LAPORAN
PRAKTIKUM ALGORITMA DAN PEMROGRAMAN

Disusun Oleh :

Nama : Ahmad Hafizuddin

NIM : 237023046

Kelas : A2 T1

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS KRISNADWIPAYANA

2023

KATA PENGANTAR

Segala puji bagi Allah yang telah memberikan kemudahan sehingga dapat menyelesaikan laporan ini. Tanpa pertolongan-Nya mungkin saya tidak akan sanggup menyelesaikannya dengan baik. Sholawat dan salam semoga terlimpah curahkan kepada baginda tercinta kita yakin Nabi Muhammad SAW.

Selama pembuatan laporan ini, penulis mendapat banyak masukan dan bimbingan dari berbagai pihak sehingga tidak lupa penulis mengucapkan terima kasih sebesar-besarnya kepada pihak yang terkait. Penulis menyadari bahwa di dalam Laporan Algoritma Pemrograman ini masih terdapat banyak kesalahan dan jauh dari kata sempurna. Maka dari itu, penulis memohon maaf apabila terdapat kesalahan kata maupun penulisan di dalam Laporan Algoritma Pemrograman ini.

Bekasi, November 2023

Ahmad Hafizuddin

DAFTAR ISI

BAB I

PENDAHULUAN

Latar Belakang

Maraknya penggunaan komputer di berbagai bidang kehidupan manusia, menuntut setiap orang untuk mengetahui dan mempelajari berbagai macam *software* pendukung yang dapat beroperasi pada komputer tersebut. Komputer merupakan salah satu penemuan tercanggih pada abad ini.

Kemampuan komputer dalam melakukan perhitungan yang sangat cepat, dapat mempermudah *manusia* atau *user* dalam dalam mengoperasikannya. Berbagai macam media peendukung untuk mengoptimalkan kinerja komputer banyak dibuat. Media tersebut berupa *software* atau program aplikasi yang hubungannya tidak dapat terpisahkan dari komputer.

Software atau program aplikasi tersebut dibangun dengan menggunakan *software* lain. Banyak software yang digunakan untuk membangun *software* atau program aplikasi. Salah satunya adalah bahas pemrograman (*programming language*).

Dalam laporan ini, akan membahas mengenai Bahasa Pemrograman Python untuk memberikan sedikit gambaran kepada masyarakat tentang Bahasa Pemrograman Python, mengingat bahasa pemrograman ini masih sangat jarang digunakan.

Tujuan

Adapun tujuan dari pembuatan laporan ini adalah sebagai berikut.

1. Untuk mengetahui konsep dasar Python.
2. Untuk mengetahui sejarah Python.
3. Untuk mengetahui dasar – dasar pemrograman Python.

BAB II

LANDASAN TEORI

Konsep Dasar Python

Pada awalnya, motivasi pembuatan bahasa pemrograman ini adalah untuk bahasa skrip tinggi pada sistem operasi terdistribusi Amoeba. Bahasa pemrograman ini menjadi umum digunakan untuk kalangan *engineer* seluruh dunia dalam pembuatan perangkat lunaknya, bahkan beberapa perusahaan menggunakan python sebagai pembuat perangkat lunak komersial.

Python merupakan bahasa pemrograman yang *freeware* atau perangkat bebas dalam arti sebenarnya, tidak ada batasan dalam penyalinannya atau mendistribusikannya. Lengkap dengan *source* codenya, *debugger* dan *profier*, antarmuka yang terkandung di dalamnya unutk pelayanan antarmuka, fungsi sistem, GUI (antarmuka pengguna grafis), dan basis datanya.

Sejarah 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. Saat ini distribusi Python sudah mencapai versi 2.6.1 dan versi 3.0 keatas. Nama Python dipilih oleh Guido sebagai nama bahasa ciptaanya 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.

BAB III

PEMBAHASAN

MODUL 1
PENGENALAN PYTHON

Pengenalan Python

Bahasa Pemograman yang akan Anda pelajari adalah Python, Python termasuk dari jajaran bahasa pemograman tingkat tinggi, lainnya Anda mungkin mengenal bahasa pemograman C, C++, Java, Perl dan Pascal.

Bilamana terdapat bahasa pemograman tingkat tinggi, juga dikenal bahasa pemogramantingkat rendah, yang dikenal sebagai bahasa mesin yaitu bahasa pemograman Assembly, Kenyataannya Komputer hanya dapat mengeksekusi bahasa tingkat rendah, jadi bahasa pemograman tingkat tinggi harus melewati beberapa proses untuk diubah ke bahasa pemograman tingkat rendah, hal tersebut merupakan kelemahan yang tidak berarti bagi bahasa pemograman tingkat tinggi.

Tetapi kekurangan tersebut tidak sebanding dengan kelebihannya. Pertama, lebih mudah memprogram sebuah aplikasi dengan bahasa tingkat tinggi. Lebih cepat, lebih mudah

dimengerti menulis program komputer dengan bahasa tingkat tinggi, dan juga kesalahan dalam penulisan program cenderung tidak mengalami kesalahan yang berarti. Kedua bahasa pemograman tingkat tinggi lebih portable dalam arti bisa digunakan untuk menulis di berbagai jenis arsitektur komputer (seperti Intel 386, 486, 586, SPARC, RISC/6000) yang berlainan dengan sedikit modifikasi ataupun tidak memerlukan modifikasi sama sekali. Bahasa pemograman tingkat rendah hanya dapat berjalan di satu jenis arsitektur komputer dan harus ditulis ulang untuk menjalankannya di lain mesin, hal ini dikarenakan karena perbedaan urutan register dan services -servicesnya.

Dengan keuntungan keuntungan tersebut, kebanyakan aplikasi - aplikasi komputer di tulis dengan bahasa pemograman tingkat tinggi. Penggunaan bahasa pemograman tingkat rendah hanya digunakan di aplikasi - aplikasitertentu.

Terdapat 2 jenis aplikasi untuk memproses bahasa tingkat tinggi ke bahasa tingkat rendah, yaitu : compiler dan interpreter. Sebuah interpreter membaca sebuah program yang ditulis dengan bahasa tingkat tinggi dan langsung menjalankannya per baris, memakan waktu sedikit.

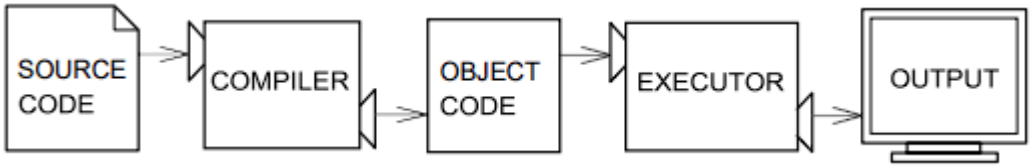
Terdapat 2 jenis aplikasi untuk memproses bahasa tingkat tinggi ke bahasa tingkat rendah, yaitu : compiler dan interpreter. Sebuah interpreter membaca sebuah program yang ditulis dengan bahasa tingkat tinggi dan langsung menjalankannya per baris, memakan waktu sedikit.

Gambar: Interpreter

Sebuah kompiler menerjemahkan keseluruhan kode program sebelum menjalankan program tersebut. Dalam kasus ini kode tersebut disebut sebagai source code dan program yang diterjemahkan disebut dengan object code atau executable. Sekali program tersebut di

kompilasikan, Anda dapat mengeksekusinya berulang kali tanpa menerjemahkannya lagi kedalam object code.

Gambar: Kompiler



Bekerja pada modus baris perintah sangat baik untuk membuat program dan untuk mencoba - coba algoritma, karena Anda dapat langsung menjalankan perintah tersebut dan melihat hasilnya. Tetapi pada saat Anda ingin membuat program atau aplikasi yang real, Anda seharusnyamenyimpan ke dalam bentuk script, jadi dapat Anda jalankan dan dimodifikasi untuk pengembangan program selanjutnya. Kata tercadang atau sering di sebuy *reserved – word* adalah kata-kata yang digunakan oleh Python dengan makna khusus. Kata-kata seperti ini tidak dapat diubah maknanya. Daftar *reserved- word* pada Python :

and	if
assert	import
break	in
class	is
continue	lambda
def	not
elif	or
else	pass
except	print
exec	raise
finally	return
for	try
from	while
global	

