanya..... | 13 |
| Tabel 4. Jenis-jenis operator aritmatika | 22 |
| Tabel 5. Jenis-jenis operator perbandingan | 23 |
| Tabel 6. Jenis-jenis operator penugasan..... | 25 |
| Tabel 7. Jenis-jenis operator logika | 26 |
| Tabel 8. Jenis-jenis operator keanggotaan | 27 |
| Tabel 9. Jenis-jenis operator identitas..... | 27 |
| Tabel 10. Metode-metode pada list..... | 46 |
| Tabel 11. Function yang ada di list..... | 46 |
| Tabel 12. Daftar berat badan mahasiswa | 71 |
| Tabel 13. Dataset kasus covid-19 | 87 |

BAB I

PENGANTAR PEMROGRAMAN KOMPUTER

1.1 Tujuan Praktikum

- Mahasiswa dapat memahami konsep dasar algoritma.
- Mahasiswa dapat memahami konsep dasar pemrograman komputer.
- Mahasiswa dapat memahami konsep dasar pemrograman python.
- Mahasiswa dapat melakukan instalasi python secara mandiri.
- Mahasiswa dapat memahami konsep dasar pemrograman python.
- Mahasiswa dapat memahami cara membuat variable.

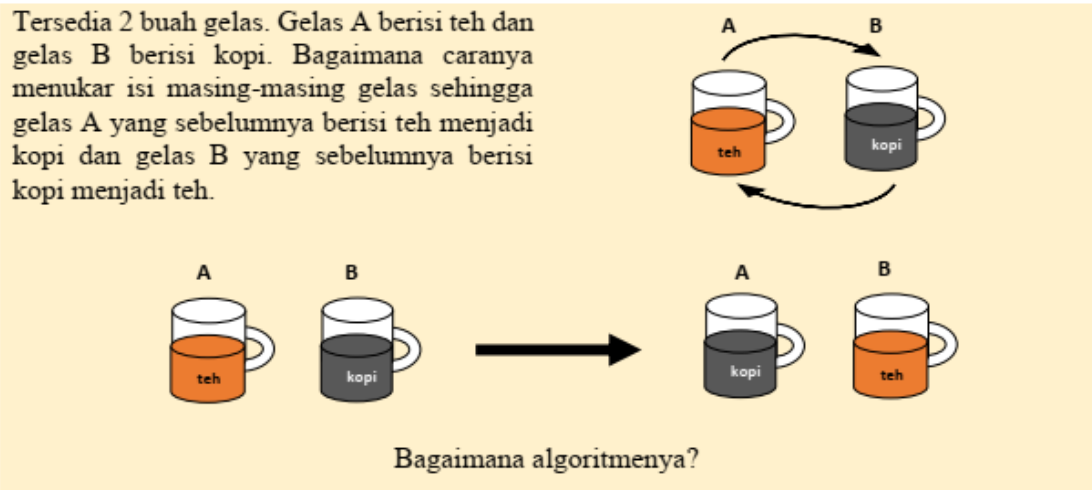
1.2 Alokasi Waktu

1 x Pertemuan = 50 Menit

1.3 Dasar Teori

1.3.1 Pengantar Algoritma

Sebelum membahas materi pemrograman komputer alangkah baiknya mengulang sedikit tentang konsep dasar algoritma. Algoritma menurut KBBI (Kamus Besar Bahasa Indonesia) adalah urutan logis pengambilan keputusan untuk pemecahan masalah. Menurut pendapat lain algoritma juga bisa diartikan sebagai langkah-langkah yang disusun secara tertulis dan berurutan untuk menyelesaikan suatu masalah atau bisa juga diartikan sebagai langkah-langkah logis penyelesaian masalah yang disusun secara sistematis. Berdasarkan beberapa pengertian atau definisi dari algoritma tersebut ada beberapa hal yang bisa disimpulkan yaitu sebuah algoritma harus logis atau masuk akal, kemudian disusun secara sistematis dalam bentuk urutan langkah-langkah yang digunakan untuk menyelesaikan suatu masalah. Algoritma bukan hanya diterapkan pada pemrograman tetapi juga bisa diterapkan dalam kehidupan sehari-hari. Berikut contoh gambaran sederhana terkait algoritma:



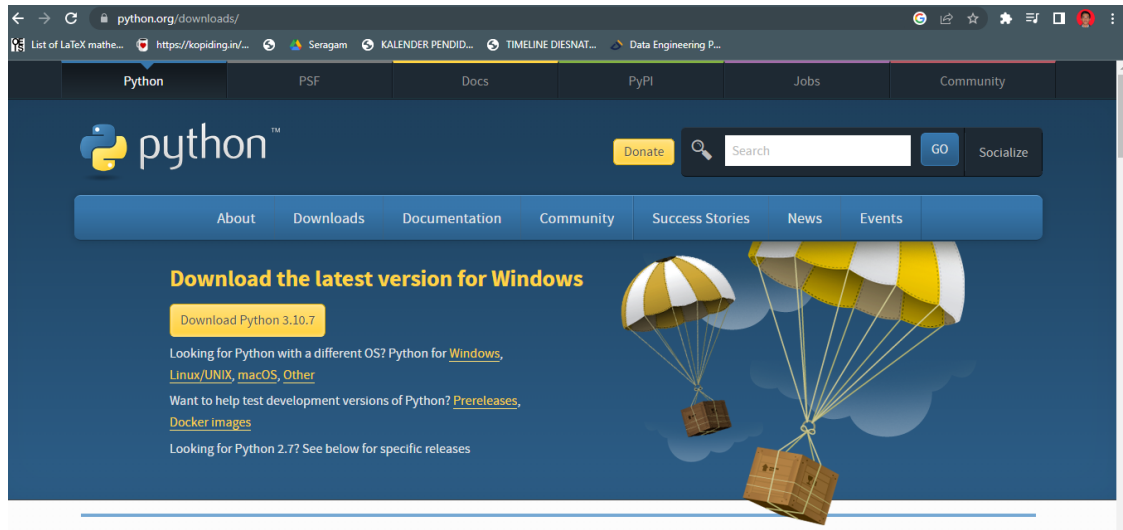
Gambar 1. Ilustrasi algoritma secara sederhana

