



Modul Praktikum

Algoritma dan Pemrograman

Jan Wantoro, S.T., M.Eng.

Irma Yuliana, S.T., M.M., M.Eng.

Arif Setiawan, S.Kom., M.Eng.

Program Studi Pendidikan Teknik Informatika

Fakultas Keguruan dan Ilmu Pendidikan Universitas Muhammadiyah Surakarta

Daftar Isi

BAB 1.....	7
<i>Pengenalan Algoritma</i>	7
1.1 Tujuan.....	7
1.2 Pengantar.....	7
1.2.1 Algoritma	7
1.2.2 Flowchart	8
1.2.3 Pseudocode	11
1.3 Kegiatan Praktikum	13
1.3.1 Kegiatan 1 : Menenal Aplikasi Drawio	13
1.3.2 Kegiatan 2: Membuat Flowchart	14
1.3.3 Kegiatan 3: Membuat Pseudocode	14
BAB 2.....	15
<i>Pengenalan Bahasa Python.....</i>	15
2.1 Tujuan.....	15
2.2 Pengantar.....	15
2.2.1 Bahasa Python.....	15
2.2.2 Librari pada Python	16
2.2.3 Bagaimana Python Bekerja.....	17
2.2.4 PyCharm	18
2.3 Kegiatan Praktikum	18
2.3.1 Kegiatan 1 : Menjalankan Python mode interaktif	18
2.3.2 Kegiatan 2 : Menjalankan Python menggunakan IDE PyCharm	20
2.3.3 Kegiatan 3 : Menjalankan Python menggunakan Repl.it	22
2.4 Tugas	24
BAB 3.....	25
<i>Struktur Dasar Bahasa Python</i>	25

3.1	Tujuan.....	25
3.2	Pengantar.....	25
3.2.1	Input, Output dan Variabel	26
3.2.2	Operator Assignment	28
3.2.3	Komentar	28
3.3	Kegiatan Praktikum	29
3.3.1	Kegiatan 1 : Penulisan kode Python	29
3.3.2	Kegiatan 2 : Penggunaan Input Output.....	30
3.3.3	Kegiatan 3 : Penggunaan Variabel.....	30
3.4	Tugas	31
BAB 4.....		32
<i>Tipe Data dan Operator</i>		32
4.1	Tujuan.....	32
4.2	Pengantar.....	32
4.2.1	Operator Aritmatika.....	32
4.2.2	Integer dan Float	33
4.2.3	String.....	34
4.2.4	Boolean	34
4.2.5	Konversi Tipe Data.....	35
4.3	Kegiatan Praktikum	36
4.3.1	Kegiatan 1 : Bekerja dengan tipe data integer	36
4.3.2	Kegiatan 2 : Bekerja dengan tipe data float.....	36
4.3.3	Kegiatan 3 : Bekerja dengan tipe data string	37
4.4	Tugas	37
BAB 5.....		38
<i>Operasi pada String.....</i>		38
5.1	Tujuan.....	38
5.2	Pengantar.....	38
5.2.1	Fungsi upper() dan lower()	38
5.2.2	Fungsi len()	39

5.2.3	Fungsi mengatur rerata teks.....	40
5.2.4	Fungsi join() dan split()	40
5.2.5	Fungsi index().....	41
5.2.6	Fungsi replace()	41
5.2.7	String formatting.....	42
5.3	Kegiatan Praktikum	42
5.3.1	Kegiatan 1 : Operasi pada String	42
5.3.2	Kegiatan 2 : String formating	43
5.3.3	Kegiatan 3 : Slicing pada String	43
5.4	Tugas.....	44
BAB 6.....	45
Struktur Data pada Python.....	45
6.1	Tujuan.....	45
6.2	Pengantar.....	45
6.2.1	List	45
6.2.2	Tuple	50
6.2.3	Set	51
6.2.4	Dictionaries.....	54
6.3	Kegiatan Praktikum	55
6.3.1	Kegiatan Praktikum 1 : Bekerja dengan list	55
6.3.2	Kegiatan Praktikum 2 : Bekerja dengan Tuple.....	56
6.3.3	Kegiatan Praktikum 3 : Bekerja dengan Set	56
6.3.4	Kegiatan Praktikum 4 : Bekerja dengan Dictionaries.....	57
6.4	Tugas.....	58
BAB 7.....	59
Percabangan.....	59
7.1	Tujuan.....	59
7.2	Pengantar.....	59
7.2.1	IF.....	59
7.2.2	IF.. ELSE	61

7.2.3	IF..ELIF..ELSE	62
7.2.4	IF Bersarang.....	63
7.2.5	Operator Logika pada IF.....	64
7.3	Kegiatan Praktikum	64
7.3.1	Kegiatan Praktikum 1 : Bekerja dengan IF ELSE	64
7.3.2	Kegiatan Praktikum 2 : Bekerja dengan IF..ELIF..ELSE.....	65
7.3.3	Kegiatan Praktikum 3 : Bekerja dengan IF Bersarang	65
7.3.4	Kegiatan Praktikum 4 : Bekerja dengan Operator Logika.....	66
7.4	Tugas	66
BAB 8.....		67
Perulangan		67
8.1	Tujuan.....	67
8.2	Pengantar.....	67
8.2.1	Perulangan dengan While	67
8.2.2	Perulangan dengan For	68
8.2.3	Perulangan dengan else.....	69
8.2.4	Break dan Continue	70
8.3	Kegiatan Praktikum	73
8.3.1	Kegiatan Praktikum 1 : Perulangan While	73
8.3.2	Kegiatan Praktikum 2 : Perulangan For.....	73
8.3.3	Kegiatan Praktikum 3 : Mencari bilangan prima	74
8.4	Tugas	74
BAB 9.....		75
Fungsi pada Python		75
9.1	Tujuan.....	75
9.2	Pengantar.....	75
9.2.1	Fungsi dengan parameter	76
9.2.2	Fungsi dengan multi parameter.....	76
9.2.3	Modul pada python	77
9.3	Kegiatan Praktikum	78

9.3.1	Kegiatan Praktikum 1 : Fungsi dengan parameter.....	78
9.3.2	Kegiatan Praktikum 2 : Fungsi dengan parameter list.....	78
9.3.3	Kegiatan Praktikum 3 : Modul pada python.....	78
9.4	Tugas	79
<i>BAB 10.....</i>	<i>80</i>	
<i>Operasi File pada Python</i>	<i>80</i>	
10.1	Tujuan.....	80
10.2	Pengantar.....	80
10.2.1	Membuka file pada python	81
10.2.2	Membaca file pada python.....	82
10.2.3	Menulis file pada python	82
10.2.4	Menutup file pada python	83
10.3	Kegiatan Praktikum	84
10.3.1	Kegiatan Praktikum 1 : Membaca dan menulis File	84
10.3.2	Kegiatan Praktikum 2 : Menambah data pada file.....	84
10.3.3	Kegiatan Praktikum 3 : Attribute pada file	85
10.4	Tugas	85
<i>DAFTAR PUSTAKA.....</i>	<i>86</i>	
<i>LAMPIRAN.....</i>	<i>87</i>	

BAB 1

Pengenalan Algoritma

1.1 Tujuan

1. Dapat menjelaskan dan menggunakan Algoritma, Flowchart dan Pseudocode dalam membuat program
2. Mengenal lingkungan dan perlengkapan untuk membuat Algoritma, Flowchart dan Pseudocode

1.2 Pengantar

1.2.1 Algoritma

Algoritma adalah sekumpulan instruksi yang merupakan penyelesaian masalah itu dinamakan program. Agar program dapat dilaksanakan oleh komputer, program tersebut harus ditulis dalam suatu bahasa yang dimengerti oleh komputer. Bahasa komputer yang digunakan dalam menulis program dinamakan bahasa pemrograman. Urutan langkahlangkah yang sistematis untuk menyelesaikan sebuah masalah

dinamakan algoritma. Jadi algoritma adalah urutan logis pengambilan keputusan untuk pemecahan masalah. Kata logis merupakan kata kunci. Langkah-langkah tersebut harus logis, ini berarti nilai kebenarannya harus dapat ditentukan, benar atau salah. Urutan langkah-langkah yang sistematis dan logis untuk menyelesaikan suatu permasalahan.









Ciri-ciri algoritma:

- Langkah tersebut akan berhenti dan benar
- Mempunyai bentuk yang sederhana sehingga efektif
- Langkah-langkahnya jelas dan pasti

1.2.2 Flowchart











Kadang-kadang perlu digambarkan bagaimana arus data dari algoritma yang sudah dibuat, terutama kalau algoritma sudah cukup kompleks. Untuk itu algoritma dapat disajikan dalam bentuk flowchart (diagram alir). Untuk membantu memahami nalar suatu program digunakan grafik/symbol yang mengekspresikan kegiatan-kegiatan dalam sebuah program.

Tabel 1. 1 Simbol Dasar Flowchart

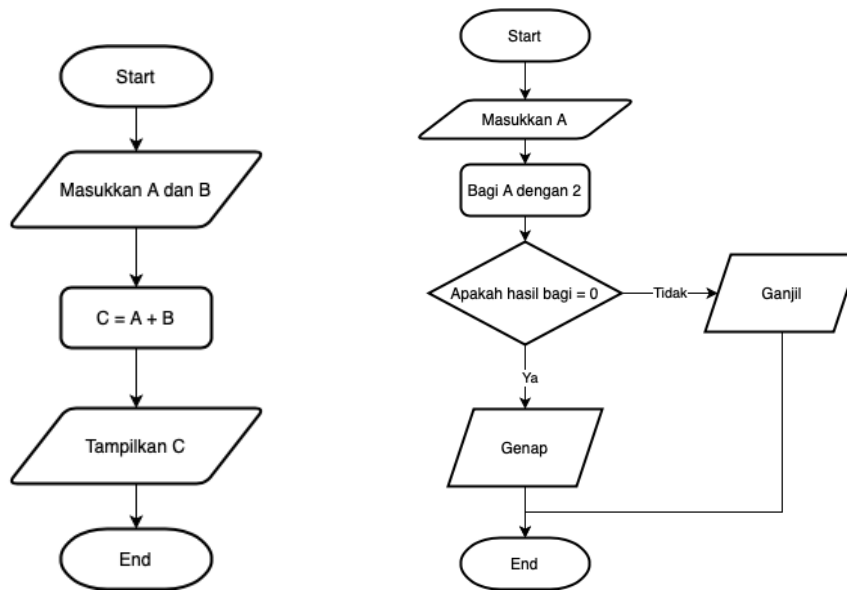
Simbol	Fungsi	Simbol	Fungsi
	Start/End		Input/Output
	Arah aliran		Konektor dalam satu halaman
	Konektor dengan halaman berbeda		Percabangan/ Pengambilan keputusan
	Proses		Sub proses

Tabel 1.1 merupakan daftar simbol-simbol flowchart dasar yang paling sering digunakan, sedangkan Tabel 1.2 adalah daftar simbol-simbol flowchart lain yang dapat digunakan untuk menggambarkan aliran suatu algoritma.

Tabel 1. 2 Simbol Flochart Lainnya

Simbol	Fungsi	Simbol	Fungsi
	Preparation		Dokumen
	Internal storage		Manual input
	Operasi manual		Merger
	Multiple document		Catatan/ Keterangan
	Penyimpanan data		Summing junction

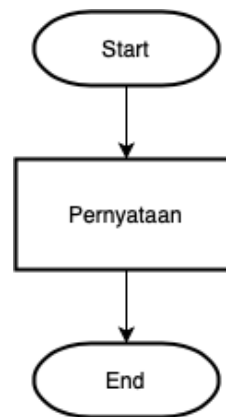
Contoh flowchart untuk menghitung penjumlahan dan perhitungan bilangan genap dan ganjil dapat dilihat pada Gambar 1.1



Gambar 1. 1 Flowchart Penjumlahan dan Hitungan Bilangan Ganjil Genap

Ada beberapa aturan yang harus dipahami dalam membuat flowchart, di antaranya adalah :

- Tidak ada kaidah yang baku.
- Flowchart = gambaran hasil analisa suatu masalah
- Flowchart dapat bervariasi antara satu pemrogram dengan pemrograman lainnya.
- Hindari pengulangan proses yang tidak perlu dan logika yang berbelit sehingga jalannya proses menjadi singkat.
- Jalannya proses digambarkan dari atas ke bawah dan diberikan tanda panah untuk memperjelas.
- Selalu dimulai dengan "Start/Begin" dan diakhiri dengan "Finish/End" seperti pada Gambar 1.2.



Gambar 1. 2 Penggunaan Start dan End

1.2.3 Pseudocode

Pseudocode berasal dari kata pseudo dan code yang artinya kode semu atau menyerupai kode program sebenarnya. Itu berarti pseudocode bukanlah kode program yang sebenarnya, melainkan menggunakan suatu bahasa pemrograman tertentu. Pseudocode berbeda dengan flowchart. Jika flowchart menggunakan simbol-simbol berbentuk gambar untuk menjelaskan alur logika berpikir sehingga dapat digunakan untuk membuat atau menjelaskan suatu program, pseudocode menggunakan bahasa sehingga tidak berupa gambar lagi. Namun, keduanya tetap memiliki tujuan yang sama yaitu membantu menuangkan alur pemikiran ke dalam bentuk tertulis. Dapat dilihat pada Tabel 1.3 untuk memperjelas perbedaan antara flowchart dan pseudocode.

Tabel 1. 3 Perbedaan Algoritma, Flowchart dan Pseudocode

Algoritma	Flowchart	Pseudocode
Menggunakan bahasa tingkat tinggi	Menggunakan simbol berbentuk gambar	Menggunakan bahasa tingkat tinggi yang menyerupai kode program
Tidak standar	Standar	Belum standar
Mudah dibaca	Mudah dimengerti	Mudah dibaca
Tidak dapat langsung dibuat programnya	Tidak dapat langsung dibuat programnya	Bisa langsung dibuat programnya
Masih berupa ide	Berupa rancangan	Sudah hampir berupa implementasi

Tabel 1.4 memberikan contoh algoritma untuk mendapatkan luas persegi panjang dengan pseudocode-nya.