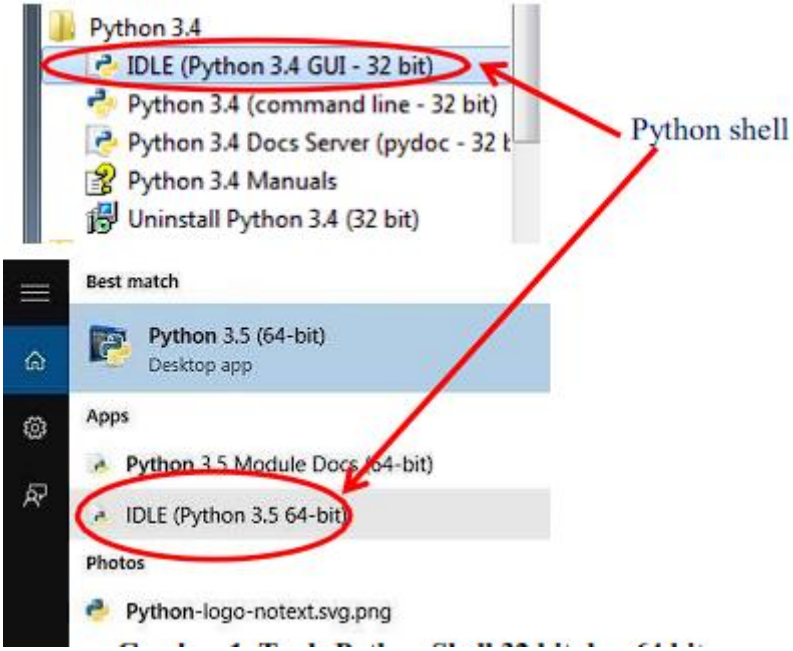
Aturan dalam memberikan nama untuk pengenal (identifier) adalah seperti berikut:

1. Dapat melibatkan huruf (A-Z,a-z), digit (0,9), dam garis bawah (_).
2. Tidak boleh berawalan dengan digit .
3. Huruf kecil dan huruf kapital dibedakan,

- Tidak menggunakan *Reserved – word* Misal:

x, N, kuartal_2 dan Kuartal_2

Membuka Python



Gambar 1 memperlihatkan tampilan IDLE (Integrated Development and Learning Environment) yaitu Python sebagai lingkungan belajar berisi tampilan GUI yang menarik, bekerja pada OS (Windows, Linux dan Mac OS X), interaktif interpreter (penterjemah) berupa kode *input/output dan error messages, multi windows, multiple file* (grep) berupa *search within any windows, fiture debugger* (pencari kesalahan), konfigurasi/*browsers* dan dialog.

Program Latihan

1. **Latihan**

Biasanya, dalam mempelajari sebuah bahasa pemograman komputer, selalu diawali dengan program yang terkenal dengan program "Hello World!". Dalam Python, program tersebut dapat langsung dijalankan sebagai berikut:

```
print("helo dunia!!!!!!!")
print("apa kabar kalian?")
print("instalasi berhasil")
```

2. Alur Python

Sebuah program adalah sejumlah instruksi yang berisi perintah - perintah dalam bahasa pemograman komputer untuk menyelesaikan masalah dengan bantuan komputer. Masalah - masalah komputasi tersebut mungkin seperti permasalahan matematika, seperti menyelesaikan sebuah fungsi eksponen, rumus- rumus dalam matematika, tetapi dapat juga berupa mencari dan menggantikan teks, menyusun teks dalam dokumen, dan sebagainya.

```
import time
start_time = time.time()
print("Hello")
print("World")
print("Hello World")
print("Halo Chantiiek")
# ini adalah comment
a = 10
print(a)
print(time.time() - start_time, "detik")
```

3. Variable

Fitur yang paling kuat dalam sebuah bahasa pemograman komputer adalah kemampuan untuk memanipulasi variabel - variabel. Sebuah variabel adalah sebuah nama yang mempunyai sebuah nilai. Pengdeklarasian kalimat membuat sebuah variabel - variabel baru dan memberinya nilai.

```
>>> pesan = "nasi goreng satu!"
>>> banyak = 4
>>> phi = 3.14159
```

Pada contoh di atas, pendeklarasian tersebut menciptakan 3 variabel baru. Pendeklarasian pertama, menunjukkan string "nasi goreng satu!" ke sebuah variabel yang bernama pesan. Kedua, variabel banyak diberi nilai 4 sebagai integer. Dan yang terakhir variabel phi diberi nilai 3.14159 sebagai nilai pecahan.

Cara yang umum untuk pemberian nama variabel adalah dengan tanda panah menunjuk ke nilai variabel tersebut. Jenis ini dinamai dengan state diagram karena menunjukkan nilai - nilai yang merupakan nilai dari variabel - variabel tersebut, contohnya :

```
# Variabel adalah tempat menyimpan data
# menaru / assignment nilai
a = 10
x = 5
panjang = 1000
# pemanggilan pertama
print("Nilai a = ", a)
print("Nilai x = ", x)
print("Nilai panjang = ", panjang)
# penamaan
nilai_y = 15 # dengan menggunakan underscore
jutal0 = 10000000 # ini boleh
nilaiZ = 17.5 # ini boleh
# pemanggilan kedua
print("Nilai a = ", a)
a = 7
print("Nilai a = ", a)
# assignment indirect
b = a
print("Nilai b = ", b)
```

4. Tipe Data

Sebuah nilai adalah hal yang paling mendasar seperti sebuah huruf atau sebuah angka yang akan di manipulasi oleh program. Nilai yang selama ini kita lihat adalah 2 (hasil yang kita dapat, ketika kita menambahkan 1 + 1), dan "HelloWorld!".

Nilai - nilai tersebut berbeda tipe data, yakni 2 sebagai sebuah integer, dan "Hello World!" sebagai sebuah string, disebutstring, karena terdiri darisebuah kata yang terdiri dari beberapa huruf - huruf. Anda dapat mengidentifikasikan string karena mereka di dalam tanda kutip dua(").

Perintah print juga dapat menampilkan integer

>>>

print 4

4

Bila Anda tidak yakin dengan tipe data yang Anda sebutkan, interpreter dapat memberitahu Anda yaitu dengan menggunakan fungsi built_in type() yang ada bersama interpreter.

```
>>> type ("Hello World!")
 <'type string'>
 >>> type 5
 <'type int'>
```

Lebih lanjut, angka desimal dengan tanda (.) dibelakang angka dikenal dengan bilangan pecahan atau float karena angka tersebut merepresentasikan suatu bentuk dengan nama floating point.

```
>>> type (6.5)
 <'type float'>
```

Bagaimana dengan nilai "17.5" dan "5"? Mereka seperti angka - angka, tetapi mereka berada di dalam tanda kutip ("), nah! berarti mereka adalah string.

```
>>> type ("17.5")
<'type string'>
```

Pada saat Anda ingin menuliskan nilai integer yang besar, Anda mungkin menggunakan koma diantara 3 kelompok digit, seperti 1,000,000. Angka tersebut bukan integer yang baik di Python, tetapi itu bisa dilakukan di Python:

```
>>> print
1,000,000 1 0 0
```

Itu bukan tampilan yang kita harapkan bukan? contoh di atas menunjukan bahwa 1,000,000 adalah sebuah tuple (larik / baris), kita akan membahas hal tersebut di bab selanjutnya, jadi sekarang jangan lupa untuk tidak menempatkan koma pada integer

- integer Anda.

Berikut adalah tipe data dari bahasa pemrograman Python :

Tipe Data	Contoh	Penjelasan
Boolean	True atau False	Menyatakan benar True yang bernilai 1 , atau salah False yang bernilai 0
String	“Ayo belajar Python”	Menyatakan karakter/kalimat bisa berupa huruf angka, dll (diapit tanda " atau ')

Integer	25 atau 1209	Menyatakan bilangan bulat
Float	3.14 atau 0.99	Menyatakan bilangan yang mempunyai koma (,)
Hexadecimal	9a atau 1d3	Menyatakan bilangan dalam format heksa (bilangan berbasis 16)
Complex	1 + 5j	Menyatakan pasangan angka real dan imajiner
List	[‘xyz’, 786, 2.23]	Data untaian yang menyimpan berbagai tipe data dan isinya bisa diubah-ubah
Tuple	[‘xyz’, 786, 2.23]	Data untaian yang menyimpan berbagai tipe data tapi isinya tidak bisa diubah
Dictionary	{‘nama’: ‘adi’,’id’:2}	Data untaian yang menyimpan berbagai tipe data berupa pasangan penunjuk dan nilai

Untuk mencoba berbagai macam tipe data, silahkan coba script Python dibawah ini.