1.4 Latihan Praktikum

1.4.1 Langkah-langkah instalasi python

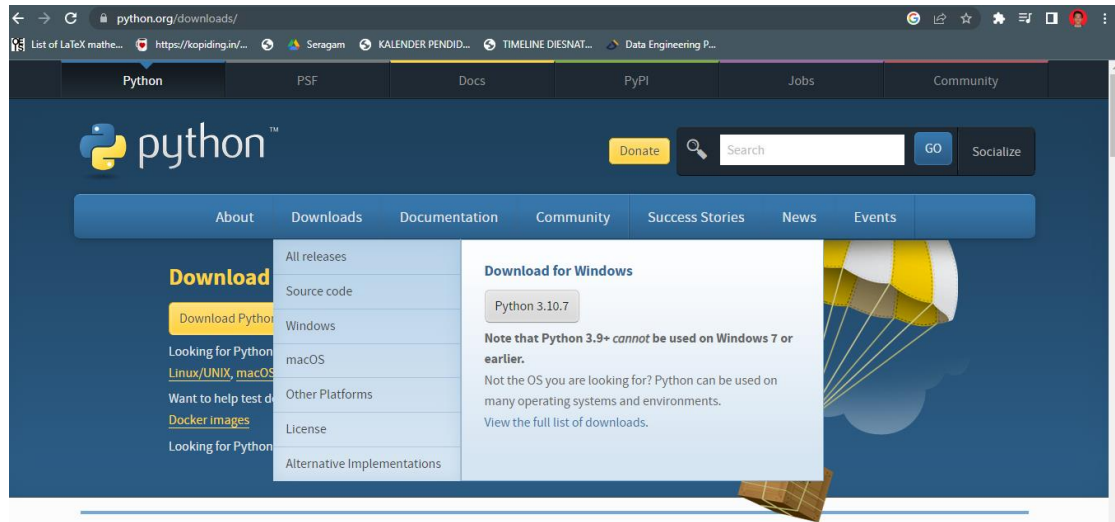
- Buka browser kemudian akses halaman website

<https://www.python.org/downloads>



Gambar 2. Website python.org

- Pilih menu download, kemudian klik Download Python 3.10.2 (versi bisa saja berbeda tergantung update)



Gambar 3. Download python 3.10.2

- Setelah proses download selesai, langsung install file **.exe** yang tadi sudah di download

1.4.2 Latihan 1

- Ketikkan kode program berikut pada Google Colab kemudian jalankan

```
1 print('Hello World')
```

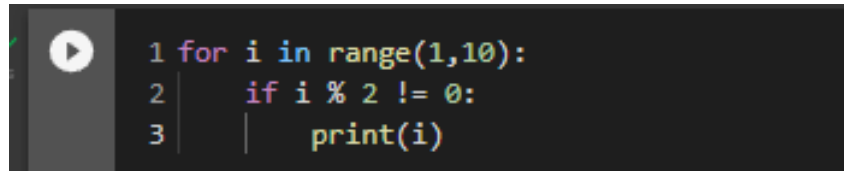
Gambar 4. Print "hello world"

- Pada gambar 4, ketika dijalankan memunculkan pesan Hello World. Perintah `print()` adalah sebuah fungsi yang digunakan untuk menampilkan sesuatu.
- Contoh di sini menampilkan kata Hello World yang sebelumnya diapit menggunakan petik ganda. Silahkan coba tampilkan kata lain menggunakan fungsi `print()` kemudian amati hasil yang muncul.

Gambar 5. Hasil coding

1.4.3 Latihan 2

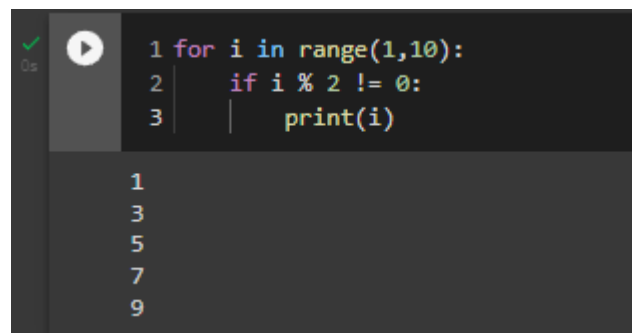
- Kita akan mencoba menampilkan angka ganjil dari 1-10. Ketikkan kode program berikut pada Google Colab kemudian jalankan



```
1 for i in range(1,10):
2     if i % 2 != 0:
3         print(i)
```