Tabel 1. 4 Algoritma dan Pseudocode Luas Persegi Panjang

Algoritma	Pseudocode
Masukkan Panjang	Input Panjang
Masukkan lebar	Input Lebar
Nilai Luas adalah panjang x lebar	Luas <- panjang x lebar
Tampilkan Luas	Print Luas

Berikut ini adalah contoh pseudocode untuk Hitungan Bilangan Ganjil Genap

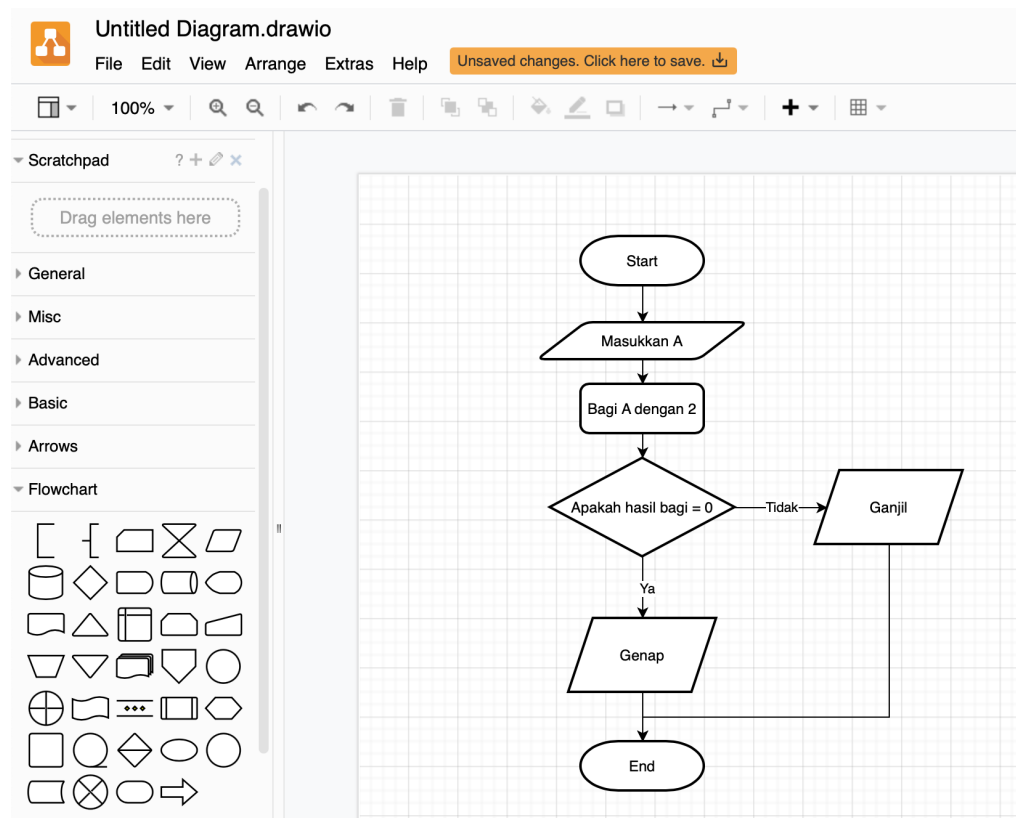
1. STORE 'angka' with any value (number)
- 2.
3. IF 'angka' MOD 2 = 0 THEN

4. DISPLAY "GENAP"
5. ELSE
6. DISPLAY "GANJIL"
7. ENDIF

1.3 Kegiatan Praktikum

1.3.1 Kegiatan 1 : Mengenal Aplikasi Drawio

1. Unduh aplikasi drawio pada alamat <https://www.diagrams.net> . Unduh sesuai dengan sistem operasi yang digunakan.
2. Pasang aplikasi dengan menjalankan file installer Drawio hasil download tadi, ikuti petunjuk yang ada. Jika mengalami kesulitan, mintalah panduan dari asisten praktikum. Aplikasi drawio juga dapat dijalankan secara online melalui halaman <https://draw.io>
3. Setelah selesai, jalankan aplikasi dan pada toolbox sebelah kiri pilihlah *flowchart* seperti pada Gambar 1.3.



Gambar 1. 3 Flowchart pada aplikasi draw.io

4. Mulailah mencoba menggunakan simbol-simbol flowchart dengan meng-klik pada icon yang ditampilkan di toolbox sebelah kiri
5. Jika ada presentasi dari asisten, perhatikan dan ikuti percobaan yang dilakukan.
6. Tulis analisis singkat mengenai hasil praktikum ini.

1.3.2 Kegiatan 2: Membuat Flowchart

1. Dengan menggunakan aplikasi Drawio, buatlah flowchart untuk algoritma menghitung:
 - Keliling lingkaran
 - Luas lingkaran
2. Tulis analisa singkat mengenai hasil praktikum ini

1.3.3 Kegiatan 3: Membuat Pseudocode

1. Jalankan aplikasi pengolah kata (Word/WPS/Libre/Open Office) kemudian buatlah pseudocode untuk kasus pada Praktikum 1.3.2.
2. Tulis analisis singkat mengenai hasil praktikum ini.

BAB 2

Pengenalan Bahasa Python

2.1 Tujuan

1. Dapat menyebutkan peralatan yang dibutuhkan dalam membuat program.
2. Dapat menggunakan salah satu IDE untuk bahasa python.
3. Dapat menulis, mengcompile, dan menjalankan program python sederhana.

2.2 Pengantar

2.2.1 Bahasa Python

Python dikembangkan oleh Guido van Rossum pada tahun 1990 di CWI, Amsterdam sebagai kelanjutan dari bahasa pemrograman ABC. Versi terakhir yang dikeluarkan CWI adalah 1.2. Tahun 1995, Guido pindah ke CNRI sambil terus melanjutkan pengembangan Python. Versi terakhir yang dikeluarkan adalah 1.6. Tahun 2000, Guido dan para pengembang inti Python pindah ke BeOpen.com yang merupakan sebuah perusahaan komersial dan membentuk BeOpen PythonLabs. Python 2.0 dikeluarkan oleh BeOpen. Setelah mengeluarkan Python 2.0, Guido dan beberapa anggota tim PythonLabs pindah ke DigitalCreations.

Saat ini pengembangan Python terus dilakukan oleh sekumpulan pemrogram yang dikoordinir Guido dan Python Software Foundation. Python Software Foundation adalah sebuah organisasi non-profit yang dibentuk sebagai pemegang hak cipta intelektual Python sejak versi 2.1 dan dengan demikian mencegah Python dimiliki oleh perusahaan komersial.

Nama Python dipilih oleh Guido sebagai nama bahasa ciptaannya karena kecintaan Guido pada acara televisi Monty Python's Flying Circus. Oleh karena itu seringkali ungkapan-ungkapan khas dari acara tersebut seringkali muncul dalam korespondensi antar pengguna Python.

Python pun terus berkembang dalam penggunaannya, sehingga fitur-fitur baru dibutuhkan untuk dikembangkan. Versi 2.0 dirilis Oktober 2000 dengan beberapa pengembangan fitur termasuk *Garbage Collector* dan *Memory Management* yang juga menjadi fitur pada beberapa bahasa pemrograman modern lainnya, di antaranya Java dan C#.

Python 3.0 adalah versi perubahan mayor yang dirilis pada Desember 2008. Beberapa sintaksis/statement yang sebelumnya berjalan di versi 2.x, kini tidak lagi berjalan. Contohnya, fungsi print yang sebelumnya adalah statement di Python 2.x, menjadi function di Python 3.x.

Pada saat modul ini ditulis versi Python paling baru adalah 3.8.5 yang dirilis pada 20 Juli 2020. Versi Python tersebut yang akan kita pakai selama praktikum ini.

2.2.2 Librari pada Python

Penggunaan bahasa Python semakin populer karena banyak librari yang tersedia. Librari ini menyediakan beberapa fungsionalitas baru dan membuat pengembang mudah dalam menciptakan suatu aplikasi. Beberapa librari tersebut antara lain

- Framework web seperti Django dan Flask
- Email client seperti smtplib
- Data visualisasi seperti Matplotlib dan PyOpenGL

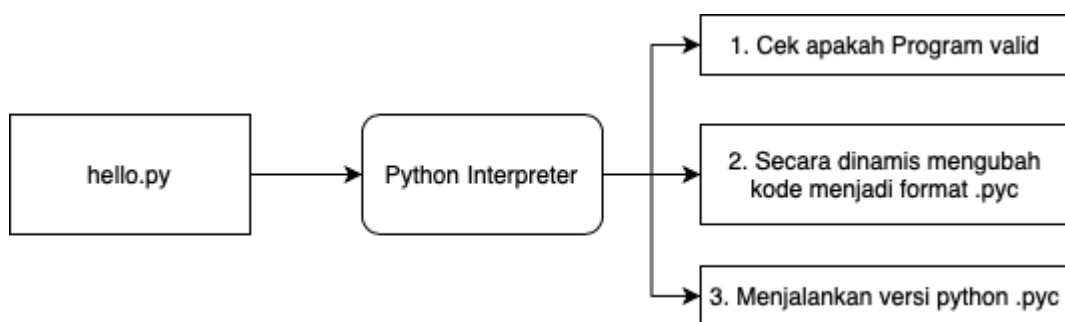
- Machine learning seperti SKLearn dan Tensorflow

Jika kalian ingin belajar lebih jauh tentang librari yang tersedia pada python maka silakan kunjungi website <https://pymotw.com/3/> yang berisi berbagai macam librari yang dapat digunakan beserta penjelasan singkat dan cara penggunaannya.

2.2.3 Bagaimana Python Bekerja

Tidak seperti bahasa pemrograman lainnya, python bukan merupakan bahasa precompiled seperti halnya C++. Python sendiri termasuk dalam bahasa interpreter (penerjemah). Sebuah bahasa interpreter bekerja dengan melakukan eksekusi sejumlah kode yang ditulis dalam bahasa pemrograman tanpa perlu menyusunnya dalam bentuk bahasa mesin. Proses ini berbeda dengan bahasa compiler, dimana kode harus diubah terlebih dahulu ke dalam bahasa mesin sebelum dijalankan.

Interpreter pada python bekerja dengan mengubah kode bahasa python menjadi sebuah format baru yang lebih mudah dijalankan oleh mesin. Python menyimpan format baru ini dengan ekstensi .pyc (huruf c mengindikasikan format python yang telah dicompile) kemudian format baru tersebut dijalankan untuk menghasilkan suatu aplikasi. Ilustrasi proses ini dapat dilihat pada gambar 2.1



Gambar 2. 1 Proses Interpreter Python

Ada beberapa cara yang dapat dilakukan untuk menjalankan program python, beberapa diantaranya antara lain :

- Secara interaktif menggunakan python interpreter

- Disimpan dalam file .py kemudian dijalankan menggunakan command python
- Dijalankan melalui text editor / IDE (Integrated Development Environment)
- Dijalankan melalui IDE berbasis browser

Pada praktikum ini kita akan menggunakan Visual Studio Code dalam pengembangan program python

2.2.4 PyCharm

PyCharm merupakan IDE yang digunakan khusus untuk pengembangan aplikasi menggunakan bahasa Python. PyCharm dikembangkan oleh perusahaan bernama JetBrains, perusahaan yang juga membuat beberapa IDE lain seperti IntelliJ Idea dan Android Studio. Beberapa fitur dari PyCharm antara lain code analysis, graphical debugger, version control system, hingga support pengembangan web melalui Django.

PyCharm merupakan IDE yang dapat berjalan di Windows, macOS dan Linux. Tersedia dua versi yang dapat digunakan. Versi Professional dengan harga mulai \$199 pertahun dan versi Community yang dapat digunakan secara gratis.

2.3 Kegiatan Praktikum

2.3.1 Kegiatan 1 : Menjalankan Python mode interaktif

1. Unduh python dari <https://www.python.org/downloads/windows/> . Pada saat modul ini ditulis versi paling baru adalah 3.8.5
2. Setelah file selesai diunduh kemudian lakukan instalasi. Pastikan mencentang **Add Python 3.8 to PATH** untuk menambahkan python dalam Environment Variables.



Gambar 2. 2 Instalasi Python

3. Buka command prompt kemudian ketik **python --version** untuk mengecek apakah python sudah terinstall dengan benar



Gambar 2. 3 Cek apakah python sudah terinstal

4. Untuk masuk ke mode interaktif ketikkan **python** pada cmd, kemudian ketik **print("Hello World PTI")** untuk menampilkan tulisan **Hello World PTI**

```

C:\WINDOWS\system32\cmd.exe - python
Microsoft Windows [Version 10.0.18363.1016]
(c) 2019 Microsoft Corporation. All rights reserved.

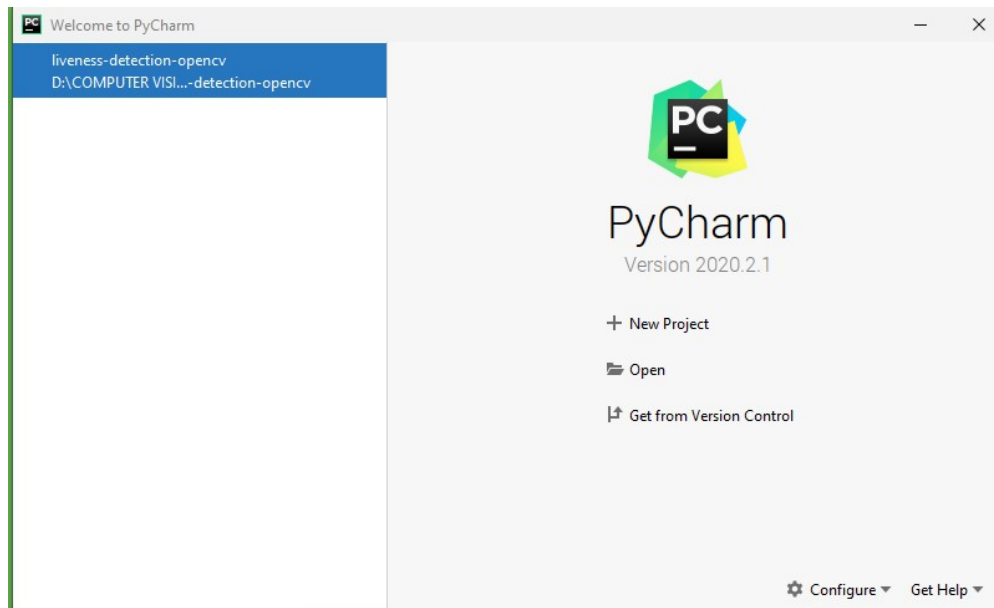
C:\Users\PTI-UMS>python
Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020, 15:43:08) [MSC v.1926 3
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello World PTI")
Hello World PTI
>>> █

```

Gambar 2. 4 Python mode interaktif

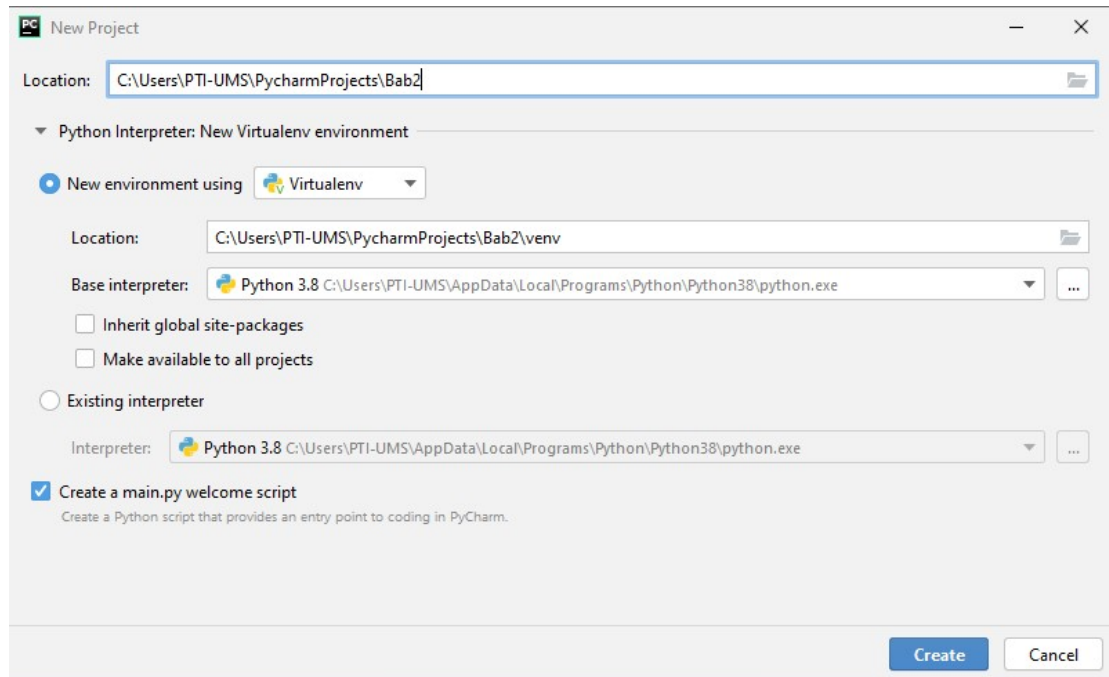
2.3.2 Kegiatan 2 : Menjalankan Python menggunakan IDE PyCharm