Contoh Program:

```
#tipe data Boolean
print(True)
#tipe data String
print("Ayo belajar Python")
print('Belajar Python Sangat Mudah')
#tipe data Integer
print(20)
#tipe data Float
print(3.14)
#tipe data Hexadecimal
print(9a)
#tipe data Complex
print(5j)
#tipe data List
print([1,2,3,4,5])
print(["satu", "dua", "tiga"])
#tipe data Tuple
print((1,2,3,4,5))
print(("satu", "dua", "tiga"))
#tipe data Dictionary
print({"nama":"Budi", 'umur':20})
#tipe data Dictionary dimasukan ke dalam variabel biodata
biodata = {"nama":"Andi", 'umur':21} #proses inisialisasi variabel biodata
print(biodata) #proses pencetakan variabel biodata yang berisi tipe data Dictior
print(type(biodata)) #fungsi untuk mengecek jenis tipe data. akan tampil <class
berarti dict adalah tipe data dictionary
```

5. Casting Tipe Data

Type casting atau pengubahan suatu tipe data ke tipe lainnya biasa dilakukan dalam suatu bahasa pemrograman manakala, data yang diinginkan terbentuk dari tipe data lain. Misal

ketika kamu menginginkan suatu bilangan float dari formdata yang dikirimkan client malah diterima dalam bentuk string. Tentu saja hal ini akan mengakibatkan masalah bila tidak melakukan type casting terlebih dahulu sebelum diproses ke suatu kode selanjutnya.

Berikut Contoh Program Nya:

Integer (int):

```
print("====INTEGER====")
data_int = 9;
print("data = ", data_int, ",type =",type(data_int))
data_float = float(data_int)
data_str = str(data_int)
data_bool = bool(data_int) # akan false jika nilai int = 0
print("data = ", data_float, ",type =",type(data_float))
print("data = ", data_str, ",type =",type(data_str))
print("data = ", data_bool, ",type =",type(data_bool))
```

Float:

```
print("====FLOAT====")
data_float = 0;
print("data = ", data_float, ",type =",type(data_float))
data_int = int(data_float) # akan dibulatkan ke bawah
data_str = str(data_float)
data_bool = bool(data_float) # akan false jika nilai float = 0
print("data = ", data_int, ",type =",type(data_int))
print("data = ", data_str, ",type =",type(data_str))
print("data = ", data_bool, ",type =",type(data_bool))
```

Boolean (bool):

```
print("====BOOLEAN====")
data_bool = False;
print("data = ", data_bool, ",type =",type(data_bool))
data_int = int(data_bool) # akan dibulatkan ke bawah
data_str = str(data_bool)
data_float = float(data_bool) # akan false jika nilai float = 0
print("data = ", data_int, ",type =",type(data_int))
print("data = ", data_str, ",type =",type(data_str))
print("data = ", data_float, ",type =",type(data_float))
```

String (str):

```
print("====STRING====")
data_str = "10";
print("data = ", data_str, ",type =",type(data_str))
data_int = int(data_str) # string harus angka
data_float = float(data_str) # string harus angka
data_bool = bool(data_str) # false jika string kosong
print("data = ", data_int, ",type =",type(data_int))
print("data = ", data_float, ",type =",type(data_float))
print("data = ", data_bool, ",type =",type(data_bool))
```

6. Mengambil Data dari User

Python sudah menyediakan fungsi input() dan raw_input() untuk mengambil inputan dari keyboard.

Cara pakainya:

nama_varabel = input("Sebuah Teks")

Artinya, teks yang kita inputkan dari keyboard akan disimpan ke dalam nama_variabel.

Mari kita coba sebuah contoh :

```
# input user
# data yang dimasukan pasti string
data = input("Masukan data: ")
print("data = ",data,",type =",type(data))
# jika kita ingin mengambil int, maka
angka = float(input("masukan angka: "))
angka = int(input("masukan angka: "))
print("data = ",angka,",type =",type(angka))
bagaimana dengan boolean
biner = bool(int(input("masukan nilai boolean: ")))
print("data = ",biner,",type =",type(biner))
```

Tugas 1
Form data diri

Ini Merupakan source code dari Program Form data diri ynag dimana terdapat

1. Nama
2. NIM
3. Jurusan

4. Tempat, Tanggal Lahir
5. Umur
6. No HP

```
#TUGAS PRAKTIKUM 1 FORM DATA DIRI

print("Form data diri!")
nama=input("Siapakah nama anda? ")
nim=input("Berapa NIM anda? ")
jurusan=input("Apa jurusan yang anda ambil? ")
tempat_lahir=input("Dimanakah anda dilahirkan? ")
tanggal_lahir=input("Tanggal dan bulan anda dilahirkan? ")
tahun_lahir=input("Tahun berapa anda dilahirkan?? ")
telp=input("Masukkan nomor HP anda! ")
now=2023

print("Data diri anda adalah")
print("Nama = ", nama)
print("NIM = ", nim)
print("Jurusan = ", jurusan)
print("Tempat & Tanggal Lahir = ", tempat_lahir , tanggal_lahir, tahun_lah
tahun_int=int(tahun_lahir)
now_int=int(now)
print("Umur = ", now_int - tahun_int)
telp_int=int(telp)
print("Nomor HP = ",telp)
```

Dan ini adalah hasil **Output** dari source code diatas

```
Form data diri!
Siapakah nama anda? Ahmad Hafizuddin
Berapa NIM anda? 2370231046
Apa jurusan yang anda ambil? Teknik Informatika
Dimanakah anda dilahirkan? Bekasi
Tanggal dan bulan anda dilahirkan? 31 Februari
Tahun berapa anda dilahirkan?? 2005
Masukkan nomor HP anda! 0857
Data diri anda adalah
Nama =  Ahmad Hafizuddin
NIM =  2370231046
Jurusan =  Teknik Informatika
Tempat & Tanggal Lahir =  Bekasi 31 Februari 2005
Umur =  18
Nomor HP =  0857
```

MODUL 2

Operator di dalam Python adalah simbol khusus yang berfungsi untuk menjalankan suatu operasi tertentu, baik operasi aritmatika maupun operasi logika. Sedangkan nilai yang dioperasikan oleh operator dinamakan sebagai operan [1].

Berikut ini salah satu contoh paling sederhana dari operator aritmatika pada Python:

```
>>> 10 + 5
15
```

Pada kode program di atas, tanda + adalah sebuah operator. Sedangkan angka 10 dan 5 keduanya merupakan operan.

Dari operasi tersebut, didapatkanlah sebuah hasil akhir berupa nilai integer yaitu 1.

Bahasa pemrograman Python mendukung berbagai macam operator, diantaranya :

- Operator Aritmatika (Arithmetic Operators)
- Operator Perbandingan (Comparison (Relational) Operators)
- Operator Penugasan (Assignment Operators)
- Operator Logika (Logical Operators)
- Operator Bitwise (Bitwise Operators)
- Operator Keanggotaan (Membership Operators)
- Operator Identitas (Identity Operators)

1. Operasi Aritmatika

Operator matematika adalah operator yang kita gunakan untuk menghitung operasi matematika, mulai dari penjumlahan, pengurangan, perkalian, perpangkatan, dan lain sebagainya

Berikut ini tabel operator aritmatika pada python :

Operator	Contoh	Penjelasan
Penjumlahan +	1 + 3 = 4	Menjumlahkan nilai dari masing-masing operan atau bilangan

Pengurangan -	4 – 1 = 3	Mengurangi nilai operan di sebelah kiri menggunakan operan di sebelah kanan
Perkalian *	2 * 4 = 8	Mengalikan operan/bilangan
Pembagian /	10 / 2 = 5	Untuk membagi operan di sebelah kiri menggunakan operan di sebelah kanan
Sisa Bagi %	11 % 2 = 1	Mendapatkan sisa pembagian dari operan di sebelah kiri operator ketika dibagi oleh operan di sebelah kanan
Pangkat **	8 ** 2 = 64	Memangkatkan operan disebelah kiri operator dengan operan di sebelah kanan operator
Pembagian Bulat //	10 // 3 = 3	Sama seperti pembagian. Hanya saja angka dibelakang koma dihilangkan

Dibawah ini adalah contoh penggunaan Operator Aritmatika dalam bahasa pemrograman Python

```
#OPERATOR ARITMATIKA
#Penjumlahan
print(13 + 2)
apel = 7
jeruk = 9
buah = apel + jeruk #
print(buah)
#Pengurangan
hutang = 10000
bayar = 5000
sisaHutang = hutang - bayar
print("Sisa hutang Anda adalah ", sisaHutang)
#Perkalian
panjang = 15
lebar = 8
luas = panjang * lebar
print(luas)
#Pembagian
kue = 16
anak = 4
kuePerAnak = kue / anak
print("Setiap anak akan mendapatkan bagian kue sebanyak ", kuePerAnak)
#Sisa Bagi / Modulus
bilangan1 = 14
bilangan2 = 5
hasil = bilangan1 % bilangan2
print("Sisa bagi dari bilangan ", bilangan1, " dan ", bilangan2, " adalah ", hasil
#Pangkat
bilangan3 = 8
bilangan4 = 2
hasilPangkat = bilangan3 ** bilangan4
print(hasilPangkat)
#Pembagian Bulat
print(10//3)
#10 dibagi 3 adalah 3.3333. Karena dibulatkan maka akan menghasilkan nilai 3
```

2. Latihan Program Perhitungan Sederhana

```
# latihan konversi satuan temperature
# program konversi celcius ke satuan lain
print("\nPROGRAM KONVERSI TEMPERATUR\n")
celcius = float(input('Masukan suhu dalam celcius : '))
print("suhu adalah",celcius, "Celcius")
# reamur
reamur = (4/5) * celcius
print("suhu dalam reamur adalah ",reamur, "Reamur")
# fahrenheit
fahrenheit = ((9/5) * celcius) + 32
print("suhu dalam fahrenheit adalah ",fahrenheit, "Fahrenheit")
# Kelvin
kelvin = celcius + 273
print("suhu dalam kelvin adalah ",kelvin, "Kelvin")
```

3. Operasi Komparasi

Operator perbandingan adalah operator yang bertugas untuk membandingkan antar dua operan. Apakah operan 1 lebih besar dari pada operan 2, atau apakah keduanya sama? Dan lain sebagainya.