Gambar 6. Coding range angka ganjil

- Pada gambar 6, ketika dijalankan akan menampilkan angka ganjil dari 1-10, perintah for yang berarti melakukan perulangan untuk variable i dengan jarak 1 sampai 10. Setelah dilakukan perulangan ada perintah if untuk melakukan pengkondisian dan mencetak hanya angka ganjil saja (angka yang tidak habis dibagi 2). Setelah di print, akan memunculkan output seperti dibawah



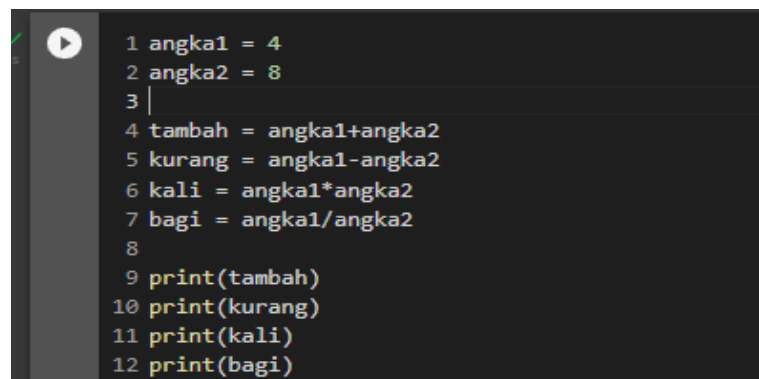
```
1 for i in range(1,10):
2     if i % 2 != 0:
3         print(i)
```

1
3
5
7
9

Gambar 7. Hasil coding

1.4.4 Latihan 3

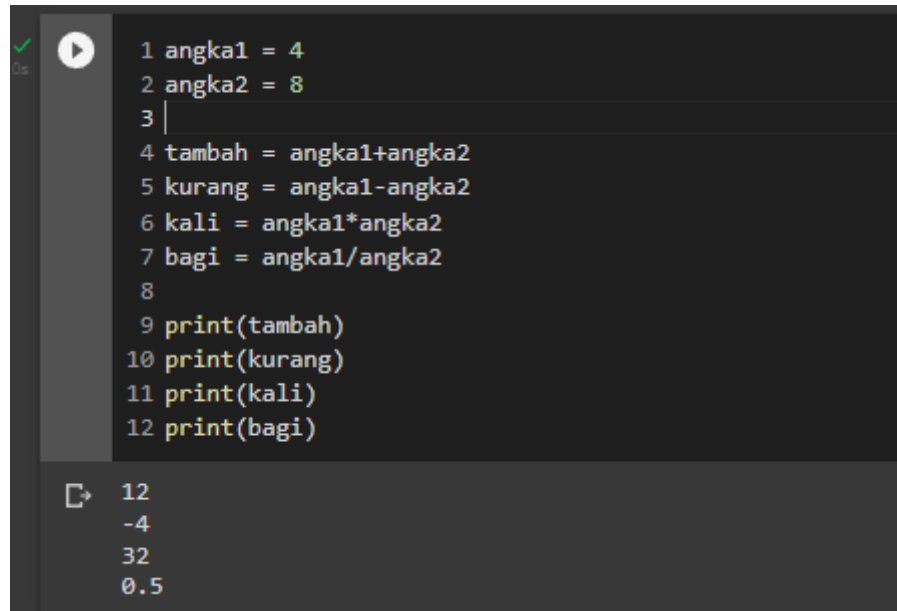
- Kita akan mencoba kembali membuat program operasi matematika menggunakan 2 variable yang sama. Ketikkan kode program dibawah ini menggunakan Google Colab lalu amatilah



```
1 angka1 = 4
2 angka2 = 8
3
4 tambah = angka1+angka2
5 kurang = angka1-angka2
6 kali = angka1*angka2
7 bagi = angka1/angka2
8
9 print(tambah)
10 print(kurang)
11 print(kali)
12 print(bagi)
```

Gambar 8. Coding operasi matematika sederhana

- Pada gambar 8, ketika dijalankan akan menampilkan hasil operasi matematika yang dilakukan. Sebelumnya kita tentukan terlebih dahulu angka1 dan angka2 lalu dijadikan variable, setelah itu kita buat variable lagi untuk menampung hasil operasi matematika yang dilakukan. Setelah di print dan dijalankan, akan memunculkan output seperti ini

A screenshot of a Python code editor with a dark background. The code is written in a light-colored font. It defines two variables, 'angka1' and 'angka2', with values 4 and 8 respectively. Then, it performs four arithmetic operations: addition ('tambah'), subtraction ('kurang'), multiplication ('kali'), and division ('bagi'). Each result is printed to the console. The output at the bottom shows the values 12, -4, 32, and 0.5 corresponding to the operations.

```
1 angka1 = 4
2 angka2 = 8
3
4 tambah = angka1+angka2
5 kurang = angka1-angka2
6 kali = angka1*angka2
7 bagi = angka1/angka2
8
9 print(tambah)
10 print(kurang)
11 print(kali)
12 print(bagi)
```