1. Unduh PyCharm dari <https://www.jetbrains.com/pycharm/download/> . Terdapat dua pilihan download yaitu versi Professional dan Community. Untuk praktikum ini silakan download versi **Community**. Setelah terdownload kemudian install PyCharm pada komputer.
2. Jalankan PyCharm kemudian pilih New Project



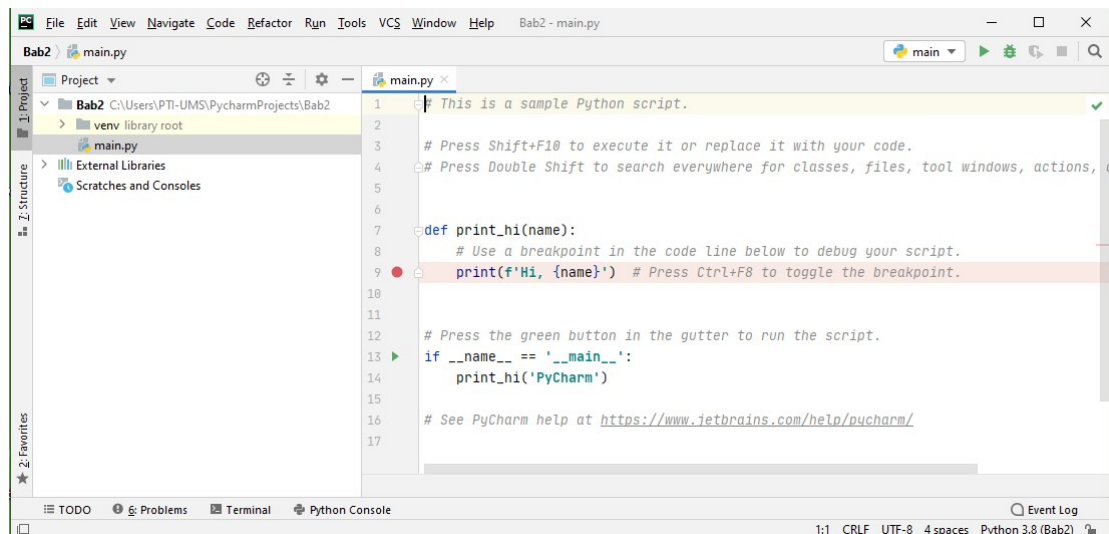
Gambar 2. 5 Tampilan Welcome PyCharm

3. Sesuaikan Location pada folder yang akan digunakan sebagai tempat penyimpanan project, kemudian klik **Create**



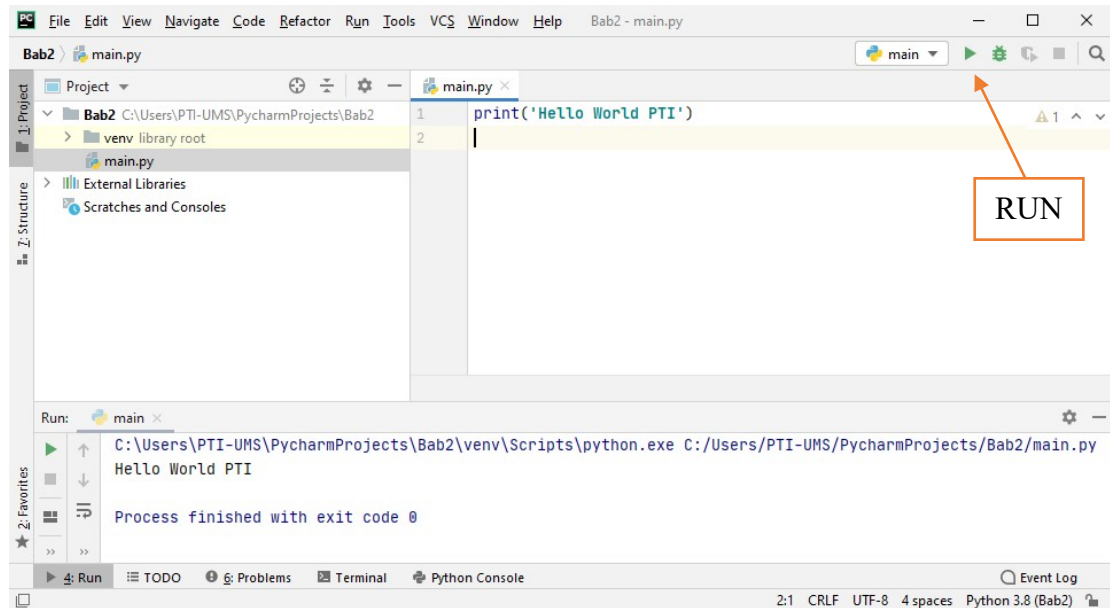
Gambar 2. 6 Tampilan New Project

4. Gambar 2.7 merupakan tampilan UI dari PyCharm



Gambar 2. 7 Tampilan UI PyCharm

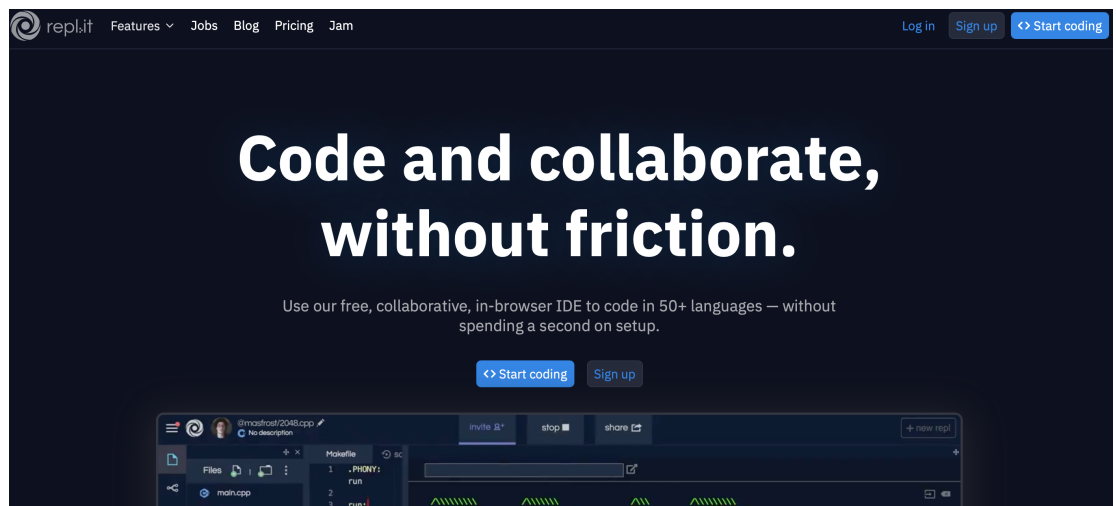
5. Hapus semua kode pada main.py kemudian ketik `print('Hello World PTI')`, kemudian klik logo **RUN** untuk menjalankan kode tersebut



Gambar 2. 8 Run Kode Python

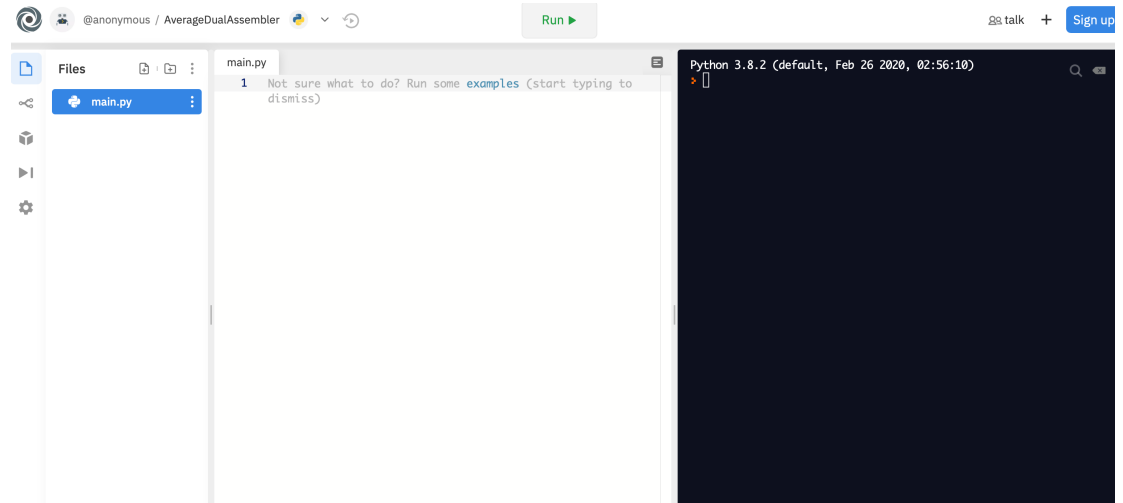
2.3.3 Kegiatan 3 : Menjalankan Python menggunakan Repl.it

1. Buka halaman <https://repl.it> melalui browser
2. Klik logo **Start coding** pada pojok kanan atas untuk membuka halaman IDE, kemudian pilih **Python** sebagai bahasa pemrograman. Kemudian klik **Create repl.**



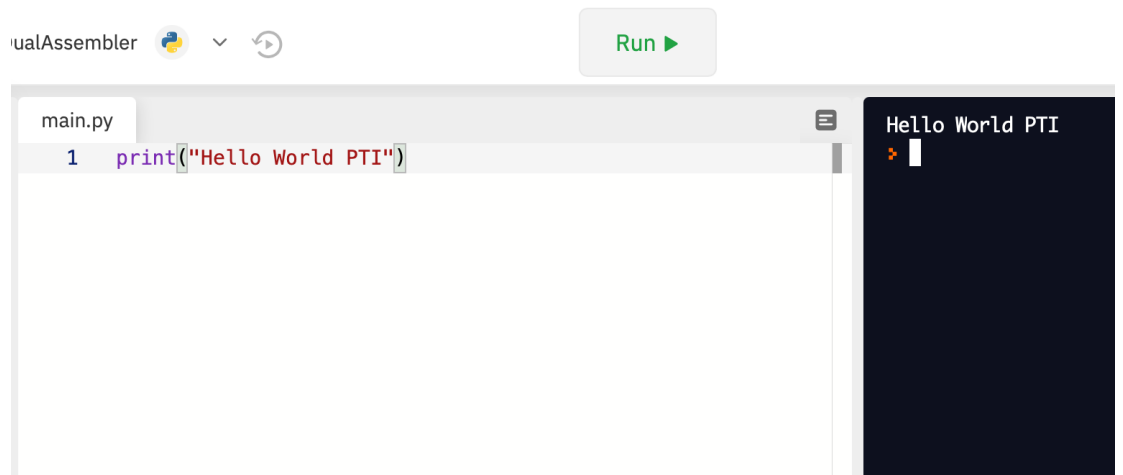
Gambar 2. 9 Tampilan halaman repl.it

3. Gambar merupakan tampilan UI dari IDE repl



Gambar 2. 10 Tampilan UI halaman IDE

4. Pada main.py ketik **print("Hello Word PTI")** kemudian klik **Run** untuk menjalankan kode tersebut. Jika ingin menyimpan kode yang sudah dibuat, maka kalian harus **Sign Up** terlebih dahulu.



Gambar 2. 11 Run Kode Python

2.4 Tugas

1. Berikan analisis singkat tentang tiga metode menjalankan python yang sudah kita lakukan pada kegiatan praktikum diatas
2. Sebutkan dan jelaskan bahasa pemrograman lain yang termasuk dalam bahasa interpreter seperti python
3. Sebutkan 5 Text Editor atau IDE yang dapat digunakan untuk mengembangkan program python beserta website resminya.

Struktur Dasar Bahasa Python

3.1 Tujuan

1. Dapat menjelaskan struktur dasar bahasa python.
2. Dapat menjelaskan input output dan variabel

3.2 Pengantar

Pada bab sebelumnya kita sudah mencoba untuk menjalankan program Hello World menggunakan python. Pada bab ini kita akan membahas lebih detail tentang struktur dasar pada bahasa python. Sebuah program Hello World pada python sangat sederhana jika dibandingkan dengan bahasa lain seperti C++ atau Java. Pada python kita cukup menulis satu baris kode seperti berikut

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

Kita dapat menggunakan text editor atau IDE seperti Notepad, Visual Studio Code, Sublime dan PyCharm untuk membuat file python. Sebuah file python memiliki akhiran .py . Di dalam suatu file .py bisa memiliki satu atau ribuan baris kode python.

Sebagai contoh kita dapat membuat sebuah file bernama main.py yang berisikan function `print('Hello World PTI')` di dalamnya.

Yang menjadi pertanyaan saat ini, darimana fungsi `print()` berasal?

`Print()` merupakan fungsi bawaan yang sudah tersedia pada python, berguna untuk mencetak output ke user. Fungsi bawaan merupakan fungsi yang sudah ada pada Python dan dikenali oleh interpreter sehingga kita tidak perlu membuat kode definisinya.

Sebuah fungsi `Print()` akan mencetak apapun yang ada diantara tanda kurung. Jika berisi String maka akan mencetak String. Jika berisi integer seperti angka 42 maka akan mencetak angka 42. Jika berisi angka desimal seperti 22.4 maka akan mencetak 22.4. Sehingga jika kita menjalankan main.py maka akan mencetak tulisan Hello World PTI.

3.2.1 Input, Output dan Variabel

Mari kita ubah kode program Hello World menjadi lebih interaktif seperti berikut ini

```
1. nama = input('Siapa nama kamu : ')\n2. print ('hallo ', nama)
```

kode di atas jika dijalankan menampilkan output seperti berikut

```
Siapa nama kamu : arif\nhallo  arif
```

Pada baris 1 kita menggunakan fungsi bawaan yang bernama `input()`. Fungsi `input()` berguna untuk menerima inputan dari user. Sehingga jika baris 1 dijalankan maka akan menampilkan kata **Siapa nama kamu :** dan user harus menuliskan kata agar program dapat berjalan. Pada contoh diatas, user mengetik kata arif. Hasil dari inputan user akan disimpan dalam sebuah variabel bernama **nama**. Variabel merupakan suatu tempat pada memory yang digunakan untuk menyimpan suatu data. Pada contoh kode diatas, apapun yang diinputkan oleh user maka akan disimpan

dalam sebuah variabel bernama **nama**. Kemudian pada baris ke-2 kita dapat menampilkan isi data variabel tersebut menggunakan fungsi `print()` yang diikuti dengan menuliskan nama variabelnya.