Operator perbandingan (comparison operators) digunakan untuk membandingkan suatu nilai dari masing-masing operan.

Operator	Contoh	Penjelasan
Sama dengan ==	1 == 1	bernilai True Jika masing-masing operan memiliki nilai yang sama, maka kondisi bernilai benar atau True.
Tidak sama dengan !=	2 != 2	bernilai False Akan menghasilkan nilai kebalikan dari kondisi sebenarnya.
Tidak sama dengan <>	2 <> 2	bernilai False Akan menghasilkan nilai kebalikan dari kondisi sebenarnya.
Lebih besar dari >	5 > 3	bernilai True Jika nilai operan kiri lebih besar dari nilai operan kanan, maka kondisi menjadi benar.
Lebih kecil dari <	3 < 5	bernilai True Jika nilai operan kiri lebih kecil dari nilai operan kanan, maka kondisi menjadi benar.
Lebih besar atau sama dengan >=	5 >= 3	bernilai True Jika nilai operan kiri lebih besar dari nilai operan kanan, atau sama, maka kondisi menjadi benar.
Lebih kecil atau sama dengan <=	5 <= 3	bernilai True Jika nilai operan kiri lebih kecil dari nilai operan kanan, atau sama, maka kondisi menjadi benar.

Berikut contoh Source Code nya:

1. **Sama dengan ==**

```
a = 4
b = 2
print('======== sama dengan(==)')
hasil = a == 4
print(a,'==',4,'=',hasil)
hasil = b == 4
print(b,'==',4,'=',hasil)
```

2. **Tidak sama dengan !=**

```
a = 4
b = 2
print('======== sama dengan(!=)')
hasil = a != 4
print(a,'!=',4,'=',hasil)
hasil = b != 4
print(b,'!=',4,'=',hasil)
```

3. **Lebih besar dari >**

```
a = 4
b = 2
print('========== lebih besar dari (>)')
hasil = a > 3
print(a,'>',3,'=',hasil)
hasil = b > 3
print(b,'>',3,'=',hasil)
hasil = b > 2
print(b,'>',2,'=',hasil)
```

4. **Lebih kecil dari <**

```
a = 4
b = 2
print('========== kurang dari (<)')
hasil = a < 3
print(a,'<',3,'=',hasil)
hasil = b < 3
print(b,'<',3,'=',hasil)
hasil = b < 2
print(b,'<',2,'=',hasil)
```

5. **Lebih besar atau sama dengan >=**

```
a = 4
b = 2
print('========== lebih dari sama dengan(>=)')
hasil = a >= 3
print(a,'>=',3,'=',hasil)
hasil = b >= 3
print(b,'>=',3,'=',hasil)
hasil = b >= 2
print(b,'>=',2,'=',hasil)
```

6. **Lebih kecil atau sama dengan <=**

```
a = 4
b = 2
print('======== kurang dari sama dengan(<=)')
hasil = a <= 3
print(a,'<=',3,'=',hasil)
hasil = b <= 3
print(b,'<=',3,'=',hasil)
hasil = b <= 2
print(b,'<=',2,'=',hasil)
```

7. **Is dan Is Not**

```
# 'is' sebagai komparasi object identity
print('======== object identity(is)')
x = 5 # ini adalah assignment membuat object
y = 5
print('nilai x =',x,',id = ',hex(id(x)))
print('nilai y =',y,',id = ',hex(id(y)))
hasil = x is y
print('x is y =',hasil)
x = 5 # ini adalah assignment membuat object
y = 6
print('nilai x =',x,',id = ',hex(id(x)))
print('nilai y =',y,',id = ',hex(id(y)))
hasil = x is y
print('x is y =',hasil)
print('======== object identity(is not)')
x = 5 # ini adalah assignment membuat object
y = 5
print('nilai x =',x,',id = ',hex(id(x)))
print('nilai y =',y,',id = ',hex(id(y)))
hasil = x is not y
print('x is y =',hasil)
x = 5 # ini adalah assignment membuat object
y = 6
print('nilai x =',x,',id = ',hex(id(x)))
print('nilai y =',y,',id = ',hex(id(y)))
hasil = x is not y
print('x is y =',hasil
```

4. Operasi Logika_Boolean

Operator logika adalah operator yang sangat penting. Operator ini sangat berkaitan erat dengan operator perbandingan. Dan kedua-duanya juga mengembalikan nilai dengan tipe data yang sama yaitu boolean.

Berikut ini tabel dari operator logika pada python.

Simbol	Tugas	Contoh
and	Mengembalikan True jika dua statement sama-sama benar	True and True
or	Mengembalikan True jika salah satu statement bernilai benar	2 > 5 or 1 < 3
not	Menegasikan hasil. True menjadi False dan sebaliknya	not (1 > 5)

Berikut contoh Source Code nya :

1. OR (jika salah satu true, maka hasilnya adalah true)

```
print('====OR====')
a = False
b = False
c = a or b
print(a,'OR',b,'=',c)
a = False
b = True
c = a or b
print(a,'OR',b,' =',c)
a = True
b = False
c = a or b
print(a,' OR',b,'=',c)
a = True
b = True
c = a or b
print(a,' OR',b,' =',c)
```

2. **AND (jika dua buah nilai true, maka hasil true)**

```
print('====AND====')
a = False
b = False
c = a and b
print(a,'AND',b,'=',c)
a = False
b = True
c = a and b
print(a,'AND',b,' =',c)
a = True
b = False
c = a and b
print(a,' AND',b,'=',c)
a = True
b = True
c = a and b
print(a,' AND',b,' =',c)
```

3. **XOR (akan true jika salah satu true, sisanya false)**

```
print('====XOR====')
a = False
b = False
c = a ^ b
print(a,'XOR',b,'=',c)
a = False
b = True
c = a ^ b
print(a,'XOR',b,' =',c)
a = True
b = False
c = a ^ b
print(a,' XOR',b,'=',c)
a = True
b = True
c = a ^ b
print(a,' XOR',b,' =',c)
```

5. **Latihan Komparasi dan Logika**

```
# episode latihan logika dan komparasi
# membuat gabungan area rentang dari angka
# ++++++3--------10+++++++
inputUser = float(input("masukan angka yang bernilai\nkurang dari 3
\natau \nlebih besar dari 10\n:"))
# ++++++3
# Memeriksa angka kurang dari 3
isKurangDari = (inputUser < 3)
print("Kurang dari 3 =", isKurangDari)
# ---------------10++++++++
# Memeriksa angka lebih dari 10
isLebihDari = (inputUser > 10)
print("Lebih dari 10 =", isLebihDari)
# ++++++3--------10+++++++
isCorrect = isKurangDari or isLebihDari
print("angka yang anda masukan: ", isCorrect)
# -----3++++++++10--------
# kasus irisan
print("\n",10*"=","\n")
inputUser = float(input("masukan angka yang bernilai\nlebih dari 3
\ndan \nkurang dari 10\n:"))
# -----3++++++++++++++++++
# lebih dari 3
isLebihDari = inputUser > 3
print("Lebih dari 3 = ",isLebihDari)
# +++++++++++++++10-------
# kurang dari 10
isKurangDari = inputUser < 10
print("Kurang dari 10 = ",isKurangDari)
# -----3++++++++10--------
isCorrect = isKurangDari and isLebihDari
print("angka yang anda masukan: ", isCorrect)
```

6. **Operator Bitwise**

Operator bitwise adalah operator yang berhubungan dengan angka-angka biner.

Angka-angka biner adalah angka 0 dan 1. Dan pada hakikatnya hanya ini lah angka yang dipahami oleh mesin.