12
-4
32
0.5

Gambar 9. Hasil coding

1.5 Tugas Praktikum

1. Lakukan instalasi python pada perangkat komputer / laptop masing-masing
2. Hasil pekerjaan dituangkan dalam laporan praktikum
3. Format pengumpulan laporan praktikum : Tugas1-[NIM].pdf

BAB II

TIPE DATA DAN VARIABEL

2.1 Tujuan Praktikum

- Mahasiswa mampu memahami jenis dan tipe data
- Mahasiswa mampu membedakan jenis dan tipe data
- Mahasiswa dapat mendefinisikan tipe data
- Mahasiswa dapat membuat variabel

2.2 Alokasi Waktu

1 x Pertemuan = 50 Menit

2.3 Dasar Teori

2.3.1 Tipe data

Tipe data adalah jenis data yang tersimpan dalam sebuah variabel. Berikut tipe data yang ada pada python:

Tabel 1. Beberapa tipe data

| Tipe data | | Penjelasan | Contoh |
|-----------|------------------|--|--|
| Numbers | int (integer) | Terdiri dari bilangan bulat positif dan negatif | 12, 100 |
| | float | Terdiri dari bilangan pecahan atau desimal | 0.54, 3.14 |
| | complex | Terdiri dari bilangan kompleks atau bilangan imajiner | 1+2j |
| String | string | Terdiri dari karakter, kalimat, bisa berupa huruf, angka, dan lain-lain (diapit menggunakan single quote ‘/double quote “) | “Institut Teknologi Telkom Purwokerto”, ‘ITTP’, ‘F’ |

| | | | |
|------------|--------|---|-------------------------------|
| List | list | Tipe data yang bisa menyimpan berbagai macam tipe data dan sifat datanya mutable/bisa diubah-ubah | [‘abc’,123,3.21] |
| Tuples | tuples | Tipe data yang bisa menyimpan berbagai macam tipe data dan sifat datanya immutable/tidak bisa diubah-ubah | (‘abc’,123,3.21) |
| Dictionary | dict | Tipe data yang bisa menyimpan berbagai macam tipe data, sifat datanya mutable/bisa diubah-ubah, didefinisikan secara berpasangan menggunakan key dan value. | {‘nama’: Budi, ‘umur’ : 21} |
| Boolean | bool | Terdiri dari pernyataan True yang bernilai 1 dan False yang bernilai 0 | True atau False |

Untuk melihat jenis tipe data tertentu pada sebuah variabel bisa menggunakan fungsi **type()**.

2.3.2 Identifier

Identifier adalah pengenalan atau nama yang diberikan untuk mengidentifikasi sesuatu. Contoh identifier adalah variabel, fungsi, kelas dan lain sebagainya. Python memiliki aturan-aturan tertentu dalam penulisan identifier, aturan-aturan tersebut sebagai berikut:

- Identifier bersifat case sensitif artinya penulisan huruf besar dan huruf kecil dibedakan. Contoh penulisan variabel Nama dengan nama dianggap variabel yang berbeda.
- Karakter pertama pada identifier harus diawali huruf seperti (A-Z), (a-z) atau *underscore* (_). Contoh **nama_mahasiswa**, **Alamat_mahasiswa**.
- Identifier tidak boleh menggunakan karakter special atau simbol seperti (!, @, #, \$, %,., dan lain sebagainya)
- Identifier tidak boleh menggunakan reserved keyword (kata kunci yang sudah dipakai) oleh python, seperti:

Tabel 2. Keyword yang sudah ada di python

| | | | | | |
|----------|--------|---------|----------|--------|-------|
| and | def | False | import | Not | True |
| as | del | finally | in | Or | try |
| assert | elif | for | is | Pass | while |
| break | else | from | lambda | print | with |
| class | except | global | None | raise | yield |
| continue | exec | if | nonlocal | return | |

- Identifier tidak boleh dipisahkan menggunakan spasi contoh variabel nama mahasiswa
- Pemberian nama identifier sebaiknya disesuaikan dengan kebutuhan atau nama identifier sebaiknya bisa menjelaskan maksudnya. Contoh variabel nama_mahasiswa menjelaskan variabel tersebut akan diisi dengan nilai nama mahasiswa.

2.3.3 Variabel

Variabel adalah wadah atau tempat yang digunakan untuk menampung suatu nilai. Python termasuk dynamically-typed language artinya kita tidak perlu mendefinisikan atau mendeklarasikan tipe data terlebih dahulu sebelum membuat variabel. Variabel bisa diisi dengan nilai apapun dan python secara otomatis akan mendeteksi tipe data dari variabel yang kita buat berdasarkan nilai yang diberikan.

2.4 Latihan Praktikum

2.4.1 Mengenal tipe data

Tuliskan code pada interactive shell, amati dan pahami output yang ditampilkan.

```
print(5)
type(5)

print(3.14)
type(3.14)

print(1+2j)
type(1+2j)

print([1, 2, 3, 4])
type([1, 2, 3, 4])
print(['satu', 'dua', 'tiga', 'empat'])
type(['satu', 'dua', 'tiga', 'empat'])

print((1, 2, 3, 4))
type((1, 2, 3, 4))
print(('satu', 'dua', 'tiga', 'empat'))
type(('satu', 'dua', 'tiga', 'empat'))

print({'nama' : 'Budi', 'umur' : 20})
type({'nama' : 'Budi', 'umur' : 20})
```