Isi dari suatu variabel dapat berubah-ubah sesuai dengan kegunaannya. Mari kita lihat kode berikut ini

```
1. nama = input('Siapa nama kamu : ')
2. print ('hallo ', nama)
3. nama = input ('Siapa nama Ayah kamu : ')
4. print ('Ayah kamu bernama ', nama)
```

Jika kita menginputkan kata arif dan deni maka hasil dari kode diatas akan seperti berikut

```
Siapa nama kamu : arif
hallo  arif
Siapa nama Ayah kamu : deni
Ayah kamu bernama  deni
```

Bisa kita lihat pada baris ke-2 dan ke-4, kita tetap memanggil variabel `nama` tapi jika kita lihat dari outputnya. Variabel `nama` memiliki isi yang berbeda. Hal ini terjadi karena bagian memori variabel `nama` yang sebelumnya berisi kata `arif` berubah menjadi `deni`.

Pada python, sebuah variabel memiliki sifat `dynamic typing`. Yaitu sebuah tipe variabel yang dapat berubah secara dinamis saat program berjalan. Sehingga kita tidak perlu memerlukan deklarasi variabel. Perhatikan isi variabel `variabel_satu` pada kode berikut ini.

```
1. variabel_satu = 'Arif'
2. print(variabel_satu)
3. variabel_satu = 40
4. print(variabel_satu)
5. variabel_satu = True
6. print(variabel_satu)
```

Pada baris 1, variabel_satu berisi String 'Arif'. Pada baris 3 , variabel_satu berisi integer 40. Pada baris ke-5, variabel_satu berisi True. Fleksibilitas variabel seperti ini yang merupakan salah satu keunggulan dari bahasa Python.

Penulisan variabel memiliki aturan sebagai berikut ini:

- Menggunakan huruf kecil (lowercase), gunakan format camelCase
- Gunakan kata deskriptif yang mewakili isi dari variabel. Hindari penamaan satu huruf seperti a dan b kecuali untuk penggunaan dalam looping.
- Jika menggunakan lebih dari satu kata, pisahkan menggunakan underscore (_)

Berikut ini merupakan contoh penamaan variabel yang benar pada python

- nama_saya, nama_kamu, user_name
- total, total_pengguna, nomor_rumah
- is_okay, is_correct

3.2.2 Operator Assignment

Perhatikan kode berikut ini, apa fungsi tanda = (sama dengan) yang berada diantara nama dan input()?

```
1. nama = input('Siapa nama kamu : ')
```

tanda = disebut dengan operator assignment. Tanda ini digunakan untuk memberi nilai pada suatu variabel. Pada contoh kode diatas maka, apapun hasil input dari user akan disimpan pada variabel nama.

3.2.3 Komentar

Komentar digunakan untuk menambahkan keterangan pada kode sehingga akan membantu siapapun yang membaca kode tersebut agar lebih mengerti. Suatu komentar pada kode tidak akan dieksekusi oleh interpreter. Pada Python, suatu komentar diawali dengan tanda #. Apapun yang berada setelah tanda # akan dianggap

sebagai komentar dan tidak akan dieksekusi. Perhatikan kode berikut untuk penggunaan komentar pada Python

```
1. #mengambil inputan nama
2. nama = input('Siapa nama kamu : ')
3.
4. print ('hallo ', nama) #menampilkan variabel nama
```

3.3 Kegiatan Praktikum

3.3.1 Kegiatan 1 : Penulisan kode Python

1. Buat file program baru, kemudian jalankan kode berikut ini

```
1. #program pertama saya
2. print('hello world')
3. print('ini bahasa python')
```

2. Ubah kode menjadi seperti berikut ini

```
1. #program
2. #pertama saya
3. print(
4. 'hello world'
5. )
6. print(
7. 'ini bahasa python'
8. )
```

3. Amati hasilnya kemudian tulis analisis singkat mengenai kode program dan hasil tampilan dari langkah 1 dan 3

3.3.2 Kegiatan 2 : Penggunaan Input Output

1. Buat sebuah file program baru kemudian tulis kode program berikut ini

```
1. #mengambil inputan nama
2. nama = input('Siapa nama kamu : ')
3.
4. #mengambil inputan nama ayah
5. umur = input ('Berapa umur kamu: ')
6.
7. #mengambil inputan alamat
8. alamat = input ('Dimana alamatmu : ')
9.
10. #menampilkan variabel
11. print('hallo ', nama)
12. print('umur kamu ', umur)
13. print('alamat kamu ', alamat)
```

2. Amati hasilnya kemudian tulis analisis singkat mengenai kegiatan 2 ini

3.3.3 Kegiatan 3 : Penggunaan Variabel

1. Buat sebuah file program baru kemudian tulis kode program berikut ini

```
1. print("Menghitung Luas dan Keliling Persegi")
2. #mengambil inputan panjang
3. panjang = input('Masukkan nilai Panjang: ')
4.
5. #mengambil inputan lebar
6. lebar = input ('Masukkan nilai Lebar: ')
7.
8. #menghitung luas dan keliling
9. luas = int(panjang) * int(lebar)
10. keliling = 2 * (int(panjang) + int(lebar))
11.
12. #menampilkan hasil
```

```
13. print ('Luas Persegi adalah ', luas , ' sedangkan kelilingnya adalah ',  
keliling)
```

2. Amati hasilnya kemudian tulis analisis singkat mengenai kegiatan 3 ini

3.4 Tugas

1. Buatlah program untuk menampilkan data diri anda dengan menggunakan variabel untuk menempatkan data nama, nim, dan angkatan.
2. Buatlah program untuk melakukan perhitungan akar pangkat tiga dari sebuah angka, dengan angka dan hasilnya disimpan dalam suatu variabel.

Tipe Data dan Operator

4.1 Tujuan

3. Dapat menjelaskan tipe data dan operator pada python
4. Dapat mengimplementasikan tipe data dan operator pada pemrograman

4.2 Pengantar

Tipe data dan operator merupakan komponen penting dalam pemrograman Python. Memiliki pemahaman tentang tipe data yang baik akan sangat membantu meningkatkan kemampuan dalam bidang pemrograman.

4.2.1 Operator Aritmatika

Operator aritmatika digunakan untuk melakukan operasi matematika seperti tambah, kurang, perkalian dan pembagian. Pada python operator aritmatika biasanya dilambangkan dalam 1 atau 2 karakter seperti pada tabel 4.1 berikut.

Tabel 4. 1 Tabel Operator Aritmatika

Operator	Keterangan	Contoh
+	Penjumlahan nilai di kiri dengan kanan	1 + 2

-	Mengurangi nilai kiri dengan kanan	4-3
*	Mengalikan nilai kiri dan kanan	4*3
/	Membagi nilai kiri dengan kanan	4/2
%	Modulus, mengembalikan sisa hasil bagi	4%3
**	Nilai kiri pangkat nilai kanan	2**3
//	Pembagian bilangan, hasil akan dibulatkan	4//3

4.2.2 Integer dan Float

Pada python terdapat dua tipe data yang dapat digunakan untuk nilai berupa angka yaitu:

- `int` : untuk nilai bilangan integer / bulat
- `float` : untuk nilai bilangan desimal

Berikut ini merupakan kode untuk membuat tipe data integer dan float

```

1. #deklarasi variabel
2. nilai_x = int(4)
3. nilai_y = float(4.2)
4.
5. #menampilkan nilai dan tipe data nilai X
6. print('nilai x adalah : ', nilai_x)
7. print('tipe data x adalah : ', type(nilai_x))
8.
9. #menampilkan nilai dan tipe data nilai Y
10. print('nilai y adalah : ', nilai_y)
11. print('tipe data y adalah : ', type(nilai_y))

```

kita dapat menggunakan fungsi **type** untuk menampilkan tipe data seperti pada baris 7 dan 11. Hasil dari kode diatas adalah :

```

nilai x adalah : 4
tipe data x adalah : <class 'int'>

```

```
nilai y adalah : 4.2  
tipe data y adalah : <class 'float'>
```

4.2.3 String

String merupakan tipe data jika kita ingin menggunakan nilai berupa huruf, angka atau kalimat. Dalam python String biasa ditulis dengan kode **str**. Untuk menggunakan tipe data ini harus diapit oleh tanda petik (' atau ") di awal dan di akhir kalimat.

Perhatikan kode berikut ini untuk mengetahui cara penulisan string.

```
1. string_satu = 'Halo, selamat Datang'  
2. string_dua = "Halo, selamat Datang"  
3. string_tiga = '\Halo\ ', selamat datang'  
4.  
5. print(string_satu)  
6. print(string_dua)  
7. print(string_tiga)
```

Kode diatas akan menghasilkan output seperti berikut

```
Halo, selamat Datang  
Halo, selamat Datang  
'Halo', selamat datang
```

4.2.4 Boolean

Boolean merupakan tipe data yang memiliki nilai True dan False, True akan bernilai 1 sedangkan False akan bernilai 0.

Untuk dapat menggunakan tipe data boolean, kita bisa menggunakan operator perbandingan seperti pada tabel 4.2 berikut ini

Tabel 4. 2 Operator Perbandingan

Simbol	Nilai Boolean	Keterangan
5 < 3	False	Kurang dari

5 > 3	True	Lebih dari
3 <= 3	True	Kurang dari atau sama dengan
3 >= 5	False	Lebih dari atau sama dengan
3 == 5	False	Sama dengan
3 != 5	True	Tidak sama dengan

Selain operator perbandingan, pada tabel 4.3 terdapat juga operator logika yang perlu kita pelajari.

Tabel 4. 3 Operator Logika

Operator Logika	Nilai Boolean	Keterangan
5 < 3 and 5 == 5	False	And , bernilai True jika kedua sisi bernilai benar
5 < 3 or 5 == 5	True	Or , bernilai True jika ada satu sisi bernilai benar
Not 5<3	True	Not , kebalikan dari nilai boolean

4.2.5 Konversi Tipe Data

Kita dapat melakukan konversi tipe data dengan menggunakan fungsi konversi seperti **int()**, **float()**, **str()**. Perhatikan kode berikut ini :

```

1. #deklarasi variabel
2. nilai_x = int(4)
3.
4. #menampilkan nilai dan tipe data nilai X
5. print('nilai x adalah : ', nilai_x)
6. print('tipe data x adalah : ', type(nilai_x))
7.
8. #konversi tipe data nilai X ke float
9. nilai_konversi = float(nilai_x)
10.

```

```

11. #menampilkan nilai dan tipe data hasil konversi
12. print('nilai x setelah konversi adalah : ', nilai_konversi)
13. print('tipe data x setelah konversi adalah : ', type(nilai_konversi))

```

kode diatas akan menghasilkan output seperti berikut

```

nilai x adalah : 4
tipe data x adalah : <class 'int'>
nilai x setelah konversi adalah : 4.0
tipe data x setelah konversi adalah : <class 'float'>

```

Terlihat bahwa nilai x yang awalnya bertipe integer menjadi tipe float

4.3 Kegiatan Praktikum

4.3.1 Kegiatan 1 : Bekerja dengan tipe data integer

1. Buat sebuah file program baru kemudian tulis kode berikut ini

```

1. #fungsi input() akan membaca inputan menjadi string
2. umur = input('Masukkan umur kamu sekarang : ')
3. print(type(umur))
4.
5. #ubah string ke int untuk melakukan operasi aritmatika
6. umur_baru = int(umur) + 10
7. print('Umur kamu 10 tahun lagi adalah : ', umur_baru)

```

2. Amati hasilnya kemudian tulis analisis singkat mengenai kegiatan ini

4.3.2 Kegiatan 2 : Bekerja dengan tipe data float

1. Buat sebuah file program baru kemudian tulis kode berikut ini

```

1. angka_1 = input('Masukkan angka 1 :')
2. angka_2 = input('Masukkan angka 2 :')
3.
4. hasil_bagi = float(angka_1) / float(angka_2)
5. hasil_modulus = float(angka_1) % float(angka_2)

```

```
6. print('Hasil angka 1 dibagi angka 2 = ', hasil_bagi)
7. print('Hasil modulus angka 1 dengan angka 2 = ', hasil_modulus)
```

2. Amati hasilnya kemudian tulis analisis singkat mengenai kegiatan ini

4.3.3 Kegiatan 3 : Bekerja dengan tipe data string

1. Buat sebuah file program baru kemudian tulis kode berikut ini

```
1. #deklarasi string
2. jml_mangga = "34"
3. jml_semangka = "15"
4.
5. #penjumlahan jika tipe data string
6. total_buah = jml_mangga + jml_semangka
7. print(total_buah)
8.
9. #penjumlahan jika tipe data diubah ke int
10. total_buah = int(jml_mangga) + int(jml_semangka)
11. print(total_buah)
```

2. Amati hasilnya kemudian tulis analisis singkat mengenai kegiatan ini

4.4 Tugas

1. Buatlah sebuah program python yang akan mengkonversi jarak dalam kilometer menjadi meter
 - a. Program akan meminta input berupa jarak dalam kilometer
 - b. Ubah input dari user menjadi tipe int kemudian buat perhitungan untuk mencari jarak tersebut dalam satuan meter
 - c. Tampilkan output berupa jarak dalam satuan meter.

Operasi pada String

5.1 Tujuan

5. Dapat menjelaskan fungsi bawaan dari string
6. Dapat mengimplementasikan fungsi string pada pemrograman

5.2 Pengantar

Python menyediakan beberapa fungsi bawaan yang dapat kita gunakan untuk melakukan operasi-operasi yang melibatkan string. Beberapa fungsi tersebut antara lain

5.2.1 Fungsi `upper()` dan `lower()`

Kedua fungsi ini digunakan untuk mengubah ukuran huruf menjadi huruf kapital dan huruf kecil. Untuk mengubah ke huruf kapital digunakan fungsi **`upper()`**, sedangkan jika ingin mengubah ke huruf kecil gunakan fungsi **`lower()`**. Perhatikan kode berikut ini untuk penggunaan kedua fungsi tersebut

```
1. #kata asli
2. kata = 'Selamat datang di Prodi PTI UMS'
3.
```

```

4. #ubah menjadi kapital dan kecil
5. kapital = kata.upper()
6. kecil = kata.lower()
7.
8. #cetak kata
9. print(kata)
10. print(kapital)
11. print(kecil)

```

Hasil output dari kode diatas seperti berikut

```

Selamat datang di Prodi PTI UMS
SELAMAT DATANG DI PRODI PTI UMS
selamat datang di prodi pti ums

```

5.2.2 Fungsi len()

Fungsi len() digunakan untuk mengetahui panjang dari suatu string. Perhatikan contoh kode berikut ini untuk menggunakan fungsi len().

```

1. #kata asli
2. kata = 'Selamat datang di Prodi PTI UMS'
3.
4. #cari panjang kata
5. panjang = len(kata)
6.
7. #cetak kata
8. print(kata)
9. print('panjang string diatas adalah : ', panjang)

```

Hasil output dari kode diatas adalah

```

Selamat datang di Prodi PTI UMS
panjang string diatas adalah : 31

```

5.2.3 Fungsi mengatur rerata teks

Untuk merapikan output dari kalimat kita bisa menggunakan beberapa fungsi bawaan dari string. Untuk mengatur rata tengah kita gunakan `center()`. Untuk mengatur rata kanan gunakan `rjust()`. Untuk mengatur rata kiri gunakan `ljust()`. Untuk menggunakan fungsi ini kita perlu menambahkan argumen tambahan berupa integer yang mewakili panjang string.