Sebelum kita mulai, kita bisa mengetahui nilai biner dari suatu angka desimal dengan melakukan perintah format() dengan parameter kedua berupa string '08b'. Berikut ini demonstrasi menggunakan python mode interaktif.

```
>>> # biner dari angka 0
>>> print(format(0, '08b'))
00000000
>>> # biner dari angka 1
>>> print(format(1, '08b'))
 00000001
 >>> # biner dari angka 2
>>> print(format(2, '08b'))
 00000010
>>> # biner dari angka 37
>>> print(format(37, '08b'))
 00100101
>>>
```

Setelah sedikit perkenalan dengan binary, berikut ini adalah tabel yang menjelaskan tentang operator bitwise pada python.

Simbol	Nama	Tugas
&	Bitwise AND	Mengembalikan bit 1 jika **dua bit** bernilai 1
l	Bitwise OR	Mengembalikan bit 1 jika **salah satu bit** bernilai 1
^	Bitwise XOR	Mengembalikan bit 1 jika **hanya satu bit** saja yang bernilai 1
-	Bitwise NOT	Membalikkan semua bit
>>	Bitwise right shift	Menggeser bit ke kanan dengan mendorong salinan digit sebelah kiri dan membiarkan digit sebalah kanan terlepas
<<	Bitwise left shift	Menggeser bit ke kiri dengan mendorong digit 0 dan membiarkan bit paling kiri terlepas

Mari kita coba satu persatu dari kode operator bitwise di atas.

```
a = 1
b = 64
print('a =', a, '=', format(a, '08b'))
print('b =', b, '=', format(b, '08b'), '\n')
print('[and]')
print('a & b =', a & b)
print(format(a, '08b'), '&', format(b, '08b'), '=', format(a & b, '08b'),
print('[or]')
print('a | b =', a | b)
print(format(a, '08b'), '|', format(b, '08b'), '=', format(a | b, '08b'),
print('[xor]')
print('a ^ b =', a ^ b)
print(format(a, '08b'), '^', format(b, '08b'), '=', format(a ^ b, '08b'),
print('[not]')
print('~a ~b =', ~a, ~b)
print('~' + format(a, '08b'), '~' + format(b, '08b'), '=', format(~a, '08b
format(~b, '08b'), '\n')
print('[shift right]')
print('a >> b =', a >> b)
print(format(a, '08b'), '>>', format(b, '08b'), '=', format(a >> b, '08b')
print('[shift left]')
print('b << a =', b << a)
print(format(b, '08b'), '<<', format(a, '08b'), '=', format(b << a, '08b')
```

7. Operator Assignment

Operator penugasan adalah operator yang digunakan untuk memberikan sebuah tugas terhadap suatu variabel. Atau dalam bahasa yang lebih manusiawi: operator penugasan adalah operator yang berfungsi untuk memberikan nilai ke dalam sebuah variabel.

Sebenarnya operator penugasan ini hanya ada 1 saja, yaitu operator =.

Akan tetapi, ada banyak variant shortcut yang memudahkan kita untuk melakukan operasi aritmatika atau operasi bitwise bersamaan dengan operasi penugasan.

Berikut ini adalah tabel operator penugasan pada Python.

```
# operasi yang dapat dilakukan dengan penyingkatan
# operasi ditambah dengan assignment
a = 5 # adalah assignment
print("nilai a =",a)
a += 1 # artinya adalah a = a + 1
print("nilai a += 1, nilai a menjadi",a)
a -= 2 # artinya adalah a = a - 2
print("nilai a -= 2, nilai a menjadi",a)
a *= 5 # artinya adalah a = a * 5
print("nilai a *= 5, nilai a menjadi",a)
a /= 2 # artinya adalah a = a / 2
print("nilai a /= 2, nilai a menjadi",a)
b = 10
print("\nnilai b =",b)
# modulus dan floor division
b %= 3
print("nilai b %= 3, nilai b menjadi",b)
b = 10
print("\nnilai b =",b)
b //= 3
print("nilai b //= 3, nilai b menjadi",b)
# pangkat atau eksponen
a = 5
print("\nnilai a =",a)
a **= 3
print("nilai a **= 3, nilai a menjadi",a)
# operasi bitwise
# OR
c = True
print("\nnilai c =",c)
c |= False
print("nilai c |= False, nilai c menjadi",c)
c = False
print("nilai c =",c)
c |= False
print("nilai c |= False, nilai c menjadi",c)
```

```
# AND
c = True
print("\nnilai c =",c)
c &= False
print("nilai c &= False, nilai c menjadi",c)
c = True
print("nilai c =",c)
c &= True
print("nilai c &= True, nilai c menjadi",c)
# XOR
c = True
print("\nnilai c =",c)
c ^= False
print("nilai c ^= False, nilai c menjadi",c)
c = True
print("nilai c =",c)
c ^= True
print("nilai c ^= True, nilai c menjadi",c)
# geser geser
d = 0b0100
print("\nnilai d =",format(d,'04b'))
d >>= 2
print("nilai d >>= 2, nilai d menjadi",format(d,'04b'))
d <<= 1
print("nilai d <<= 1, nilai d menjadi",format(d,'04b'))
```

Tugas 2

Membuat *Program Menghitung Luas dan Keliling* :

1. Luas Lingkaran (3.14 * (r**2)
2. Keliling Persegi Lima (5 * k)
3. Keliling Lingkaran (3.14 * r)
4. Luas Oval (jari-jari mayor * jari- jari minor * 3.14)
5. Luas Belah Ketupat (d1 * d2) / 2

Berikut adalah Source Code nya:

```
#Program Menghitung Luas
print ('Selamat Datang di Program ini')
print ('--------------------')
print
#Mencetak Menu
def menu ():
    print ('Menu Pilihan')
    print ('1. Luas Lingkaran')
    print ('2. Keliling Persegi Lima')
    print ('3. Keliling Lingkaran')
    print ('4. Luas Oval')
    print ('5. Luas Belah Ketupat')

def luaslingkaran ():
    print ('Menghitung Luas Lingkaran')
    r=int(input('Masukkan jari-jari:'))
    luas=3.14*(r**2)
    rounded=round(luas)
    print ('Luas Lingkaran adalah : ', rounded)
    print ('Mau coba lagi [y/n]')
    back=input().upper()
    if back == 'Y':
        menu()
    else:
        exit()

def kelilingpersegilima ():
    print ('Menghitung Keliling Persegi Lima')
    k=int(input('Masukkan sisi Persegi Lima:'))
    keliling=5*k
    print ('Keliling Persegi Lima adalah: ', keliling)
    print ('Mau coba lagi [y/n]')
    back=input().upper()
    if back == 'Y':
        menu()
    else:
        exit()
```

```
def kelilinglingkaran():
    print ('Menghitung Keliling Lingkaran')
    r=int(input('Masukkan diameter lingkaran:'))
    keliling=3.14 * r
    rounded=round(keliling)
    print ('Keliling Lingkaran adalah: ', rounded)
    print ('Mau coba lagi [y/n]')
    back=input().upper()
    if back == 'Y':
        menu()
    else:
        exit()

def luasoval():
    print ('Mehitung Luas Oval')
    jarimayor=int(input('Masukkan jari-jari mayor:'))
    jariminor=int(input('Masukkan jari-jari minor:'))
    luas=jarimayor * jariminor * 3.14
    rounded=round(luas)
    print ("Luas oval adalah : ", rounded)
    print ('Mau coba lagi [y/n]')
    back=input().upper()
    if back == 'Y':
        menu()
    else:
        exit()

def luasbelahketupat():
    print ('Menghitung Luas Belah Ketupat')
    d1=float(input('Masukkan Diagonal 1:'))
    d2=float(input('Masukkan Diagonal 2:'))
    luas=(d1*d2)/2
    print ('Luas Belah Ketupat adalah : ', luas)
    print ('Mau coba lagi [y/n]')
    back=input().upper()
    if back == 'Y':
        menu()
    else:
        exit()
```

```
menu ()

while 1:
    pilih=int(input('Masukkan Pilihan: '))
    if pilih==1:
        luaslingkaran()
    elif pilih==2:
        kelilingpersegilima()
    elif pilih==3:
        kelilinglingkaran()
    elif pilih==4:
        luasoval()
    elif pilih==5:
        luasbelahketupat()
        break
    else:
        print ('Maaf pilihan yang anda masukkan tidak ada')
        print ('Mau coba lagi [y/n]')
        coba=input().upper()
        if coba== 'Y':
            menu()
        else:
            exit()
```

Dan ini adalah hasil Output nya:

```
Selamat Datang di Program ini
-------------------
Menu Pilihan
1. Luas Lingkaran
2. Keliling Persegi Lima
3. Keliling Lingkaran
4. Luas Oval
5. Luas Belah Ketupat
Masukkan Pilihan: 2
Menghitung Keliling Persegi Lima
Masukkan sisi Persegi Lima:20
Keliling Persegi Lima adalah:  100
Mau coba lagi [y/n]
y
Menu Pilihan
1. Luas Lingkaran
2. Keliling Persegi Lima
3. Keliling Lingkaran
4. Luas Oval
5. Luas Belah Ketupat
Masukkan Pilihan: 5
Menghitung Luas Belah Ketupat
Masukkan Diagonal 1:20
Masukkan Diagonal 2:20
Luas Belah Ketupat adalah :  200.0
Mau coba lagi [y/n]
```

MODUL 3

1. Pengenalan String

String adalah jenis yang paling populer di bahasa pemrograman. Kita bisa membuatnya hanya dengan melampirkan karakter dalam tanda kutip. Python memperlakukan tanda kutip tunggal sama dengan tanda kutip ganda. Membuat string semudah memberi nilai pada sebuah variabel.