Gambar 10. Code tipe data

2.4.2 Membuat variable

Tuliskan code pada interactive shell, amati dan pahami output yang ditampilkan.

```
Nama = "Arif"
Alamat = "Purwokerto, Jawa Tengah"
Pekerjaan = "Mahasiswa"

print>Nama)
print(Alamat)
print(Pekerjaan)
```

Gambar 11. Code mencetak identitas

Multiple line

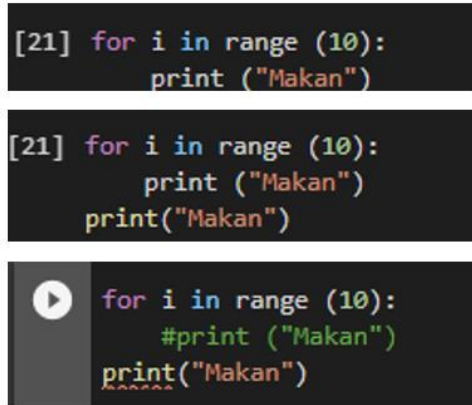
```
Banyak_baris = """
Saya
Berkuliah
di
Institut Teknologi Telkom Purwokerto
"""

print(Banyak_baris)
```

Gambar 12. Komentar dengan banyak baris

2.4.3 Memahami indentasi pada python

Tuliskan code berikut pada *interactive shell*, amati dan fahami penulisan indentasi pada python.

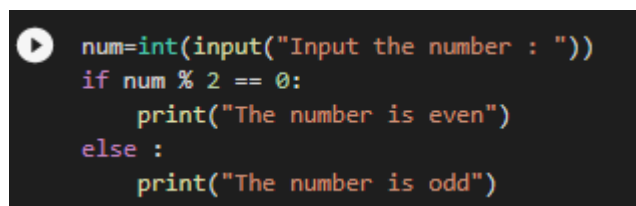


```
[21] for i in range (10):  
      print ("Makan")  
  
[21] for i in range (10):  
      print ("Makan")  
      print("Makan")  
  
▶ for i in range (10):  
    #print ("Makan")  
    print("Makan")
```

Gambar 13. Code untuk perintah for

2.4.4 Memahami indentasi pada python

Amati dan fahami output yang ditampilkan, perhatikan kembali konsep indentasi pada python.

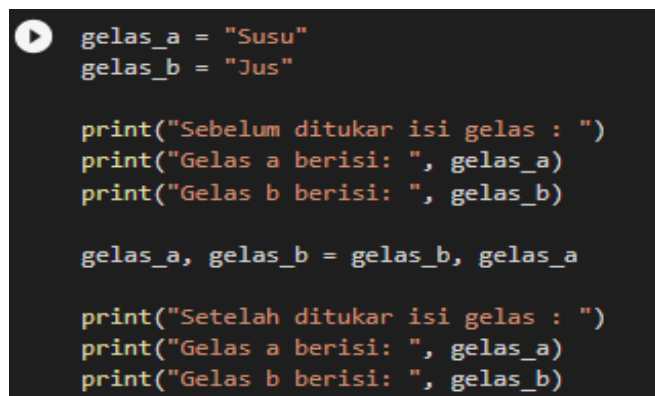


```
▶ num=int(input("Input the number : "))  
if num % 2 == 0:  
    print("The number is even")  
else :  
    print("The number is odd")
```

Gambar 14. Indentasi pada python

2.4.5 Memahami kembali konsep pembuatan variable dengan kasus menukar isi gelas

Amati dan fahami output yang ditampilkan, perhatikan kembali konsep pembuatan variabel pada python.



```
▶ gelas_a = "Susu"  
gelas_b = "Jus"  
  
print("Sebelum ditukar isi gelas : ")  
print("Gelas a berisi: ", gelas_a)  
print("Gelas b berisi: ", gelas_b)  
  
gelas_a, gelas_b = gelas_b, gelas_a  
  
print("Setelah ditukar isi gelas : ")  
print("Gelas a berisi: ", gelas_a)  
print("Gelas b berisi: ", gelas_b)
```

Gambar 15. Code untuk konsep menukar isi gelas

2.5 Tugas Praktikum

1. Buatlah beberapa variabel yang mendeskripsikan diri kalian dan tampilkan menggunakan fungsi *print()*, contoh:
 - Nama
 - NIM
 - Alamat
 - tanggal lahir
 - Jenis kelamin
 - umur
 - No. HP
2. Buatlah variabel yang berisi multiple line dengan isi menceritakan impian kalian setelah lulus dari IT Telkom Purwokerto, kemudian panggil variabel tersebut dengan fungsi *print()*

BAB III

OPERASI DASAR INPUT DAN OUTPUT

3.1 Tujuan Praktikum

- Mahasiswa mampu memahami operasi dasar input.
- Mahasiswa mampu memahami operasi dasar output.
- Mahasiswa mampu memahami cara *casting* sebuah nilai.
- Mahasiswa dapat membuat program sederhana untuk operasi input dan output.

3.2 Alokasi Waktu

1 x Pertemuan = 50 Menit

3.3 Dasar Teori

Sebelum masuk bahasan utama operasi dasar input output pada python, sebaiknya kita memahami kembali tentang *syntax* atau aturan dalam menuliskan kode yang benar pada python.

