BAB 1

PENGENALAN PYTHON

Python adalah bahasa pemrograman interpretatif multiguna. Tidak seperti bahasa lain yang susah untuk dibaca dan dipahami, python lebih menekankan pada keterbacaan kode agar lebih mudah untuk memahami sintaks. Hal ini membuat Python sangat mudah dipelajari baik untuk pemula maupun untuk yang sudah menguasai bahasa pemrograman lain. Bahasa ini muncul pertama kali pada tahun 1991, dirancang oleh seorang bernama Guido van Rossum. Sampai saat ini Python masih dikembangkan oleh Python Software Foundation. Bahasa Python mendukung hampir semua sistem operasi, bahkan untuk sistem operasi Linux, hampir semua distronya sudah menyertakan Python di dalamnya. Dengan kode yang simpel dan mudah diimplementasikan, seorang programmer dapat lebih mengutamakan pengembangan aplikasi yang dibuat, bukan malah sibuk mencari syntax error.

```
print("Python sangat simpel")
```

Hanya dengan menuliskan kode print seperti yang diatas, anda sudah bisa mencetak apapun yang anda inginkan di dalam tanda kurung (). Dibagian akhir kode pun, anda tidak harus mengakhirnya dengan tanda semicolon ;

BAB 2
INSTALASI PYTHON

Sebelum Anda menggunakan Python, Anda harus menginstalnya terlebih dahulu di sistem operasi komputer Anda. Saat ini Python memiliki 2 versi yang berbeda, yaitu Python versi 3.4.3 dan Python versi 2.7.10. Disini kita akan belajar bahasa pemrograman Python menggunakan versi terbaru 3.4.3 Cara menginstal python sangat mudah, ikuti panduan dibawah ini. Dibawah adalah panduan cara instal python di platform Linux, Windows dan Mac OS.

2.1 Linux

- Buka browser, kunjungi http://www.python.org/downloads/source/
- Download versi terbaru Python berbentuk file zip untuk Unix/Linux
- Ekstrak file zip yang baru saja di download
- Edit file Modules/Setup jika Anda ingin kostumisasi Python
- Jalankan ./configure script
- make
- make install

  Langkah ini akan menginstal Python di lokasi standar /usr/local/bin dan library di /usr/local/lib/pythonXX dimana XX adalah versi terbaru Python yang anda gunakan.

2.2 Windows

- Buka browser, kunjungi http://www.python.org/downloads/windows/
- **ATAU**, klik direct link https://www.python.org/ftp/python/3.4.3/python-3.4.3.msi
- Buka (klik 2x) file installer python yang baru saja di download
- Ikuti langkah instalasi sampai selesai

2.3 Mac OS

- Buka browser, kunjungi http://www.python.org/download/mac/
- Download versi terbaru Python untuk Macintosh
- Buka file yang baru saja di download
- Ikuti langkah instalasi sampai selesai

BAB 3
MENJALANKAN PYTHON

Untuk menjalankan Python ada banyak cara yang bisa dilakukan. Anda bisa menggunakan sheel, terminal atau menggunakan IDE (Integrated Development Environment). Di bawah ini adalah langkah-langkah menjalankan Python dengan cara yang paling mudah.

3.1 Linux

- Buka terminal **(Ctrl + Alt + T)**
- Ketik **python** maka Anda akan masuk ke sheel Python.
- Tuliskan script Python Anda, contoh: **print("Selamat datang di Python")**. jika sudah tekan tombol **Enter**, dan script Python akan dijalankan/eksekusi.
- Untuk keluar dari sheel Python ketik **exit()** atau
- Gunakan teks editor, misalnya gedit.
- Buat file baru, dan ketikan script python Anda, contoh: **print("Selamat datang di Python")**.
- Save As dengan ekstensi **.py** (contoh: cetak.py).
- Jalankan file dengan menggunakan Terminal.
- Buka terminal **(Ctrl + Alt + T)**.
- Masuk ke direktori dimana file Python Anda disimpan (contoh: **cd /Users/admin/ Desktop/**).
- Jalankan script Python dengan menggunakan **python** diikuti dengan nama file (contoh: **python cetak.py**).
- Script Python Anda akan dieksekusi/dijalankan.