Dibawah ini adalah contoh sederhana dari sebuah string pada bahasa pemrograman Python.

```
print("Hello World")
```

Dibawah ini adalah tabel dari daftar karakter escape atau karakter non-printable yang dapat diwakili/ditulis dengan awalan notasi backslash.

Notasi Backslash	Karakter Hexadecimal	Penjelasan
\a	0x07	Bell atau alert
\b	0x08	Backspace
\cx		Control-x
\C-x		Control-x
\e	0x1b	Escape
\f	0x0c	Formfeed
\M-\C-x		Meta-Control-x
\n	0x0a	Newline
\nnn		Octal notation, dimana n berada di range 0.7
\r	0x0d	Carriage return
\s	0x20	Space
\t	0x09	Tab
\v	0x0b	Vertical tab
\x		Character x
\xnn		Notasi Hexadecimal, dimana n berada di range 0.9, a.f, atau A.F

Berikut adalah contoh program nya:

1. **Menggunakan Single qoute (‘) atau Double qoute (“)**

```
data = "ini adalah string"
print(data)
print(type(data)
data = "ini adalah string"
print(data)
print(type(data))
# 1. cara membuat string
'''
1. dengan menggunakan single quote '...'
2. dengan menggunakan double quote "..."
'''
data = 'Menggunakan single quote'
print(data)
data = "Menggunakan double quote"
print(data)
print('"Halo, apa kabar?"')
print("'Halo, apa kabar?'")
print("ini adalah hari jum'at")
```

2. **Menggunakan tanda (\ )**

```
# membuat tanda ' menjadi string
print('mari shalat jum\'at')
print('g\'day, isn\'t it?')
# backlash
print("C:\\user\\Ucup")
# tab
print("ucup\t\t\totong, semakin jauhan")
# backspace
print("ucup \botong, jadi deketan")
# newline
print("baris pertama.\nbaris kedua.") # LF -> line feed -> unix, mac
linux
print("baris pertama.\rbaris kedua.") # CR -> carriage return ->
commodore, acorn, lisp
print("baris pertama.\r\nbaris kedua.") # CRLF -> line feed carriage
return -> dipakai oleh windows
```

3. String literal atau raw

```
# hati-hati
print('C:\new folder') # akan salah pathnya
# menggunakan raw string
print(r'C:\new folder')
# multiline literal string
print("""
Nama : Ucup
Kelas : 3 SD
""")
# multiline literal string dan RAW
print(r"""
Nama : Ucup
Kelas : 3 SD\new normal
Website : www.ucup.com/newID
""")
```

2. *Operasi dan Manipulasi String* (Part 1)

Asumsikan variabel string adalah 'Belajar' dan variabel b adalah 'Python', lalu dibawah ini adalah operator yang bisa dipakai pada kedua string di variabel tersebut.

a = "Belajar" b = "Python"

Berikut adalah daftar operator spesial string pada Python :

Operator	Contoh	Penjelasan
+	a + b	akan menghasilkan BelajarPython Concatenation - Menambahkan nilai pada kedua sisi operator
*	a*2	akan menghasilkan BelajarBelajar Pengulangan - Membuat string baru, menggabungkan beberapa salinan dari string yang sama
[]	a[1]	akan menghasilkan e Slice - Memberikan karakter dari indeks yang diberikan
[:]	a[1:4]	akan menghasilkan ela Range Slice - Memberikan karakter dari kisaran yang diberikan
In	B in a	akan menghasilkan 1 Keanggotaan - Mengembalikan nilai true jika ada karakter dalam string yang diberikan
not in	Z not in a	akan menghasilkan 1 Keanggotaan - Mengembalikan nilai true jika karakter tidak ada dalam string yang diberikan
r//R	print r'\n' prints \n dan print R'\n'prints \n Raw String -	Menekan arti aktual karakter Escape. Sintaks untuk string mentah sama persis dengan string biasa kecuali operator string mentah, huruf "r", yang mendahului tanda petik. "R" bisa berupa huruf kecil (r) atau huruf besar (R) dan harus ditempatkan tepat sebelum tanda kutip pertama.
%		Format - Melakukan format String

Berikut adalah codinganya :

1. **Menyambung String (concatenate)**

```
nama_pertama = "Ucup"
nama_tengah = "D"
nama_akhir = "Fame"
nama_lengkap = nama_pertama + nama_tengah + nama_akhir
print(nama_lengkap)
nama_lengkap = nama_pertama + " " + nama_tengah + "'" + nama_akhi
print(nama_lengkap)
```

2. **Menghitung Panjang String**

```
panjang = len(nama_lengkap)
print("panjang " + nama_lengkap + " adalah " + str(panjang))
```

3. **Operator untuk String**

```
# cek apakah ada komponen pada sebuah string
d = "d"
status = d in nama_lengkap
print("apakah " + d + " ada di " + nama_lengkap + ", " + str(statu
D = "D"
status = D in nama_lengkap
print("apakah " + D + " ada di " + nama_lengkap + ", " + str(statu
x = "x"
status = x not in nama_lengkap
print("apakah " + x + " tidak ada di " + nama_lengkap + ", " + st
# mengulang string
print("wk"*100)
print(100*"wk")
# indexing
print("index ke-0 : " + nama_lengkap[0]) # dimulai dari 0
print("index ke-6 : " + nama_lengkap[6]) # index bebas
print("index ke-(-1) : " + nama_lengkap[-1]) # indexing dari dibe
print("index ke-[6,8) : " + nama_lengkap[6:8]) # dimulai dari ind
sampai sebelum 8
print("index ke-[0,2,4,6,8] : " + nama_lengkap[0:10:2]) # diambil
0,2,4,6,8
# item paling kecil
print("nilai terkecil : " + min(nama_lengkap))
# item paling besar
print("nilai terbesar : " + max(nama_lengkap))
ascii_code = ord(" ")
print("ASCII number dari spasi : " + str(ascii_code))
data = 117
print("Character dari ascii code 117 : " + chr(data))
```

4. **Operator dalam bentuk Method**

```
data = "otong surotong pararotong"
jumlah = data.count("o")
print("jumlah o di " + data + " : " + str(jumlah))
```

3. Format String

Fungsi format() berfungsi untuk melakukan pengaturan format string yang akan dicetak atau ditampilkan ke monitor. Sintaks dari fungsi format() adalah sebagai berikut:

format(value[, format_spec])

Fungsi format() memiliki dua parameter yaitu:

- **value** – objek yang akan diformat
- **format_spec** – Spesifikasi atau bagaimana objek tadi akan diformat

format_spec bersifat opsional, dan bisa terdiri dari komponen-komponen berikut:

Berikut adalah codingan nya:

```
# format string
# contoh generic
# string
nama = "ucup"
format_str = f"hello {nama}"
print(format_str)
# boolean
boolean = False
format_str = f"boolean = {boolean}"
print(format_str)
# angka
angka = 2005.5
format_str = f"angka = {angka}"
print(format_str)
# bilangan bulat
angka = 15
format_str = f"bilangan bulat = {angka:d}"
print(format_str)
# bilangan dengan ordo ribuan
angka = 2000000
format_str = f"jutaan = {angka:,}"
print(format_str)
# bilangan desimal
angka = 2005.54321
format_str = f"desimal = {angka:.3f}"
print(format_str)
# menampilkan leading zero
angka = 2005.54321
format_str = f"desimal = {angka:010.3f}"
print(format_str)
# menampilkan tanda + atau -
angka_minus = -10
angka_plus = +10.1234
format_minus = f"minus = {angka_minus:+d}"
format_plus = f"plus = {angka_plus:+.2f}"
print(format_minus)
print(format_plus)
# memformat persen
persentase = 0.045
format_persen = f"persen = {persentase:.2%}"
print(format_persen)
# melakukan operasi aritmatika di dalam placeholder
harga = 10000
jumlah = 5
format_string = f"harga total = Rp. {harga*jumlah:,}"
print(format_string)
# format angka lain (binary, octal, hexadecimal)
angka = 255
format_binary = f"binary = {bin(angka)}"
format_octal = f"octal = {oct(angka)}"
format_hex = f"hex = {hex(angka)}"
print(format_binary)
print(format_octal)
print(format_hex)
```