3.3.1 Indentasi

Bahasa pemrograman python sangat *strict* terhadap aturan dan tata cara penulisan kode, salah satunya adalah harus memperhatikan indentasi. Tidak seperti bahasa pemrograman lainnya, python tidak menggunakan kurung kurawal ({ }) untuk membatasi suatu blok kode maka indentasi menjadi sangat penting. Untuk penambahan indentasi tidak aturan baku yang terpenting adalah konsisten, tapi untuk rekomendasinya bisa menggunakan 4 spasi atau tab.

3.3.2 Komentar

Komentar adalah kode yang ada pada *script* python dan tidak kode tersebut tidak akan dieksekusi. Komentar biasanya digunakan untuk memberikan informasi tambahan terhadap script yang ditulis. Untuk menuliskan komentar diawali dengan menuliskan tanda pagar (#).

3.3.3 Python Input Output

Python menyediakan fungsi *built-in* (bawaan) yang bisa kita gunakan untuk proses input dan output. Secara tidak sadar sebenarnya kita sudah menggunakan

salah satu fungsi tersebut yaitu fungsi **print()**, fungsi **print()** pada python digunakan untuk melakukan operasi output sedangkan untuk operasi input pada python menggunakan fungsi **input()**.

➤ Fungsi output

Kita akan membahas kembali fungsi output yaitu **print()**, seperti yang sudah kita praktekan sebelumnya fungsi **print()** digunakan untuk mencetak atau menampilkan sesuatu pada layar.

➤ Fungsi input

Fungsi **input()** pada python digunakan untuk menerima inputan atau masukan dari *user* dan kemudian mengembalikannya dalam bentuk *string*.

Catatan: Segala bentuk inputan yang ditambahkan menggunakan fungsi input() tipe datanya akan dianggap sebagai string. Jadi ketika kita inputkan angka maka angka tersebut akan tetap dianggap sebagai string, untuk itu kita perlu melakukan yang namanya casting tipe data atau mengkonversi data yang diinputkan sesuai dengan apa yang kita harapkan

3.3.4 Casting Tipe Data

Casting tipe data adalah mengubah tipe data tertentu menjadi tipe data lainnya. Contoh kita mengubah tipe data *integer* menjadi tipe data *float*, supaya bisa berubah maka perlu menggunakan *casting*. Berikut fungsi *casting* yang bisa digunakan :

Tabel 3. Fungsi-fungsi casting dan kegunaanya

| Fungsi Casting | Keterangan |
|----------------|--|
| int() | Mengubah menjadi integer |
| float() | Mengubah menjadi float |
| bool() | Mengubah menjadi Boolean |
| chr() | Mengubah menjadi karakter |
| str() | Mengubah menjadi string |
| bin() | Mengubah menjadi bilangan biner |
| hex() | Mengubah menjadi bilangan heksadesimal |

| | |
|--------------------|---------------------------------|
| <code>oct()</code> | Mengubah menjadi bilangan oktal |
|--------------------|---------------------------------|

3.4 Latihan Praktikum

3.4.1 Latihan Indentasi

➤ Contoh 1 :

Tuliskan kode berikut pada *interactive shell*, amati dan fahami penulisan indentasi pada gambar dibawah ini :

```
print("5")
```

Gambar 16. Print angka 5

Hasil :

```
5
```

Gambar 17. Hasil coding

➤ Contoh 2 :

Tuliskan kode berikut pada *interactive shell*, amati dan pahami penulisan indentasi pada gambar dibawah ini :

```
print("5")
    print("7")
```

Gambar 18. Print angka dengan indentasi

Hasil :

```
File "<ipython-input-3-c37403e9cf0e>", line 2
    print("2")
    ^
IndentationError: unexpected indent
```

Gambar 19. Hasil coding menunjukkan error

Pada contoh yang kedua terdapat pesan *error* **SyntaxError: unexpected indent**, artinya terjadi kesalahan dalam pemberian indentasi. Ketika menggunakan IDLE python ini pesan *error* sangat jelas berada pada baris yang mana.

3.4.2 Latihan Komentar

➤ Contoh 1 :

Tuliskan kode berikut pada *interactive shell*, amati dan pahami penulisan indentasi pada gambar dibawah ini :

```
# Ini adalah komentar
# Komentar 1
# Komentar 2
# Komentar 3
```

Gambar 20. Komentar dengan banyak baris

Untuk lebih memahami penggunaan komentar perhatikan contoh 2 berikut :

➤ Contoh 2 :

Tuliskan kode berikut :

```
# Ini adalah komentar
# Komentar dalam baris program python tidak akan dieksekusi
# Apapun yang Anda tulis setelah tanda ini akan diabaikan oleh python

print("komentar 1 baris menggunakan tanda #")

"""jika ingin menggunakan komentar dengan banyak baris
bisa menggunakan petik 2
diawali petik 2 tiga kali
dan
diakhiri petik 2 tiga kali juga"""

print("komentar dengan banyak baris sekaligus bisa menggunakan petik 2 tiga kali")
```

Gambar 21. Komentar dengan berbagai jenis simbol

3.4.3 Latihan Output

➤ Contoh 1 :

Tuliskan kode berikut :

```
print(11,22,33)

print("Belajar membuat komputer dalam python")

prodi = "S1 Sains Data"
kampus = "Institut Teknologi Telkom Purwokerto"

print("Program Studi: ", prodi, kampus )
```

Gambar 22. Membuat coding input

Hasil :

```
11 22 33
Belajar membuat komputer dalam python
Program Studi: S1 Sains Data Institut Teknologi Telkom Purwokerto
```

Gambar 23. Hasil coding

Dari hasil contoh 1, fungsi **print()** digunakan untuk menampilkan atau mencetak sesuatu ke layar, bisa menggunakan satu *parameter* atau beberapa *parameter* sekaligus dengan menggunakan tanda koma (,) sebagai pemisah.