Perhatikan kode berikut ini

```
1. #kata asli
2. kata = 'Selamat datang di Prodi PTI UMS'
3.
4. #cetak rata tengah
5. print(kata.center(50))
6.
7. #cetak rata kiri
8. print(kata.ljust(50))
9.
10. #cetak rata kanan
11. print(kata.rjust(50))
```

Output dari kode diatas adalah seperti berikut

```
        Selamat datang di Prodi PTI UMS
Selamat datang di Prodi PTI UMS
                Selamat datang di Prodi PTI UMS
```

5.2.4 Fungsi join() dan split()

Fungsi `join()` digunakan untuk menggabungkan string yang terdapat dalam list, tuple ataupun set. Kebalikan dari fungsi `join()` adalah `split()`, digunakan untuk memisahkan string.

Perhatikan contoh kode berikut ini

```
1. #fungsi join
```



```

2. kata = ['Pendidikan', 'Teknik', 'Informatika']
3. print(' '.join(kata))
4.
5. #fungsi split
6. kalimat = 'Fakultas Keguruan dan Ilmu Pendidikan'
7. print(kalimat.split(' '))

```

Output dari kode diatas adalah seperti berikut

```

Pendidikan Teknik Informatika
['Fakultas', 'Keguruan', 'dan', 'Ilmu', 'Pendidikan']

```

5.2.5 Fungsi index()

Fungsi index() digunakan untuk mengetahui posisi indeks pertama suatu karakter pada string. Perhatikan kode berikut ini untuk mengetahui penggunaan index()

```

1. kata = 'Selamat datang di Prodi PTI UMS'
2.
3. #penggunaan index()
4. print('indeks pertama huruf e berada di : ',kata.index('e'))
5. print('indeks pertama huruf a berada di : ',kata.index('a'))
6. print('indeks pertama huruf da berada di : ',kata.index('da'))

```

output dari kode diatas adalah

```

indeks pertama huruf e berada di : 1
indeks pertama huruf a berada di : 3
indeks pertama huruf da berada di : 8

```

5.2.6 Fungsi replace()

Fungsi replace() digunakan jika ada karakter atau bagian dari suatu string yang ingin diganti. Perhatikan kode berikut ini untuk contoh penggunaan replace().

```

1. kata = 'Selamat datang di Prodi PTI UMS'
2.
3. #fungsi replace()

```

```

4. kata_baru = kata.replace('PTI', 'Pendidikan Informatika')
5.
6. #cetak kata
7. print(kata)
8. print(kata_baru)

```

output dari kode diatas adalah

```

Selamat datang di Prodi PTI UMS
Selamat datang di Prodi Pendidikan Informatika UMS

```

5.2.7 String formatting

Pada kegiatan praktikum sebelumnya, kita biasa menggunakan fungsi print() untuk menampilkan string. Terdapat satu cara lagi yang bisa digunakan yaitu menggunakan fungsi format(). Untuk menggunakan fungsi format() kita perlu menggunakan placeholder berupa tanda {}. Tanda {} nantinya akan diganti dengan string yang kita masukkan. Perhatikan contoh kode berikut ini.

```

1. nama = 'Arif'
2. umur = '20'
3.
4. print('Pada tahun ini, umur {} adalah {}'.format(nama, umur))

```

hasil dari output kode diatas adalah

```

Pada tahun ini, umur Arif adalah 20

```

5.3 Kegiatan Praktikum

5.3.1 Kegiatan 1 : Operasi pada String

1. Buat sebuah file program baru, kemudian tuliskan kode berikut ini

```

1. string = "Hi Semua, Selamat datang di Channel Sobat PTI"
2.
3. print("Panjang string = ", len(string))
4.

```

```

5. print("Index pertama huruf a berada di ",string.index("a"))
6.
7. print("huruf a muncul sebanyak ", string.count("a"))
8.
9. print(string.upper())
10.
11. print(string.lower())
12.
13. print(string.split(" "))

```

2. Amati hasilnya kemudian tulis analisis singkat mengenai kegiatan 1 ini

5.3.2 Kegiatan 2 : String formatting

1. Buat sebuah file program baru, kemudian tuliskan kode berikut ini

```

1. kampus = 'UMS'
2. lokasi = 'Surakarta'
3. print('Selamat Datang di {}, kampus {} tercinta'.format(lokasi,kampus))
4.
5. string = "Nilai {1} {0}, adalah {2}%"
6. print(string.format('Irawan', 'Arthur', 75))
7.
8. string_dua = '{band} menciptakan lagu {lagu} pada tahun {tahun}'
9. print(string_dua.format(band='Barasudara',lagu='Mengunci Ingatan',tahun='2015'))

```

2. Amati hasilnya kemudian tulis analisis singkat mengenai kegiatan 2 ini

5.3.3 Kegiatan 3 : Slicing pada String

1. Buat sebuah file program baru, kemudian tuliskan kode berikut ini

```

1. kata='Pendidikan Teknik Informatika UMS'

```

```
2.  
3. print ("Hasilnya [:] = ",(kata[:]))  
4. print ("Hasilnya [8:] = ",(kata[8:]))  
5. print ("Hasilnya [8:18] = ",(kata[8:15]))  
6. print ("Hasilnya [:15] = ",(kata[:5]))
```

2. Amati hasilnya kemudian tulis analisis singkat mengenai kegiatan 3 ini

5.4 Tugas

1. Buatlah sebuah program python yang menerima inputan berupa nama depan, nama belakang dan nim.
2. Gabungkan nama depan dan nama belakang menjadi satu variabel bernama nama lengkap
3. Print output ke layar ke dalam bentuk berikut ini. Gunakan fungsi upper() untuk kalimat Universitas Muhammadiyah Surakarta

Pendidikan Teknik Informatika

Nama : [nama lengkap]

Nim : [nim]

Fakultas Keguruan dan Ilmu Pendidikan

BAB 6

Struktur Data pada Python

6.1 Tujuan

7. Dapat menjelaskan list, tuple, set dan dictionary pada bahasa python.
8. Dapat mengimplementasikan list, tuple, set dan dictionary pada pemrograman

6.2 Pengantar

Struktur data merupakan cara yang digunakan untuk menangani suatu data pada bahasa pemrograman. Pada bab ini kita akan belajar empat struktur data yang disediakan oleh python yaitu list, tuple, set dan dictionary.

6.2.1 List

List merupakan sekumpulan data yang terdapat dalam sebuah variabel. Penggunaan list hampir sama dengan fungsi array pada bahasa pemrograman yang lain. Setiap elemen anggota pada list diakses menggunakan indeks yang dimulai dari 0. Data yang tersimpan pada list diapit dengan tanda kurung siku [].

Berikut contoh penggunaan list.

```

1. #deklarasi list
2. pti = ['fkip', 'ums', 2014, True]
3.
4. #akses list menggunakan indeks 0
5. print(pti[0])
6.
7. #akses list menggunakan indeks 2
8. print(pti[2])

```

Baris 1 merupakan deklarasi list yang memiliki 4 element yang berbeda tipe yaitu string, integer dan boolean. Indeks elemen ini diawali dari angka 0. Sehingga ketika program tersebut dijalankan akan menghasilkan output.

```

fkip
2014

```

Indeks pada list dapat bernilai negatif. Indeks -1 akan menunjuk elemen terakhir pada list. Indeks -2 akan menunjukkan elemen kedua dari akhir. Kode dibawah ini akan mengakses elemen terakhir dari list dan elemen ketiga dari akhir.

```

1. #deklarasi list
2. pti = ['fkip', 'ums', 2014, True]
3.
4. #akses elemen terakhir menggunakan indeks -1
5. print(pti[-1])
6.
7. #akses elemen ketiga dari akhir menggunakan indeks -3
8. print(pti[-3])

```

Kita juga dapat mengakses elemen yang berada pada rentang indeks tertentu menggunakan slicing operator atau tanda titik dua (:). Perhatikan kode berikut ini untuk mengetahui penggunaan slicing operator.

```

1. #deklarasi list
2. pti = ['p', 't', 'i', 'u', 'm', 's']
3.

```

```

4. #akses elemen ketiga hingga kelima
5. print(pti[2:5])
6.
7. #akses elemen pertama hingga ke empat
8. print(pti[:4])
9.
10. #akses elemen keempat hingga ke terakhir
11. print(pti[3:])
12.
13. #akses elemen awal hingga akhir
14. print(pti[:])

```

List merupakan tipe data yang bersifat mutable. Artinya data yang berada di dalam list dapat diubah. Kita dapat menggunakan tanda assignment (=) untuk mengubah data yang berada dalam list.

```

1. #deklarasi list
2. kabupaten = ['sukoharjo', 'sragen', 'wonogiri', 'karanganyar', 'klaten']
3.
4. #ubah element pertama
5. kabupaten[0] = 'sleman'
6.
7. #cetak list
8. print(kabupaten)
9.
10. #ubah element kedua hingga ke empat
11. kabupaten[1:4] = ['kulonprogo', 'bantul', 'gunungkidul']
12.
13. #cetak list
14. print(kabupaten)

```

kode diatas akan menghasilkan output sebagai berikut

```

['sleman', 'sragen', 'wonogiri', 'karanganyar', 'klaten']
['sleman', 'kulonprogo', 'bantul', 'gunungkidul', 'klaten']

```

Untuk menambahkan data ke dalam list kita dapat menggunakan fungsi **append()**, sedangkan jika data yang ditambahkan berjumlah lebih dari satu maka dapat menggunakan fungsi **extend()**. Fungsi **insert()** digunakan jika kita ingin menambahkan data pada indeks tertentu pada list.

```
1. #deklarasi list
2. kabupaten = ['sukoharjo', 'sragen']
3.
4. #menambah 1 elemen
5. kabupaten.append('wonogiri')
6.
7. #cetak list
8. print(kabupaten)
9.
10. #menambah 2 elemen
11. kabupaten.extend(['karanganyar', 'klaten'])
12.
13. #cetak list
14. print(kabupaten)
15.
16. #insert elemen pada indeks ke 1
17. kabupaten.insert(1, 'boyolali')
18.
19. #cetak list
20. print(kabupaten)
```

kode diatas akan menghasilkan output seperti berikut

```
['sukoharjo', 'sragen', 'wonogiri']
['sukoharjo', 'sragen', 'wonogiri', 'karanganyar', 'klaten']
['sukoharjo', 'boyolali', 'sragen', 'wonogiri', 'karanganyar', 'klaten']
```

Fungsi **del** kita gunakan jika ingin menghapus salah satu atau keseluruhan elemen yang ada pada list. Sedangkan fungsi **remove()** dapat kita gunakan untuk menghapus elemen berdasar nilai yang diberikan. Fungsi **pop()** digunakan untuk menghapus

indeks terakhir dari list. Fungsi terakhir yaitu **clear()** digunakan untuk mengosongkan isi list.

```
1. #deklarasi list
2. kabupaten = ['sukoharjo', 'sragen', 'wonogiri', 'karanganyar', 'klaten']
3.
4. #hapus indeks ke 2
5. del kabupaten[2]
6.
7. #hapus beberapa element
8. del kabupaten[1:3]
9.
10. #hapus seluruh list
11. del kabupaten
12.
13. #deklarasi list
14. pti = ['p', 't', 'i', 'u', 'm', 's']
15.
16. #hapus huruf t
17. pti.remove('t')
18.
19. #hapus elemen terakhir
20. pti.pop()
21.
22. #kosongkan list
23. pti.clear()
```

Beberapa fungsi lain yang dapat kita gunakan dalam list yaitu

Tabel 6. 1 Fungsi lain yang tersedia pada list

Fungsi	Keterangan
len()	Mengetahui panjang list
max()	Mengembalikan nilai tertinggi pada list
min()	Mengembalikan nilai terendah pada list

sorted()	Mengurutkan list dari terendah ke besar
index()	Mengembalikan nilai index yang sesuai
count()	Menghitung jumlah pada element berdasar angka yang diinput
reverse()	Membalikkan urutan list
copy()	Membuat duplikat list

6.2.2 Tuple

Tuple merupakan bentuk lain daripada list yang memiliki perbedaan pada sifatnya. Jika list bersifat mutable maka tuple bersifat immutable. Artinya data yang ada pada tuple tidak dapat ditambah, diubah atau dihapus. Penulisan tuple diapit dengan tanda kurung (). Penulisan tanda kurung ini bersifat optional namun disarankan untuk tetap digunakan. Perhatikan kode berikut untuk contoh penggunaan tuple.

```
1. koordinat= (13.4125, 103.866667)
2. print("Latitude:", koordinat[0])
3. print("Longitude:", koordinat[1])
```

Output diatas akan menghasilkan seperti berikut

```
Latitude: 13.4125
Longitude: 103.866667
```

Data pada tuple dapat kita assign ke variabel yang berbeda. Konsep ini dinamakan tuple unpacking. Perhatikan baris ke dua pada contoh kode berikut ini

```
1. ukuran = 20,30,40
2. panjang, lebar, tinggi = ukuran
3. print('Buku ini memiliki dimensi {} x {} x {}'.format(panjang,lebar,tinggi))
```

pada baris kedua, nilai dari tuple ukuran diassign ke dalam tiga variabel yang berbeda yaitu panjang, lebar dan tinggi.