4. **Format String Width dan Alignment**

```
# Width and Multiline
# Data
data_nama = "Ucup Surucup"
data_umur = 17
data_tinggi = 150.1
data_nomor_sepatu = 44
# string standard
data_string = f"nama = {data_nama}, umur = {data_umur}, tinggi =
{data_tinggi}, sepatu = {data_nomor_sepatu}"
print(5*"="+"Data String"+5*"=")
print(data_string)
# String multiline (dengan menggunakan enter, newline, \n)
data_string = f"nama = {data_nama}, \numur = {data_umur}, \ntinggi =
{data_tinggi}, \nsepatu = {data_nomor_sepatu}"
print("\n"+5*"="+"Data String"+5*"=")
print(data_string)
# String multiline (kutip triplets)
data_string = f"""nama = {data_nama}
umur = {data_umur}
tinggi = {data_tinggi}
sepatu = {data_nomor_sepatu}
"""
print("\n"+5*"="+"Data String"+5*"=")
print(data_string)
# mengatur lebar
data_nama = "Ucup Surucup"
data_tinggi = 105.17
data_string = f"""
nama = {data_nama:>5}
umur = {data_umur:>5}
tinggi = {data_tinggi:>5}
sepatu = {data_nomor_sepatu:>5}
"""
print("\n"+5*"="+"Data String"+5*"=")
print(data_string)
```

MODUL 4

Program Python dapat menangani tanggal dan waktu dengan beberapa cara. Konversi antara format tanggal adalah tugas umum untuk komputer. Modul waktu dan kalender Python melacak tanggal dan waktu.

Interval waktu adalah bilangan floating-point dalam satuan detik. Instansi tertentu dalam waktu dinyatakan dalam hitungan detik sejak pukul 12:00 1 Januari 1970. Dibawah ini adalah contoh penggunanaya.

```
import time; # Digunakan untuk meng-import modul time
ticks = time.time()
 print "Berjalan sejak 12:00am, January 1, 1970:", ticks
```

1. **Date and Time (Latihan)**

Banyak fungsi waktu Python menangani waktu sebagai tuple dari 9 nomor, seperti yang terdapat pada tabel di bawah ini.

Index	Field	Value
0	4- digit year	2008
1	Bulan	1 sampai 12
2	Hari	1 sampai 31
3	Jam	0 sampai 23
4	Menit	0 sampai 59
5	Detik	0 sampai 61
6	Hari dalam Minggu	0 sampai 6 (0 adalah senin)
7	Hari dalam bulan	1 sampai 366
1	Daylight saving	-1, 0, 1, -1 means library determines DST

Tuple di atas setara dengan struktur struct_time. Struktur ini memiliki atribut berikut

Index	Atribut	Value
0	tm_year	
1	tm_mon	
2	tm_mday	
3	tm_hour	
4	tm_min	
5	tm_sec	
6	tm_wday	
7	tm_yday	
8	tm_isdst	

Berikut adalah contoh program nya:

```
# Date and time (latihan)
import datetime as dt
print("Silahkan masukan tanggal, \nbulan dan tahun lahir anda \n")
tanggal = int(input("Tanggal \t: "))
bulan = int(input("Bulan \t\t: "))
tahun = int(input("Tahun \t\t: "))
tanggal_lahir = dt.date(tahun,bulan,tanggal)
print(f"Tanggal lahir anda adalah : {tanggal_lahir}")
hari_ini = dt.date.today()
print(f"Hari ini tanggal: {hari_ini}")
umur_hari = hari_ini - tanggal_lahir
umur_tahun = umur_hari.days // 365
umur_bulan_sisa = (umur_hari.days % 365) // 30
print(f"Hari nya adalah : {tanggal_lahir:%A}")
print(f"Umur anda adalah: {umur_tahun} tahun, {umur_bulan_sisa}
bulan")
```

2. If dan Else Statement

Pengambilan keputusan (kondisi if) digunakan untuk mengantisipasi kondisi yang terjadi saat jalanya program dan menentukan tindakan apa yang akan diambil sesuai dengan kondisi.

Pada python ada beberapa statement/kondisi diantaranya adalah if , else dan elif dan Kondisi if digunakan untuk mengeksekusi kode jika

Kondisi bernilai benar True.

Jika kondisi bernilai salah False eksekusi. maka statement/kondisi if tidak akan di eksekusi.

Pengambilan keputusan (kondisi if else) tidak hanya digunakan untuk menentukan tindakan apa yang akan diambil sesuai dengan kondisi, tetapi juga digunakan untuk menentukan tindakan apa yang akan diambil/dijalankan jika kondisi tidak sesuai.

Pada python ada beberapa statement/kondisi diantaranya adalah if, else dan elif Kondisi if digunakan untuk mengeksekusi kode jika kondisi bernilai benar.

Kondisi if else adalah kondisi dimana jika pernyataan benar maka kode dalam if akan dieksekusi, tetapi jika bernilai salah kode di dalam else. maka akan mengeksekusi Dibawah ini adalah contoh penggunaan kondisi if pada Python.

Berikut adalah contoh penggunaan if pada Python:

```
# If dan Else Statement
# 1. if nya
# 2. kondisinya
# 3. aksinya
nama = input("Siapa nama anda? ")
# program sebelumnya
# if kondisi: aksi
# program selanjutnya
# 1. program if inline
# if nama=="ucup" : print("Kamu Ganteng abieeez!!!!")
# print(f"Terima kasih {nama}")
# 2. program if indentation
# if nama=="ucup":
# print("kamu ganteng abieeeez!") #
print("kamu juga keren banget!") #
print(f"Terima kasih {nama}")
# 3. Else Statement
if nama=="otong":
print("hai otooong, si keren!!!")
else:
print("Ah kamu bukan otong, kamu gak keren! :(")
print("akhir dari program")
```

3. Elif Statement

Pengambilan keputusan (kondisi if elif) merupakan lanjutan/percabangan logika dari “kondisi if”. Dengan elif kita bisa membuat kode program yang akan menyeleksi beberapa kemungkinan yang bisa terjadi. Hampir sama dengan kondisi “else”, bedanya kondisi “elif” bisa banyak dan tidak hanya satu.

Berikut contoh penggunaan kondisi elif pada Python:

```
# ELIF = else if statement
nama = input("Nama kamyu siapa? ")
if nama == "ucup": # kondisi 1
print("Hai ganteeeeng beuds!") # aksi true 1
elif nama == "otong": # kondisi 2
print("Hai si kece bangeeeets!!") # aksi true 2
elif nama == "mario": # kondisi 3
print("Hai humooooreeeesh!") # aksi true 3
else:
print("au ah gak kenal!!!") # aksi false
print("ini adalah akhir dari program")
```

4. Latihan Percabangan

```
# Latihan
# kalkulator sederhana
print(20*"=")
print("Kalkulator Sederhana")
print(20*"=" + "\n")
angka_1 = float(input("masukan angka 1 = "))
operator = input("operator (+,-,x,/) : ")
angka_2 = float(input("masukan angka 2 = "))
# percabangannya
if operator == "+":
hasil = angka_1 + angka_2
print(f"hasilnya adalah {hasil}")
elif operator == "-":
hasil = angka_1 - angka_2
print(f"hasilnya adalah {hasil}")
elif operator == "x" or operator == "*":
hasil = angka_1 * angka_2
print(f"hasilnya adalah {hasil}")
elif operator == "/":
hasil = angka_1 / angka_2
print(f"hasilnya adalah {hasil}")
else:
print("masukan yang bener dong!, aku pusying")
print("Akhir dari program, terima gajih!")
```

5. Loop (Perulangan)

Secara umum, pernyataan pada bahasa pemrograman akan dieksekusi secara berurutan. Pernyataan pertama dalam sebuah fungsi dijalankan pertama, diikuti oleh yang kedua, dan seterusnya. Tetapi akan ada situasi dimana Anda harus menulis banyak kode, dimana kode tersebut sangat banyak. Jika dilakukan secara manual maka Anda hanya akan membuang-buang tenaga dengan menulis beratus-ratus bahkan beribu-ribu kode. Untuk itu Anda perlu menggunakan pengulangan di dalam bahasa pemrograman Python.

Di dalam bahasa pemrograman Python pengulangan dibagi menjadi 3 bagian, yaitu :

- While Loop
- For Loop
- Nested Loop

For Loop (Perulangan)

Pengulangan for pada Python memiliki kemampuan untuk mengulangi item dari urutan apapun, seperti list atau string .