➤ Contoh 2 :

Selanjutnya kita akan belajar menggunakan *parameter* **sep=""** dan **end=""** pada fungsi **print()**.

```
# Tambahkan potongan kode berikut
# Menggunakan parameter sep= " " dan end=" "

print("Purwokerto", "Banyumas", "Jawa Tengah", sep= "-")
print("30", "Maret", "1950", sep= "/")
print("Purwokerto", "Banyumas", "Jawa Tengah", end= "****")
print("Republik", end=" ")
print("Indonesia")
```

Gambar 24. Membuat coding input dengan separator

Hasil :

```
Purwokerto-Banyumas-Jawa Tengah
30/Maret/1950
Purwokerto Banyumas Jawa Tengah****Republik Indonesia
```

Gambar 25. Hasil coding

Dari hasil contoh 2, bisa dilihat cara menggunakan *parameter* **sep=""** dan **end=""**. Jika dijelaskan lebih detail lagi *parameter* **sep=""** adalah *separator* atau pemisah bisa dilihat ketika menggunakan **sep="-"** kata yang ada di dalam fungsi **print()** akan dipisah dengan tanda (-) dan *parameter* **end="****"** adalah *end value* atau nilai akhir akan diberikan tanda (***), secara *default* nilai akhir pada fungsi **print()** adalah *new line* (baris baru) tetapi disini kita menggunakan *parameter* **end=""** dan memberikan nilai akhir pada fungsi **print()** dengan ***. Kemudian *string* di bawahnya ikut menjadi satu baris dengan *string* **Purwokerto Banyumas Jawa Tengah** karena nilai akhir sudah berganti menjadi *** bukan lagi baris baru. Contoh penggunaan *parameter* **end=""** selanjutnya adalah menggabungkan dua buah *string* pada dua fungsi **print()** yang berbeda, pada

contoh di atas kita menggabungkan *string* **Republik** dengan *string* **Indonesia** menggunakan parameter **end=""** atau nilai akhirnya diisi dengan **spasi**.

➤ Contoh 3 :

Selanjutnya kita akan belajar menggunakan menggabungkan *parameter* **sep=""** dan **end=""** dalam satu fungsi **print()** serta belajar menggunakan *escape character new line* (**\n**).

```
print("Purwokerto", "Banyumas", "Jawa Tengah", sep="#", end="\n" )  
print("Republik Indonesia", end=" ")  
print("77 tahun")
```

Gambar 26. Membuat new line

Hasil :

```
Purwokerto#Banyumas#Jawa Tengah&  
Republik Indonesia 77 tahun
```

Gambar 27. Hasil coding

Dari hasil contoh 3, bisa dilihat penggunaan *parameter* **sep=""** dan **end=""** “secara bersamaan. Perlu kita ingat kembali *parameter* **end=""** akan mengganti baris baru sesuai dengan apa yang kita inputkan, contoh disini kita memasukkan simbol **&** artinya *string* akan diakhiri dengan simbol **&** dan seharusnya *string* **Republik Indonesia** di bawahnya akan ikut sejajar dengan *string* **Purwokerto Banyumas Jawa Tengah**, tapi disini *string* tersebut tetap membuat baris baru, ini disebabkan karena kita menambahkan *escape character* **\n**. *Escape character* adalah karakter yang diawali dengan tanda **** (baca: *backslash*) dan masing-masing *escape character* mempunyai makna sendiri. *Escape character* **\n** artinya **n** disini adalah **new line** atau baris baru, jadi setelah kita menambahkan simbol **&** pada akhir *string* kemudian kita menambahkan baris baru menggunakan **\n** dan bisa kita lihat hasilnya untuk *string* pada fungsi **print()** yang kedua tetap berada di bawah fungsi **print()** yang pertama. Pada fungsi **print()** yang kedua kita menambahkan *parameter* **end=""**, disini kita tambahkan *whitespace* atau spasi sehingga *string* **77 tahun** pada fungsi **print()** yang ketiga sejajar dengan *string* **Republik Indonesia** dan dipisah dengan spasi.

3.4.4 Latihan Input

➤ Contoh 1 :

Tuliskan kode berikut :

```
# Fungsi input tanpa prompt  
kota = input()  
print("Nama kota : ", kota)
```

Gambar 28. Membuat input secara otomatis

Hasil :

Ketika file tersebut dijalankan, pada *interactive shell* akan muncul *prompt* kursor kedap-kedip menunggu inputan dan ketika kita inputkan sesuatu kemudian tekan **enter** apa yang kita inputkan sebelumnya akan muncul di layar seperti berikut :

```
Purwokerto  
Nama kota : Purwokerto
```

Gambar 29. Hasil coding

Pada *prompt* yang muncul, saya mencoba memasukkan kata **Purwokerto** kemudian tampil seperti hasil di atas.