3.2 Windows

- Buka Python sheel, Anda bisa mencarinya di tombol Start.
- Tuliskan script Python Anda, contoh: **print("Selamat datang di Python")**. jika sudah tekan tombol **Enter**, dan script Python akan dijalankan/eksekusi.
- Untuk keluar dari sheel Python ketik **exit()**

3.3 Macintosh

- Buka terminal.
- Ketik **python** maka Anda akan masuk ke sheel Python.
- Tuliskan script Python Anda, contoh: **print("Selamat datang di Python")**. jika sudah tekan tombol **Enter**, dan script Python akan dijalankan/eksekusi. • Untuk keluar dari sheel Python ketik **exit()** atau

- Gunakan teks editor.
- Buat file baru, dan ketikan script python Anda, contoh: **print("Selamat datang di Python")**.
- Save As dengan ekstensi **.py** (contoh: cetak.py).
- Jalankan file dengan menggunakan Terminal.
- Buka terminal **(Ctrl + Alt + T)**.
- Masuk ke direktori dimana file Python Anda disimpan (contoh: **cd /Users/admin/ Desktop/**).
- Jalankan script Python dengan menggunakan **python** diikuti dengan nama file (contoh: **python cetak.py**).
- Script Python Anda akan dieksekusi/dijalankan.

BAB IV
INTEGRATED DEVELOPMENT ENVIRONMENT (IDE) PYTHON

IDE adalah sebuah software aplikasi yang memberikan Anda fasilitas bermanfaat ketika membuat program. Biasanya sebuah IDE terdiri dari source code editor build automation tools dan debugger. Untuk menulis sebuah program, bisa menggunakan text editor atau IDE nya. Bagi yang sudah mahir, menulis program dengan text editor bukanlah menjadi masalah. Tetapi untuk pemula, akan lebih mudah menggunakan IDE. IDE untuk Python sangatlah banyak, tersedia bermacam-macam IDE dengan kelebihan dan kekurangan masing-masing.

Beberapa IDE untuk Python yang cukup populer adalah :

- Komodo
- LiClipse
- NetBeans
- PyCharm
- Kdevelop
- PyDev
- Wing IDE
- dan masih banyak lainnya

(http://wiki.python.org/moin/IntegratedDevelopmentEnvironments).

BAB 5
HELLO WORLD PYTHON

Syntax bahasa Python hampir sama dengan bahasa pemrograman pada umumnya seperti Java atau PHP.

5.1 Syntax Dasar

Dibawah ini adalah contoh fungsi Python yang digunakan untuk mencetak. Di Python untuk mencetak cukup gunakan fungsi **print()**, dimana sesuatu yang akan dicetak harus diletakkan diantara kurung buka dan kurung tutup, bahkan di Python versi 2.x Anda tidak harus menggunakan tanda kurung kurawal, cukup pisahkan dengan spasi. Jika ingin mencetak tipe data String langsung, Anda harus memasukanya ke dalam tanda kutip terlebih dahulu.

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

Saat anda menjalankan script diatas, Anda akan melihat output berupa text **Hello World**

5.2 Python Case Sensitive

Python bersifat case sensitif, ini artinya huruf besar dan huruf kecil memiliki perbedaan. Sebagai contoh jika Anda menggunakan fungsi print dengan huruf kecil **print()** akan berhasil. Lain hal jika anda menggunakan huruf kapital **Print()** atau **PRINT()**, akan muncul pesan error. Aturan ini berlaku untuk nama variabel ataupun fungsi-fungsi lainya.

5.3 Komentar Python

Komentar (comment) adalah kode di dalam script Python yang tidak dieksekusi atau tidak dijalankan mesin. Komentar hanya digunakan untuk menandai atau memberikan keterangan tertulis pada script. Komentar biasa digunakan untuk membiarkan orang lain memahami apa yang dilakukan script. atau untuk mengingatkan kepada programmer sendiri jika suatu saat kembali mengedit script tersebut. Untuk menggunakan komentar anda cukup menulis tanda pagar #, diikuti dengan komentar Anda. Dibawah ini adalah contoh penggunaan komentar pada Python.

```
#Ini adalah komentar
# Tulisan ini tidak akan dieksekusi
#komentar dengan tanda pagar hanya bisa digunakan
#untuk
#satu
#baris
print("Hello World") #ini juga komentar
```

```
        #print("Welcome")
        # komentar bisa berisi spesial karakter !@#$%^&*(),./;'[]\
        #mencetak nama
print("Budi")
        #mencetak angka/integer
        print(123)
```

Saat anda menjalankan script diatas, Anda akan melihat output berupa **Hello World**, **Budi** dan **123**, karena tulisan/komentar yang ditulis tidak dieksekusi.