Berikut contoh penggunaan pengulangan For Loop.

```
# Perulangan (loop)
# for kondisi:
# aksi
# ini dengan list
angka2_list = [0,2,4,8,10] # ini adalah list
print(angka2_list)
for i in angka2_list:
print(f"i sekarang -> {i}")
print("akhir dari program 1 \n")
# ini dengan range
angka2_range = range(5)
for i in angka2_range:
print(f"i sekarang -> {i}")
print("akhir dari program 2 \n")
angka2_range = range(1,10)
for i in angka2_range:
print(f"i sekarang -> {i}")
# print("saya keren")
print("akhir dari program 3 \n")
# menggunakan string
data_str = "saya ganteng abiees"
for huruf in data_str:
print(huruf)
print("akhir dari program 4 \n")
```

While Loop (Perulangan)

Pengulangan While Loop di dalam bahasa pemrograman Python dieksesusi statement berkali-kali selama kondisi bernilai benar atau True . Dibawah ini adalah contoh penggunaan pengulangan While Loop.

Berikut contoh penggunaan pengulangan While Loop.

```
# while loop
# while kondisi:
# aksi ini
# aksi itu
# akhir dari program
angka = 0
print(f"angka sekarang -> {angka}")
while angka < 5:
angka += 1
print(f"angka sekarang -> {angka}")
print("otong ganteng maxsyimaal!")
print("cukuuup")
```

Nested Loop (Perulangan)

Bahasa pemrograman Python memungkinkan pe nggunaan satu lingkaran di dalam loop lain.

Bagian berikut menunjukkan beberapa contoh untuk menggambarkan konsep tersebut.

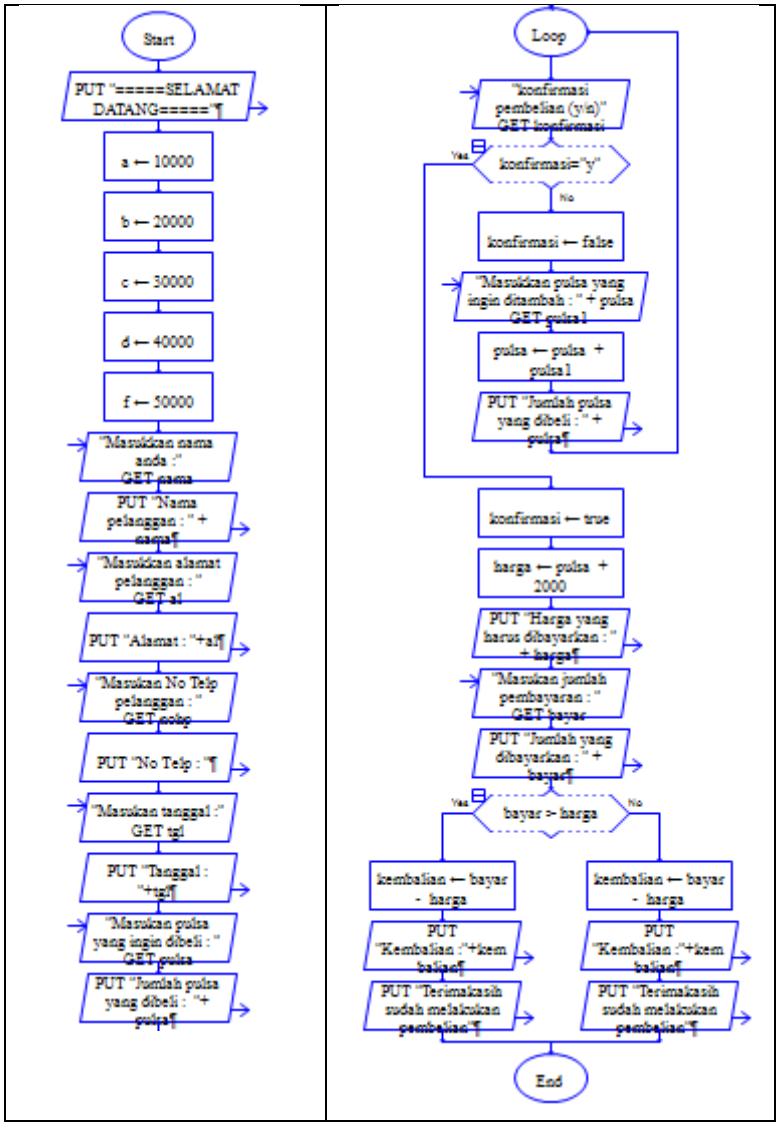
Berikut contoh penggunaan Nested Loop.

```
#Contoh penggunaan Nested Loop
#Catatan: Penggunaan modulo pada kondisional mengasumsikan nilai
selain nol sebagai True(benar) dan nol sebagai False(salah)
i = 2
while(i < 100):
j = 2
while(j <= (i/j)):
if not(i%j): break
j = j + 1
if (j > i/j) : print(i, " is prime")
i = i + 1
print("Good bye!")
```

TUGAS 3

Program Kasir Sederhana **beserta Invoice nya**

Berikut adalah Flowchart dari Program Kasir :



Dan dibawah ini adalah source code nya:

```
import datetime
today = datetime.datetime.now()

pajak=2000

print("----------------- Program Kasir Pulsa Sederhana -----------------")
pembeli = input("Masukan nama Pembeli: ")
alamat = input ("Masukan alamat Pembeli: ")
nohp = input ("Masukan No Telp Pembeli: ")
print ("Nama Pembeli :", pembeli)

def fungsi_pulsa():
   global totalpls
   global pls
   print ("\n----------------- PILIHAN PULSA -----------------")
   print("1. Pulsa Rp 10000")
   print("2. Pulsa Rp 20000")
   print("3. Pulsa Rp 30000")
   print("4. Pulsa Rp 40000")
   print("5. Pulsa Rp 50000")
   nomor=int(input("Masukan Pilihan: "))

   if nomor==1:
       totalpls=pajak+10000
       print("Pulsa 10000 = Rp", totalpls)
       pls=("Pulsa Rp.10000")
   elif nomor==2:
       totalpls=pajak+20000
       print ("Pulsa 20000 = Rp", totalpls)
       pls=("Pulsa Rp.30000")
   elif nomor==3:
       totalpls=pajak+30000
       print("Pulsa 30000 = Rp", totalpls)
       pls=("Pulsa Rp.40000")
   elif nomor==4:
        totalpls=pajak+40000
        print("Pulsa 40000 = Rp", totalpls)
        pls=("Pulsa Rp.40000")
   elif nomor==5:
        totalpls=pajak+50000
        print("Pulsa 50000 = Rp", totalpls)
        pls=("Pulsa Rp.50000")
   else:
      print("====SALAH MAS PILIHANNYA===S")
      fungsi_pulsa()
fungsi_pulsa()
```

```
print("\nTotal harus Dibayar: Rp",totalpls)
uang=int(input("Uang Tunai Pembeli: Rp "))
kembalian=int(uang-totalpls)
print("Kembalian :",kembalian)
print("\n==================================")
print("========== INVOICE =========")
print("==================================")
print ("Nama\t\t:",pembeli)
print ("Alamat\t\t:",alamat)
print ("NO Telp\t\t:",nohp)
print ("Tanggal\t\t:",today)
print ("Beli\t\t:",pls,("Rp", totalpls))
print ("Tagihan\t\t: Rp",totalpls)
print ("Dibayar\t\t: Rp",uang)
print ("Kembalian\t: Rp",kembalian)
print("==================================")
print("==================================")
```

Dan ini adalah hasil Output nya:

```
----------------- Program Kasir Pulsa Sederhana -----------------
Masukan nama Pembeli: Ahmad Hafizuddin
Masukan alamat Pembeli: Jl.Jonggol
Masukan No Telp Pembeli: 0857
Nama Pembeli : Ahmad Hafizuddin

----------------- PILIHAN PULSA -----------------
1. Pulsa Rp 10000
2. Pulsa Rp 20000
3. Pulsa Rp 30000
4. Pulsa Rp 40000
5. Pulsa Rp 50000
Masukan Pilihan: 5
Pulsa 50000 = Rp 52000

Total harus Dibayar: Rp 52000
Uang Tunai Pembeli: Rp 100000
Kembalian : 48000

==================================
========== INVOICE =========
==================================
Nama            : Ahmad Hafizuddin
Alamat          : Jl.Jonggol
NO Telp         : 0857
Tanggal         : 2023-11-29 13:21:03.220738
Beli            : Pulsa Rp.50000 ('Rp', 52000)
Tagihan         : Rp 52000
Dibayar         : Rp 100000
Kembalian       : Rp 48000
==================================
==================================
```