6.2.3 Set

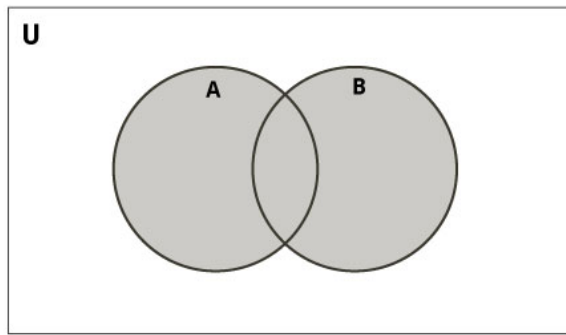
Set merupakan sekumpulan data dengan elemen-elemen yang berbeda (unique) sehingga salah satu fungsi set digunakan untuk menghapus elemen yang sama / ganda. Terdapat dua jenis set yaitu set dinamis yang bisa diubah data dan ukurannya, ditandai dengan kata set dan set statis yang isinya tidak dapat diubah, ditandai dengan kata frozenset.

```
1. #set dinamis
2. pti =set('Pendidikan Teknik Informatika')
3. print(type(pti))
4. print(pti)
5.
6. #set statis
7. kip = frozenset('Keguruan Ilmu Pendidikan')
8. print(type(kip))
9. print(kip)
```

Kode diatas akan menghasilkan output

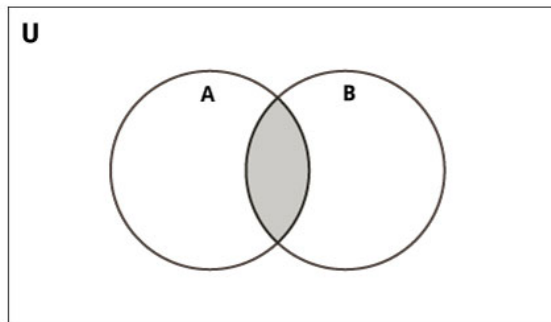
```
<class 'set'>
{'o', 'd', 'e', 'f', 'i', 'I', ' ', 'P', 'k', 'r', 't', 'T', 'a', 'n', 'm'}
<class 'frozenset'>
frozenset({'g', 'd', 'e', 'l', 'i', 'I', 'r', ' ', 'P', 'u', 'k', 'K', 'a', 'n', 'm'})
```

Set dalam python merupakan penerapan set pada ilmu Matematika. Jadi di dalam set kita dapat melakukan operasi keanggotaan seperti union dan intersection.



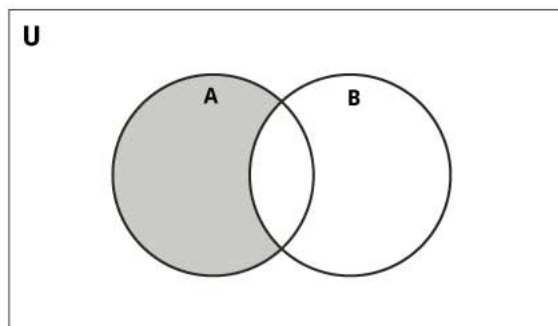
Gambar 6. 1 Operasi Union

Union merupakan gabungan dari semua elemen dari kedua set. Union dapat digunakan menggunakan tanda \cup atau fungsi **union()**.



Gambar 6. 2 Operasi Intersection

Intersection merupakan operasi untuk menampilkan elemen-elemen yang menjadi irisan antara dua set. Intersection dapat digunakan dengan tanda \cap .



Gambar 6. 3 Operasi Difference

Difference (komplemen) merupakan elemen-elemen yang hanya terdapat pada satu set saja dan tidak terdapat pada set yang lain. Difference dapat digunakan dengan tanda minus (-). Perhatikan kode berikut untuk contoh penggunaan operasi himpunan dalam set.

```
1. #deklarasi set
2. A = {1,2,3,4,5}
3. B = {4,5,6,7,8}
4.
5. #union
6. print (A|B)
7.
8. #intersect
9. print (A&B)
10.
11. #difference
12. print (A-B)
```

Kode diatas akan menghasilkan output

```
{1, 2, 3, 4, 5, 6, 7, 8}
{4, 5}
{1, 2, 3}
```

Tabel 6.2 berikut ini merupakan fungsi-fungsi yang dapat digunakan dalam set

Tabel 6. 2 Fungsi yang ada dalam Set

Fungsi	Keterangan
add()	Menambah satu elemen baru pada set
update()	Menambah elemen baru pada set
remove()	Menghapus elemen pada set
clear()	Mengosongkan isi set
pop()	Menghapus elemen pertama pada set

6.2.4 Dictionaries

Dictionaries merupakan kumpulan data yang terdiri dari sepasang kunci dan nilai. Dictionaries mendukung semua tipe data yang ada pada python. Sehingga elemen-elemen yang ada dalam dictionaries bisa berupa elemen yang berbeda tipe. Penulisan dictionaries ditandai dengan tanda kurung kurawal {}. Perhatikan contoh kode berikut ini untuk penulisan dan pengaksesan dictionaries.

```
1. #dictionaries tanpa isian
2. lantai = {}
3.
4. #deklarasi dictionaries
5. lantai = {'lobby':1,'kantor':2,'kantin':3,'parkir':'rooftop'}
6.
7. #akses dictionaries
8. print(lantai['lobby'])
9. print(lantai.get('parkir'))
```

kode diatas akan menghasilkan output

```
1
rooftop
```

Karena data pada dictionaries bersifat mutable. Maka kita dapat melakukan operasi penambahan, update dan hapus data.

```
1. #deklarasi dictionaries
2. lantai = {'lobby':1,'kantor':2,'kantin':3,'parkir':'rooftop'}
3.
4. #akses dictionaries
5. lantai['parkir'] = 'basement'
6.
7. #menambah elemen
8. lantai['labkom'] = 4
9.
10. #hapus elemen berdasar key menggunakan pop
11. lantai.pop('lobby')
12.
```

```
13. #hapus semua elemen
14. lantai.clear()
```

Tabel 6.3 berikut ini merupakan fungsi-fungsi yang ada pada dictionaries

Tabel 6. 3 Fungsi pada dictionaries

Fungsi	Keterangan
clear()	Menghapus seluruh elemen pada dictionaries
item()	Menampilkan seluruh elemen
keys()	Menampilkan seluruh kunci
values()	Menampilkan seluruh nilai yang terdapat dalam dictionaries
pop(key)	Menghapus elemen dengan kunci tertentu
get()	Mengembalikan nilai balik berupa nilai dari kunci

6.3 Kegiatan Praktikum

6.3.1 Kegiatan Praktikum 1 : Bekerja dengan list

1. Buat sebuah file program baru, kemudian tuliskan kode berikut ini

```
1. bulan1 = ['januari', 'februari', 'maret', 'april', 'mei', 'juni']
2. bulan2 = ['juli', 'agustus', 'september', 'oktober', 'november', 'desember']

3. print('Jumlah elemen pada list bulan1 : ', len(bulan1))
4. tahun = bulan1 + bulan2
5. print('Jumlah elemen pada list tahun : ', len(tahun))
6. print(tahun)
7. print(tahun[2:5])
8. print(tahun[:6])
9. print(tahun[8:])
10. del tahun[2]
11. tahun.remove('desember')
12. print(tahun)
13. tahun.insert(2, 'maret')
14. tahun.append('desember')
```

```
15. print(tahun)
16. tahun.reverse()
17. print(tahun)
```

2. Amati hasilnya kemudian tulis analisis singkat mengenai kegiatan 1 ini

6.3.2 Kegiatan Praktikum 2 : Bekerja dengan Tuple

1. Buat sebuah file program baru, kemudian tuliskan kode berikut ini

```
1. truk = ('hino', 3000, 2.5 , 130 )
2. merk, cc , berat , top_speed = truk
3. print(truk[0])
4. print(truk[:2])
5. print(truk[2:])
6. print(truk.index(3000))
7. print(2.5 in truk)
8. print('truk {} memiliki berat {} ton, kapasitas {} cc dan top speed {} km/jam'.
    format(merk,berat,cc,top_speed))
```

2. Amati hasilnya kemudian tulis analisis singkat mengenai kegiatan 2 ini

6.3.3 Kegiatan Praktikum 3 : Bekerja dengan Set

1. Buat sebuah file program baru, kemudian tuliskan kode berikut ini

```
1. set_satu = {1,2,3}
2. set_dua = {4,5,6}
3. set_satu.add(4)
4. set_dua.add(7)
5. print(set_satu)
6. print(set_dua)
7. set_satu.update([5,6])
8. set_dua.update([8,9])
9. print(set_satu)
10. print(set_dua)
11. set_satu.discard(6)
12. set_dua.remove(4)
```



```
13. print(set_satu | set_dua)
14. print(set_satu & set_dua)
15. print(1 in set_satu)
```

2. Amati hasilnya kemudian tulis analisis singkat mengenai kegiatan 3 ini

6.3.4 Kegiatan Praktikum 4 : Bekerja dengan Dictionaries

1. Buat sebuah file program baru, kemudian tuliskan kode berikut ini

```
1. mahasiswa = {'nama': 'Andika', 'umur': 21}
2. mahasiswa['umur'] = 19
3. mahasiswa['alamat'] = 'Sragen'
4. mahasiswa['angkatan'] = 2020
5. print(mahasiswa)
6. print(mahasiswa.pop('angkatan'))
7. print(mahasiswa)
8. print('nama' in mahasiswa)
9. print(len(mahasiswa))
10. print(sorted(mahasiswa))
11. mahasiswa.clear()
12. print(mahasiswa)
```

2. Amati hasilnya kemudian tulis analisis singkat mengenai kegiatan 3 ini

6.4 Tugas

1. Buat program python yang menerima inputan dari user berupa
 - Daftar nama 5 teman beserta no handphone
2. Simpan kedalam tipe data yang sesuai
3. Tampilkan output dalam bentuk berikut

Phone Book

1. [Nama 1] = [no hp 1]
2. [Nama 2] = [no hp 2]
3. [Nama 3] = [no hp 3]
4. [Nama 4] = [no hp 4]
5. [Nama 5] = [no hp 5]

BAB 7

Percabangan

7.1 Tujuan

9. Dapat menjelaskan bentuk-bentuk kontrol percabangan pada program.
10. Dapat menjelaskan alur kontrol percabangan If, If..else dan variasinya.
11. Dapat menggunakan kontrol percabangan If, If..else dan variasinya.

7.2 Pengantar

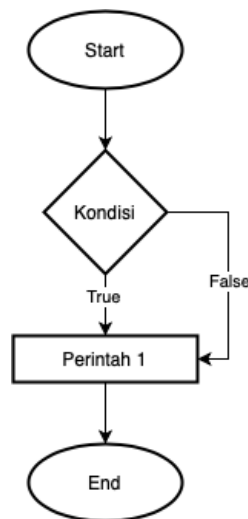
Pengambilan keputusan digunakan ketika kita ingin menjalankan kode jika ada suatu kondisi yang terpenuhi. Misalnya mencetak kata “Lulus” atau “Tidak Lulus” berdasarkan nilai ujian. Dalam bahasa pemrograman python, perintah untuk mengatur eksekusi kode dapat menggunakan beberapa cara seperti berikut.

7.2.1 IF

Perintah IF digunakan jika jalannya suatu kode didasarkan pada satu kondisi. Format penulisan IF adalah sebagai berikut

```
1. if kondisi:
2.     perintah 1
3.
```

Pada contoh format kode diatas, python akan mencari nilai dari kondisi. Jika kondisi bernilai True maka perintah 1 akan dijalankan. Begitupun sebaliknya, jika bernilai False maka perintah 1 tidak akan dijalankan. Perhatikan bahwa ada jarak indentasi pada penulisan perintah 1. Indentasi pada python merupakan tanda bahwa perintah 1 merupakan bagian dari IF. Gambar 7.1 berikut ini merupakan perintah IF dalam bentuk flowchart.



Gambar 7. 1 Flowchart Kondisi If

Perhatikan contoh kode berikut ini untuk penerapan perintah if

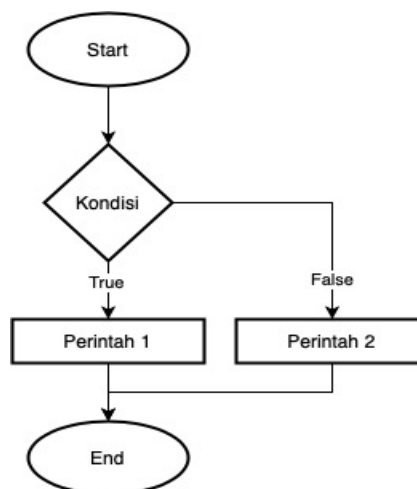
```
1. nilai = 90
2.
3. if nilai > 75:
4.     print('Kamu Lulus')
```

7.2.2 IF.. ELSE

Format perintah If..Else kita gunakan jika kita perlu mengeksekusi kode jika kondisi bernilai salah. Format dari perintah if..else adalah sebagai berikut

1. `if` kondisi:
2. Perintah 1
3. `else:`
4. Perintah 2

Pada contoh format diatas Perintah 1 akan dijalankan jika kondisi bernilai True sedangkan Perintah 2 akan dijalankan jika kondisi bernilai False. Gambar 7.2 berikut ini merupakan flowchart dari perintah if..else



Gambar 7. 2 Flowchart Kondisi If..Else

Perhatikan contoh kode berikut ini untuk penerapan perintah if..else

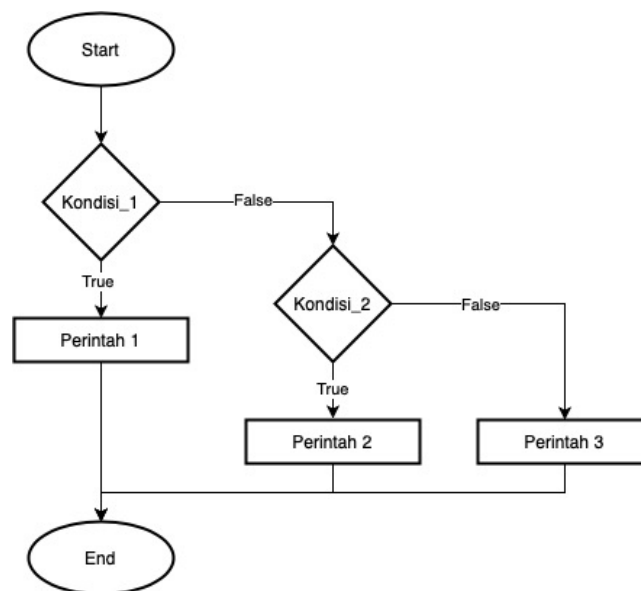
```
1. nilai = 60
2.
3. if nilai > 75:
4.     print('Kamu Lulus')
5. else:
6.     print('Kamu Mengulang')
```

7.2.3 IF..ELIF..ELSE

Sekarang bagaimana untuk program yang memerlukan seleksi lebih dari 2 kondisi. Maka kita menggunakan perintah yang ketiga yaitu if..elif..else. Format kodenya adalah sebagai berikut.

```
1. if kondisi_1:  
2.     Perintah 1  
3. elif kondisi_2:  
4.     Perintah 2  
5. else:  
6.     Perintah 3
```

Pada contoh kode diatas Perintah 1 dan Perintah 2 hanya dijalankan jika memenuhi masing-masing kondisi. Sedangkan perintah 3 akan dijalankan jika tidak memenuhi kondisi_1 dan kondisi_2. Gambar 7.3 berikut merupakan flowchart dari perintah if..elif..else.