➤ Contoh 2 :

Menggunakan fungsi **input()** dengan menambahkan *prompt*.

```
# Fungsi input dengan prompt  
Alamat = input("Masukkan alamat domisili : ")  
print("Alamat domisili saya adalah: ", Alamat)
```

Gambar 30. Membuat input untuk alamat

Hasil :

Ketika dijalankan baris kode tersebut akan muncul *prompt* menunggu inputan “**Masukkan alamat domisili:** “ dan hasilnya seperti berikut :

```
Purwokerto  
Nama kota : Purwokerto  
Masukkan alamat domisili : Jl. DI Panjaitan No 128  
Alamat domisili saya adalah: Jl. DI Panjaitan No 128
```

Gambar 31. Hasil coding

3.4.5 Latihan Casting Tipe Data

➤ **Contoh 1 :**

Tuliskan kode berikut pada *interactive shell*, kemudian amati hasil yang ditampilkan :

```
print(float(60))  
print(int(5.8967))
```

Gambar 32. Membuat tipe data

Hasil :

```
60.0  
5
```

Gambar 33. Hasil coding

Pada hasil contoh 1, pertama kita melakukan *casting* terhadap bilangan 60 dengan menjadikan bilangan tersebut menjadi bilangan dengan tipe data *float*, perlu kita ingat kembali 60 merupakan bilangan dengan tipe data *integer*. Cara mengubah bilangan dengan tipe data *integer* menjadi *float* adalah dengan menambahkan fungsi **float()**, kemudian bilangannya dimasukkan ke dalam fungsi tersebut. Hasilnya adalah tipe data bilangan 60 sekarang menjadi tipe data *float* dengan melihat hasil yang ditampilkan yaitu **60.0**. Ingat kembali tipe data *float* bentuknya pecahan atau menggunakan desimal. Begitu juga sebaliknya contoh selanjutnya adalah mengubah bilangan *float* menjadi bilangan dengan tipe data *integer* menggunakan fungsi **int()**, setelah dijalankan hasilnya adalah bilangan yang tampil berupa bilangan bulat yaitu **5**.

➤ **Contoh 2 :**

Untuk membuktikannya kembali jika inputan yang dimasukkan menggunakan fungsi **input()** akan dianggap sebagai *string* perhatikanlah kode berikut :

```
# Operasi matematika sederhana penambahan dua bilangan  
angka1 = input("Masukkan angka pertama : ")  
angka2 = input("Masukkan angka kedua : ")  
hasil = angka1 + angka2  
print("Hasil perjumlahan", angka1, "dan", angka2, "adalah", hasil)
```

Gambar 34. Membuat input untuk operasi matematika

Hasil :

Ketika kode tersebut dijalankan hasilnya tidak sesuai dengan apa yang kita harapkan, hasilnya adalah sebagai berikut :

```
Masukkan angka pertama : 234
Masukkan angka kedua : 966
Hasil perjumlahan 234 dan 966 adalah 234966
```

Gambar 35. Hasil coding

Hasil yang ditampilkan adalah bukan 1200, ini disebabkan karena semua inputan dianggap sebagai *string*. Pada python ketika dua buah *string* atau lebih diberikan operasi penjumlahan menggunakan operator + maka *string* tersebut akan digabung (*concatenation*). Untuk itu agar hasil yang ditampilkan sesuai dengan apa yang diharapkan kita perlu melakukan *casting* terhadap inputan angka tersebut, misalkan disini kita akan *casting* menjadi tipe data *integer* sehingga bisa dilakukan operasi penjumlahan yang benar.

Ubahlah kode menjadi kode berikut :

```
# Operasi matematika sederhana penambahan dua bilangan
angka1 = int(input("Masukkan angka pertama : "))
angka2 = int(input("Masukkan angka kedua : "))
hasil = angka1 + angka2
print("Hasil perjumlahan", angka1, "dan", angka2, "adalah", hasil)
```

Gambar 36. Input untuk operasi matematika

Pada kode di atas, telah ditambahkan fungsi *casting* `int()` untuk menambahkan fungsi *casting* bisa dengan beberapa cara salah satunya seperti pada kode di atas atau bisa juga dengan menambahkan fungsi *casting* di variabel. Contoh seperti pada kode di bawah berikut:

```
hasil = int(angka1) + int(angka2)
```

Gambar 37. Membuat variabel "hasil"

Sehingga ketika dijalankan hasil dari operasi matematika sederhana sudah sesuai dengan apa yang diharapkan :

```
Masukkan angka pertama : 234
Masukkan angka kedua : 966
Hasil perjumlahan 234 dan 966 adalah 1200
```

Gambar 38. Hasil coding

3.5 Tugas Praktikum

1. Tuliskan kode di dalamnya sehingga hasil yang didapatkan seperti berikut :



2. Gunakan fungsi *input()* untuk memasukkan biodata dan gunakan *casting* tipe data yang benar. Data dari biodata sebagai berikut :

- Nama
- NIM
- Alamat
- Tanggal Lahir
- Jenis Kelamin
- Umur
- Berat Badan
- Tinggi Badan
- No. HP

3. Diketahui kurs rupiah dalam dollar sebagai berikut:

1 dollar = 15000 rupiah

Buatlah sebuah program konversi dari rupiah menjadi dollar, gunakan fungsi *input()* dan *casting*.