Gambar 7. 3 Flowchart kondisi if elif else

Perhatikan contoh kode berikut ini untuk penerapan perintah if..elif..else

```
1. nilai = 80
2.
3. if nilai > 90:
4.     print('Kamu Lulus Beasiswa Penuh')
5. elif nilai >=80 :
6.     print('Kamu Mendapat Beasiswa 1')
7. elif nilai >=70 :
8.     print('Kamu Mendapat Beasiswa 2')
9. else:
10.    print('Tidak Mendapat Beasiswa')
```

7.2.4 IF Bersarang

Jika program kita sudah memerlukan seleksi kondisi yang lebih detail maka kita dapat membuat perintah If di dalam perintah if lainnya. Kondisi ini dinamakan if bersarang. Perhatikan contoh kode untuk menentukan apakah angka yang diinput oleh user bernilai positif atau negatif.

```
1. angka = float(input('Masukkan angka :'))
2.
3. if angka > 0:
4.     if angka == 0:
5.         print('Angka nol')
6.     else:
7.         print('Angka Positif')
8. else:
9.     print('Angka Negatif')
10.
```

7.2.5 Operator Logika pada IF

Selain IF bersarang kita dapat menggunakan operator logika AND dan OR untuk kondisi memerlukan seleksi lebih dari dua. Logika AND akan bernilai TRUE jika kedua kondisi bernilai benar. Sedangkan logika OR akan bernilai TRUE jika salah satu atau kedua kondisi bernilai benar. Perhatikan contoh penggunaan operator logika pada kode berikut ini

```
1. nilai = 60
2.
3. if nilai > 90:
4.     print('Nilai A')
5. if nilai >= 75 and nilai <= 89:
6.     print('Nilai B')
7. if nilai >= 60 and nilai <= 74:
8.     print('nilai C')
9. if nilai >= 45 and nilai <= 59:
10.    print('nilai D')
```

7.3 Kegiatan Praktikum

7.3.1 Kegiatan Praktikum 1 : Bekerja dengan IF ELSE

3. Buat sebuah file program baru, kemudian tuliskan kode berikut ini

```
1. angka = 5
2. if angka >= 0:
3.     print("Angka Positif atau Nol")
4. else:
5.     print("Angka Negatif")
```

4. Amati hasilnya kemudian tulis analisis singkat mengenai kegiatan 1 ini

7.3.2 Kegiatan Praktikum 2 : Bekerja dengan IF..ELIF..ELSE

3. Buat sebuah file program baru, kemudian tuliskan kode berikut ini

```
1. nohari = int(input('Masukkan no hari :'))
2.
3. if nohari = 1:
4.     print('Senin')
5. elif nohari = 2:
6.     print('Selasa')
7. elif nohari = 3:
8.     print('Rabu')
9. elif nohari=4:
10.    print('Kamis')
11. elif nohari = 5:
12.    print('Jumat')
13. elif nohari = 6:
14.    print('Sabtu')
15. elif nohari=7:
16.    print('Minggu')
17. else:
18.    print('Tidak ada no hari tersebut')
```

4. Amati hasilnya kemudian tulis analisis singkat mengenai kegiatan 2 ini

7.3.3 Kegiatan Praktikum 3 : Bekerja dengan IF Bersarang

3. Buat sebuah file program baru, kemudian tuliskan kode berikut ini

```
1. a, b, c = input('Masukkan 3 angka tanpa spasi: ')
2. if a > b:
3.     if b > c:
4.         maks = a
5.     else:
6.         if c > a:
7.             maks = c
```

```

8. elif a > c:
9.     maks = b
10. elif c > b:
11.     maks = c
12.
13. print(maks)

```

4. Amati hasilnya kemudian tulis analisis singkat mengenai kegiatan 3 ini

7.3.4 Kegiatan Praktikum 4 : Bekerja dengan Operator Logika

1. Buat sebuah file program baru, kemudian tuliskan kode berikut ini

```

1. harini = 'Minggu'
2.
3. if harini == 'Senin':
4.     print('Liburan Selesai, Saatnya Kerja')
5. elif harini == 'Minggu' or harini == 'Sabtu':
6.     print('Liburan')
7. else:
8.     print('Pergi Kerja')

```

2. Amati hasilnya kemudian tulis analisis singkat mengenai kegiatan 4 ini

7.4 Tugas

1. Buat sebuah program python untuk menentukan diskon tarif kereta api berdasarkan umur penumpang. Kondisi yang perlu di seleksi adalah
 - Usia 0 sampai 4 tahun mendapat diskon 100%
 - Usia 5 sampai 11 tahun mendapat diskon 50%
 - Usia 13 tahun keatas tidak mendapat diskon

Data yang perlu diinputkan adalah tahun kelahiran penumpang dan harga tiket kereta api

BAB 8

Perulangan

8.1 Tujuan

12. Dapat menjelaskan berbagai macam bentuk struktur kontrol perulangan yang ada pada python.
13. Dapat menggunakan struktur kontrol perulangan sederhana

8.2 Pengantar

Pada modul sebelumnya kita sudah belajar membuat program yang dieksekusi secara runtut dari baris pertama ke baris selanjutnya. Pada bagian ini kita mencoba untuk mengatur alur program agar dapat menjalankan suatu kode lebih dari satu kali. Metode ini dinamakan dengan perulangan.

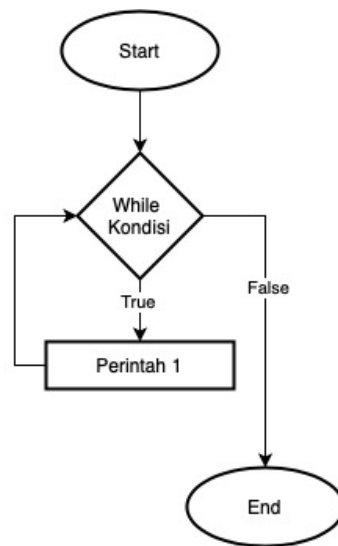
8.2.1 Perulangan dengan While

Perulangan menggunakan while mirip dengan perintah if yang akan mengeksekusi kode didalamnya jika pengecekan kondisi bernilai True. Sedangkan pada while, kode didalamnya akan dijalankan terus berulang-ulang selama kondisinya bernilai benar. Sehingga agar kode tersebut tidak berjalan secara terus menerus, maka harus ada

kode yang membuat kondisi di dalam while menjadi false. Format dari penulisan while adalah sebagai berikut

1. `while` kondisi:
2. Perintah1

Pada contoh format diatas, Perintah1 akan diulang terus menerus hingga kondisi menjadi bernilai False. Gambar 8.1 berikut ini merupakan perintah while dalam bentuk flowchart.



Gambar 8. 1 Flowchart Perintah While

Perhatikan contoh kode berikut untuk penerapan while.

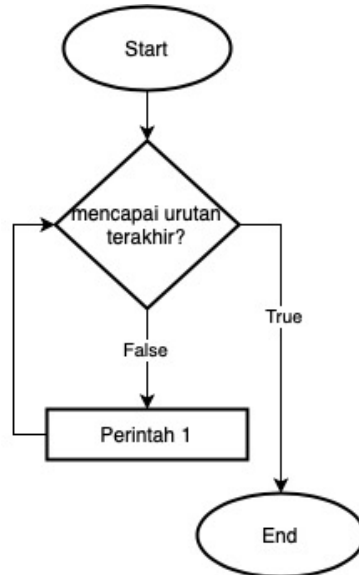
1. `angka = 0`
2. `while` `angka < 10`:
3. `print(angka, end= ' ')`
4. `angka = angka + 1`

8.2.2 Perulangan dengan For

Perulangan for pada prinsipnya merupakan iterasi pada sebuah urutan atau rangkaian. Iterasi pada urutan ini disebut dengan traversal. Format penulisan for adalah sebagai berikut.

```
1. for item in nama_urutan:  
2.     Perintah1
```

Pada contoh diatas, Perintah1 akan terus dijalankan hingga iterasi pada for mencapai urutan terakhir. Gambar 8.2 berikut ini merupakan flowchart dari perintah for.



Gambar 8. 2 Flowchart perintah for

Perhatikan contoh kode berikut untuk penerapan for

```
1. for i in range(5):  
2.     print(i)
```

pada kode diatas kita menggunakan fungsi baru yaitu range(). Fungsi ini digunakan untuk menghasilkan suatu angka. Maka jika ada fungsi range(5) maka dia akan menghasilkan angka dari 0 sampai 5.

8.2.3 Perulangan dengan else

Kita juga dapat menggunakan bantuan else pada perulangan. Kode pada bagian else akan dijalankan jika urutan yang digunakan pada perulangan sudah habis. Perhatikan contoh kode berikut ini untuk perulangan while

```
1. counter = 0
2. while counter <= 3:
3.     print (counter)
4.     counter = counter + 1
5. else:
6.     print('counter sampai batas ')
```

Output dari kode diatas adalah

```
0
1
2
3
counter sampai batas
```

Sedangkan contoh berikut ini untuk perulangan for

```
1. angka = [1,2,3]
2.
3. for i in angka:
4.     print(i)
5. else:
6.     print('Angka sudah habis')
```

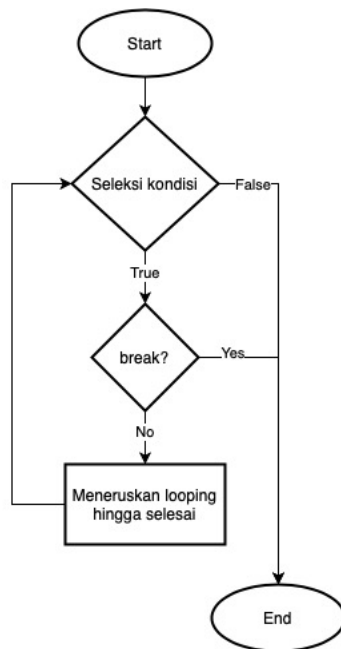
Output dari kode di atas adalah

```
1
2
3
Angka sudah habis
```

8.2.4 Break dan Continue

Pada umumnya perulangan akan terus dilakukan hingga kondisi menemui false atau urutan sudah habis. Namun kita dapat menghentikan proses perulangan di tengah jalan menggunakan perintah break dan continue

Perintah break digunakan untuk menghentikan proses perulangan dan melanjutkan eksekusi kode diluar perulangan. Gambar 8.3 berikut ini merupakan flowchart dari break



Gambar 8. 3 Flowchart break

Contoh break dapat dilihat pada kode berikut ini

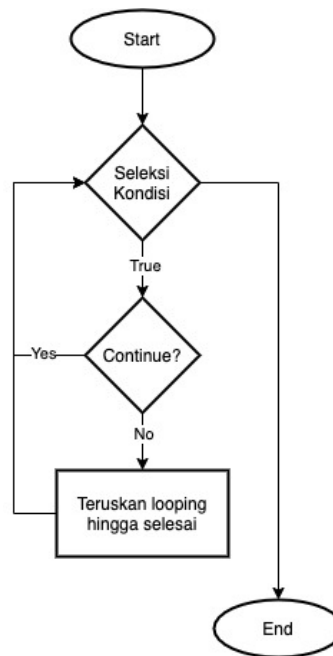
```
1. for i in range(10):  
2.     if i == 5:  
3.         break  
4.     print(i, end=' ')
```

perulangan pada kode diatas seharusnya akan dilakukan dari 0 hingga 10 namun pada baris ke dua kita cek jika perulangan sampai ke 5 maka akan dilakukan break atau keluar dari perulangan. Sehingga output dari kode diatas adalah

```
0 1 2 3 4
```

Berbeda dengan break, perintah continue tidak menghentikan perulangan. Tapi digunakan untuk melewati kode pada iterasi tersebut kemudian melanjutkan

perulangan ke iterasi selanjutnya. Gambar 8.4 berikut ini merupakan flowchart dari continue



Gambar 8. 4 Flowchart continue

Contoh continue dapat dilihat pada kode berikut ini

```
1. for i in range(10):  
2.     if i == 5:  
3.         continue  
4.     print(i, end=' ')
```

perulangan pada kode diatas akan menghasilkan output

```
0 1 2 3 4 6 7 8 9
```

Dapat dilihat dari output diatas tidak ada angka 5 karena perintah continue akan melewati kode pada iterasi tersebut. Pada contoh diatas iterasi yang dilewati adalah jika i sama dengan 5.

8.3 Kegiatan Praktikum

8.3.1 Kegiatan Praktikum 1 : Perulangan While

5. Buat sebuah file program baru, kemudian tuliskan kode berikut ini

```
1. counter = 0
2. total = 0
3.
4. jumlah_barang = int(input('Masukkan Jumlah Barang : '))
5. while counter < jumlah_barang:
6.     harga = int(input('Masukkan harga barang ke-
   {}: '.format(counter+1)))
7.     jumlah = int(input('Masukkan jumlah barang ke-
   {}: '.format(counter+1)))
8.     total = total + (harga*jumlah)
9.     counter = counter+1
10.
11. print('Total Harga Barang :',total)
```

6. Amati hasilnya kemudian tulis analisis singkat mengenai kegiatan 1 ini

8.3.2 Kegiatan Praktikum 2 : Perulangan For

5. Buat sebuah file program baru, kemudian tuliskan kode berikut ini

```
1. for i in range (1,10):
2.     print(i, end=' ')
3.
4. print(' ')
5. for j in range(10,0,-1):
6.     print(j, end=' ')
7.
8. print(' ')
9. for k in range(1,20,3):
10.    print(k, end=' ')
11.
```

```

12. print(' ')
13. for l in range(1, 20, 2):
14.     print(l, end=' ')

```

- Amati hasilnya kemudian tulis analisis singkat mengenai kegiatan 2 ini

8.3.3 Kegiatan Praktikum 3 : Mencari bilangan prima

- Buat sebuah file program baru, kemudian tuliskan kode berikut ini

```

1. i = 2
2. while(i < 30):
3.     j = 2
4.     while(j <= (i/j)):
5.         if not(i%j): break
6.         j = j + 1
7.     if (j > i/j) : print (i, " adalah bilangan prima")
8.     i = i + 1

```

- Amati hasilnya kemudian tulis analisis singkat mengenai kegiatan 3 ini

8.4 Tugas

- Buat sebuah program python untuk menampilkan lirik lagu 'Anak Bebek', dengan inputan jumlah anak bebek

Masukkan jumlah anak bebek : 5
 Anak bebek turun 5, mati satu tinggal 4
 Anak bebek turun 4, mati satu tinggal 3
 Anak bebek turun 3, mati satu tinggal 2
 Anak bebek turun 2, mati satu tinggal 1

BAB 9

Fungsi pada Python

9.1 Tujuan

14. Dapat menjelaskan manfaat pemrograman modular
15. Dapat mengimplementasikan fungsi dalam bahasa python

9.2 Pengantar

Pada python, fungsi merupakan sekumpulan perintah yang menjalankan operasi tertentu. Fungsi membantu kita untuk memecah kode program menjadi lebih kecil dan modular. Fungsi juga membantu kode kita menjadi lebih terorganisir, mengurangi pengulangan kode dan lebih mudah di kelola. Format dari penulisan fungsi pada python adalah sebagai berikut :

```
1. def namafungsi():  
2.     perintah  
3.     return nilai
```

perintah **return** digunakan apabila fungsi tersebut menghasilkan suatu nilai. Berikut ini contoh penulisan fungsi dan pemanggilannya:

```

1. def halo():
2.     print('Halo, selamat datang di PTI')
3.     return
4.
5. halo()

```

Baris 1 merupakan deklarasi fungsi dengan nama **halo()** yang berisi perintah mencetak ke layar tulisan 'Halo, selamat datang di PTI'. Pada baris ke 3 perintah **return** tidak diberikan nilai apapun. Baris ke 5 merupakan pemanggilan **fungsi halo()**.

9.2.1 Fungsi dengan parameter

Fungsi juga dapat menerima parameter untuk diolah di dalam blok perintahnya. Perhatikan contoh kode berikut ini:

```

1. def halo(name):
2.     print('Halo', name, ', selamat datang di PTI')
3.     return
4.
5. halo('Doni')

```

Pada fungsi **halo()** diatas sekarang kita menerima 1 parameter yaitu name. Parameter name kemudian akan di cetak ke layar seperti pada kode baris ke 2. Sehingga untuk memanggil fungsi **halo()** sekarang kita harus memasukkan paramater name yang bertipe string. Kode diatas akan menghasilkan output seperti berikut :

```
Halo Doni , selamat datang di PTI
```

9.2.2 Fungsi dengan multi parameter

Tidak hanya satu parameter, kita dapat menggunakan dua atau lebih parameter dalam suatu fungsi. Untuk membedakan parameter satu dengan lainnya maka kita harus menggunakan tanda koma (,). Perhatikan contoh kode berikut ini

```

1. def perkalian(angka1, angka2):
2.     return angka1 * angka2
3.

```

```
4. hasil = perkalian(2, 3)
5. print(hasil)
```

pada kode diatas fungsi **perkalian()** menerima dua parameter yaitu angka1 dan angka2 serta mengembalikan hasil perkalian dari kedua parameter tersebut. Pada baris ke-4 kita panggil fungsi perkalian dengan hasil **return** disimpan ke dalam variabel hasil kemudian dicetak ke layar.

9.2.3 Modul pada python

Modul merupakan file python yang berisi perintah dan fungsi. Modul biasa digunakan untuk memecah program yang besar menjadi program kecil yang lebih mudah dikelola. Kita dapat membuat fungsi dalam sebuah modul kemudian mengimportnya kedalam kode kita, sehingga kita tidak perlu menulis ulang seluruh fungsi tersebut. Sebagai contoh buat kode berikut kemudian simpan dengan nama file hitung.py
Nama File : hitung.py

```
1. def perkalian(angka1, angka2):
2.     hasil = angka1 * angka2
3.     return hasil
```

Pada kode diatas kita sudah membuat fungsi perkalian di dalam sebuah modul yang bernama hitung. Untuk menggunakan modul tersebut kita harus melakukan import pada file python utama kita.

Nama file : main.py

```
1. import hitung
2.
3. kali = hitung.perkalian(2, 3)
4. print(kali)
```

baris 1 merupakan cara kita melakukan import modul. Pada baris ke-3, kita menggunakan fungsi perkalian yang ada pada modul hitung dengan menggunakan tanda titik (.)

9.3 Kegiatan Praktikum

9.3.1 Kegiatan Praktikum 1 : Fungsi dengan parameter

7. Buat sebuah file program baru, kemudian tuliskan kode berikut ini

```
1. def greet(name, msg='Selamat Pagi'):  
2.     print("Halo", name + ', ' + msg)  
3.  
4.  
5. greet("Batman")  
6. greet("Robin", "Mau pergi kemana?")
```

8. Amati hasilnya kemudian tulis analisis singkat mengenai kegiatan 1 ini

9.3.2 Kegiatan Praktikum 2 : Fungsi dengan parameter list

7. Buat sebuah file program baru, kemudian tuliskan kode berikut ini

```
1. def maks(a):  
2.     m = a[0]  
3.     for i in a:  
4.         if m < i:  
5.             m = i  
6.  
7.     return m  
8.  
9.  
10. print(maks([5, 2, 1, 4]))
```

8. Amati hasilnya kemudian tulis analisis singkat mengenai kegiatan 2 ini

9.3.3 Kegiatan Praktikum 3 : Modul pada python

7. Buat sebuah file program baru, kemudian tuliskan kode berikut ini

```
1. from math import log10, factorial
2.
3. print(log10(100))
4. print(factorial(4))
5.
6. import math
7.
8. print(math.pow(5, 2))
9. print(math.sqrt(25))
```

8. Amati hasilnya kemudian tulis analisis singkat mengenai kegiatan 3 ini

9.4 Tugas

1. Buatlah sebuah fungsi bernama `apakah_prima` yang menerima sebuah parameter bilangan bulat. Jika nilai dari parameter merupakan bilangan prima maka cetak tampilan “Bilangan Prima”, jika bukan bilangan prima cetak tampilan “Bukan Bilangan Prima”

Operasi File pada Python

10.1 Tujuan

16. Dapat menjelaskan operasi open, write, append dan close pada bahasa python.
17. Dapat mengimplementasikan operasi open, write, append dan close pada pemrograman

10.2 Pengantar

Jika aplikasi yang sudah kita buat semakin besar dan kompleks maka penyimpanan data akan menjadi suatu perhatian khusus. Data tidak bisa kita simpan lagi ke dalam suatu variabel karena akan membuat aplikasi menjadi semakin rumit. Python menyediakan fungsi operasi file sehingga kita dapat menyimpan data ke dalam piranti penyimpanan seperti hard disk. Operasi file pada python bekerja layaknya manusia biasa. Jika kita ingin membaca file atau menulis data ke dalam suatu file, maka pertama-tama kita buka file tersebut atau jika belum ada maka kita harus membuat file tersebut. Kemudian melakukan penulisan ke dalam file, menyimpannya dan kemudian menutupnya.

10.2.1 Membuka file pada python

Python menyediakan fungsi bawaan yaitu **open()** untuk membuka suatu file. Fungsi ini akan mengembalikan objek yang dapat kita gunakan untuk membaca dan menulis file tersebut.

Contoh penggunaan fungsi **open()** adalah seperti berikut:

```
1. f = open("test.txt")      # jika file terletak pada direktori yang sama
2. f = open("C:/Python/README.txt") # jika file terletak pada direktori
    tertentu
3. f = open("C:/Python/README.txt", "r+") #membuka file untuk dilakukan
    modifikasi
```

Pada baris 1, kita menggunakan fungsi **open()** jika file terletak pada direktori yang sama. Jika file terletak pada direktori yang berbeda kita menggunakan baris ke-2. Kita juga dapat menentukan mode apa yang akan digunakan pada saat membuka file. Pada baris ke-3 kita menggunakan mode (r+) untuk membaca dan menulis pada file yang sama. Mode-mode lain yang tersedia dapat dilihat pada tabel 10.1 dibawah

Tabel 10. 1 Mode Open File

Mode	Keterangan
r - read mode	Fungsi default. Digunakan hanya untuk membaca data
w – write mode	Digunakan untuk menulis data atau mengubah data pada file. Mode ini akan menghapus isi file sebelumnya
a – append mode	Digunakan untuk menambahkan data pada file
r+ - read or write mode	Digunakan untuk membaca dan menulis data pada file yang sama
a+ - appen or read mode	Digunakan untuk membaca dan menambah data pada file yang sama

10.2.2 Membaca file pada python

Untuk membaca file pada python kita harus membuka suatu file menggunakan mode reading (r). Terdapat tiga fungsi yang dapat kita gunakan yaitu **read()**, **readline()** dan **readlines()**.

Kode berikut ini merupakan contoh membaca file menggunakan python:

```
1. f = open("test.txt", 'r', encoding = 'utf-8')
2. print(f.read(5))
3. print(f.read())
4. print(f.readline())
5. print(f.readlines())
```

fungsi **read(5)** pada baris kedua akan menampilkan 5 karakter pertama pada file test.txt. sedangkan jika kita tidak memberikan parameter seperti pada baris ketiga maka akan menampilkan isi semua file tersebut.

Sedangkan fungsi **readline()** akan membaca baris pertama pada file test.txt sedangkan fungsi **readlines()** akan membaca semua baris yang ada pada file tersebut.

10.2.3 Menulis file pada python

Agar dapat menulis file pada python kita harus membuka file dalam mode write (w), append (a) atau exclusive (x) mode. Hati-hati jika menggunakan mode (w) karena pada mode ini akan menghapus semua isi file sebelumnya. Sedangkan untuk operasi menulis kita dapat menggunakan fungsi **write()**. Berikut ini merupakan contoh penggunaan fungsi write:

```
1. f = open("C:/Documents/Python/test.txt", "w")
2. f.write("Hello World\n")
3. f.write("Hello Python")
```

pada kode diatas kita menggunakan mode (w) untuk menulis data ke dalam file. Tanda (\n) digunakan untuk berpindah baris. Jika ingin menambah data tanpa menghapus isi file maka kita gunakan mode (a). sehingga kode diatas akan menjadi seperti berikut:

```
1. f = open("C:/Documents/Python/test.txt", "a+")
2. f.write("PTI UMS\n")
```

10.2.4 Menutup file pada python

Untuk dapat menutup file, maka pastikan file tersebut sudah terbuka di python. Terdapat fungsi close() yang dapat kita gunakan untuk menutup file yang terbuka. Setiap kali kita membuka file, pastikan kita akhiri dengan menutupnya. Karena jika kita tidak menutup file maka data apapun yang sudah kita tulis tidak akan tersimpan pada file tersebut. Berikut ini contoh kode penggunaan fungsi close():

```
1. f = open("C:/Documents/Python/test.txt", "a+")
2. f.write("PTI UMS\n")
3. f.close()
```

Kita juga dapat menggunakan perintah with. File yang kita buka menggunakan with akan secara otomatis tertutup setelah perintah yang berada di dalam blok with selesai dieksekusi sehingga kita tidak perlu menuliskan fungsi close() lagi. Berikut ini merupakan contoh penggunaan perintah with:

```
1. with open("C:/Documents/Python/test.txt", "a+") as f:
2.     f.write("PTI\n")
3.     f.write("UMS")
```

10.3 Kegiatan Praktikum

10.3.1 Kegiatan Praktikum 1 : Membaca dan menulis File

9. Buat sebuah file program baru, kemudian tuliskan kode berikut ini

```
1. fileku = open("buah.txt", "w")
2. fileku.write('Apel\n')
3. fileku.write('Mangga\n')
4. fileku.write('Jambu')
5. fileku.close()
6.
7. bacafileku = open("buah.txt", "r")
8. print(bacafileku.read())
```

10. Amati hasilnya kemudian tulis analisis singkat mengenai kegiatan 1 ini

10.3.2 Kegiatan Praktikum 2 : Menambah data pada file

9. Buat sebuah file program baru, kemudian tuliskan kode berikut ini

```
1. fileku = open("jurusan.txt", "w")
2. fileku.write('Biologi\n')
3. fileku.write('PGSD\n')
4. fileku.write('PTI\n')
5. fileku.close()
6.
7. with open("jurusan.txt", 'a+') as f:
8.     f.write('Akuntansi\n')
9.     f.write('PAUD')
10.
11. with open("jurusan.txt", 'r') as baca:
12.     print(baca.read())
```

10. Amati hasilnya kemudian tulis analisis singkat mengenai kegiatan 2 ini

10.3.3 Kegiatan Praktikum 3 : Attribute pada file

9. Buat sebuah file program baru, kemudian tuliskan kode berikut ini

```
1. fileku = open("kereta.txt", "w")
2. fileku.write('Prameks\n')
3. fileku.write('Joglosemarkerto\n')
4. fileku.write('Sancaka\n')
5. fileku.close()
6.
7. my_file = open("kereta.txt", "a+")
8. print("Filamenya adalah : ", my_file.name)
9. print("File modenya adalah", my_file.mode)
10. print("Encoding filenya adalah", my_file.encoding)
11. print("Apakah file sudah ditutup?", my_file.closed)
12. my_file.close()
13. print("Apakah file sudah ditutup", my_file.closed)
```

10. Amati hasilnya kemudian tulis analisis singkat mengenai kegiatan 3 ini

10.4 Tugas

1. Buatlah program python yang berfungsi layaknya program pada kasir toko. Program akan menerima inputan jumlah barang dan menampilkan total harga barang pada layar. Jika transaksi telah selesai maka program akan membuat sebuah file bernama invoice.txt yang berisi rincian belanja

DAFTAR PUSTAKA

- Hunt, J. (2019). *A Beginners Guide to Python 3 Programming*. In Springer
- Romano, Fabrizio. (2015). *Learning Python*. Packt Publishing
- Swastika, W. (2019). *Pengantar Algoritma dan Penerapannya pada Python*. Ma Chung Press.
- Wadi, H. *Pemrograman Python untuk Mahasiswa dan Pelajar*. TR Publisher
- Python File Handling Tutorial: How to Create, Open, Read, Write, Append*. (n.d.). Retrieved September 7, 2020, from <https://www.softwaretestinghelp.com/python/python-file-reading-writing/>
- Learn Python Programming*. (n.d.). Retrieved September 7, 2020, from <https://www.programiz.com/python-programming>
- Tutorial Pemrograman Python*. (n.d.). Retrieved September 5, 2020, from <https://www.petanikode.com/tutorial/python/>
- PY4E - Python for Everybody*. (n.d.). Retrieved September 2, 2020, from <https://www.py4e.com/>
- Google's Python Class | Python Education | Google Developers*. (n.d.). Retrieved September 9, 2020, from <https://developers.google.com/edu/python/>
- Learn Python the Hard Way*. (n.d.). Retrieved September 2, 2020, from <https://learnpythonthehardway.org/book/>
- Python Programming Tutorials*. (n.d.). Retrieved September 9, 2020, from <https://pythonprogramming.net/python-fundamental-tutorials/>

LAMPIRAN

Laporan Sementara

Laporan sementara dilakukan dengan langkah sebagai berikut:

1. Buatlah screenshot atau gambar dari hasil kegiatan yang dilakukan.
2. Ganti nama file dengan format: bab-kegiatan-no.gambar-nim.jpg Misalnya file adalah gambar ke 1 pada Bab 1, Kegiatan 1, dan nim anda adalah A40009001, maka nama filenya adalah **1-1-1-A40009001.jpg**
3. Simpan file tersebut kemudian tempatkan pada folder sesuai instruksi dosen/asisten praktikum.

Laporan Praktikum

Laporan ditulis dalam kertas putih ukuran A4. Sedangkan urutan susunan laporan adalah sebagai berikut:

1. Cover depan: Berwarna sama dengan cover modul praktikum
2. Halaman Cover: Contoh dapat di-download di <http://bit.ly/lap-alpro>
3. Kata Pengantar
4. Daftar isi
5. Laporan tiap modul (1-10) sesuai dengan format terlampir
6. Penulis: berisi biodata penulis (disertai foto), pesan dan kesan, kritik dan saran demi kemajuan praktikum berikutnya.

Format Laporan Tiap Bab

Matakuliah : Algoritma dan Pemrograman	Acc
NIM :	
Nama :	
Tgl. Prakt.:	Tgl:

BAB I **Judul**

1. Dasar Teori

300 sampai dengan 350 kata

2. Tujuan

3. Analisa Hasil

3.1. Kegiatan 1: ...

Tampilkan hasil praktikum berupa kode yang dibuat atau hasil output, kemudian berikan analisisnya. Jika terdapat gambar, berikan juga nomor gambar.

3.2. Kegiatan 2: ...

4. Penyelesaian Tugas

Jika terdapat tugas yang dikerjakan, tuliskan disini langkah pengerjaan dan hasilnya.

5. Kesimpulan

Berikan kesimpulan yang didapatkan setelah anda menyelesaikan praktikum