5.4 Tipe Data pada Python

Tipe data adalah suatu media atau memori pada komputer yang digunakan untuk menampung informasi. Python sendiri mempunyai tipe data yang cukup unik bila kita bandingkan dengan bahasa pemrograman yang lain. 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', 768, 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.

```
#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
Dictionary
type(biodata) #fungsi untuk mengecek jenis tipe data. akan tampil <class
'dict'> yang berarti dict adalah tipe data dictionary
```

5.5 Variabel Python

Variabel adalah lokasi memori yang dicadangkan untuk menyimpan nilai-nilai. Ini berarti bahwa ketika Anda membuat sebuah variabel Anda memesan beberapa ruang di memori. Variabel menyimpan data yang dilakukan selama program dieksekusi, yang natinya isi dari variabel tersebut dapat diubah oleh operasi - operasi tertentu pada program yang menggunakan variabel.

Variabel dapat menyimpan berbagai macam tipe data. Di dalam pemrograman Python, variabel mempunyai sifat yang dinamis, artinya variabel Python tidak perlu didekralasikan tipe data tertentu dan variabel Python dapat diubah saat program dijalankan. Penulisan variabel Python sendiri juga memiliki aturan tertentu, yaitu :

1. Karakter pertama harus berupa huruf atau garis bawah/underscore _
2. Karakter selanjutnya dapat berupa huruf, garis bawah/underscore _ atau angka
3. Karakter pada nama variabel bersifat sensitif (case-sensitif). Artinya huruf kecil dan huruf besar dibedakan. Sebagai contoh, variabel **namaDepan** dan **namadepan** adalah variabel yang berbeda.

Untuk mulai membuat variabel di Python caranya sangat mudah, Anda cukup menuliskan variabel lalu mengisinya dengan suatu nilai dengan cara menambahkan tanda sama dengan = diikuti dengan nilai yang ingin dimasukan. Dibawah ini adalah contoh penggunaan variabel dalam bahasa pemrograman Python.

```
#proses memasukan data ke dalam variabel
nama = "John Doe"
#proses mencetak variabel
print(nama)
#nilai dan tipe data dalam variabel  dapat diubah
umur = 20              #nilai awal print(umur)
#mencetak nilai umur type(umur)
#mengecek tipe data umur umur = "dua puluh satu"
#nilai setelah diubah print(umur)          #mencetak
nilai umur type(umur)            #mengecek tipe data
umur
namaDepan = "Budi"
namaBelakang = "Susanto"
nama = namaDepan + " " + namaBelakang
umur = 22
hobi =
"Berenang"
print("Biodata\n", nama, "\n", umur, "\n", hobi)
#contoh variabel lainya inivariabel
= "Halo" ini_juga_variabel =
```

```
"Hai" _inivariabeljuga = "Hi"
inivariabel222 = "Bye"
panjang = 10
lebar = 5
luas = panjang * lebar
print(luas)
```

5.6 Operator

Operator adalah konstruksi yang dapan memanipulasi nilai dari operan. Sebagai contoh operasi 3 + 2 = 5. Disini 3 dan 2 adalah operan dan + adalah operator. 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 Identisas (Identity Operators) Mari kita membahasnya satu-persatu.

Operator Aritmatika

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 / 5 = 2**	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

```
#file /python_dasar/operator_aritmatika.py
#OPERATOR ARITMATIKA
#Penjumlah
an 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)
#Pembagia
n 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
```

Operator Perbandingan

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 <	**5 < 3** 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.

Assignment Operator

Operator penugasan digunakan untuk memberikan atau memodifikasi nilai ke dalam sebuah variabel.

Operator	Contoh	Penjelasan
Sama dengan =	**a = 1**	Memberikan nilai di kanan ke dalam variabel yang berada di sebelah kiri.
Tambah sama dengan +=	**a += 2**	Memberikan nilai variabel dengan nilai variabel itu sendiri ditambah dengan nilai di sebelah kanan.
Kurang sama dengan -=	**a -= 2**	Memberikan nilai variabel dengan nilai variabel itu sendiri dikurangi dengan nilai di sebelah kanan.
Kali sama dengan *=	**a *= 2**	Memberikan nilai variabel dengan nilai variabel itu sendiri dikali dengan nilai di sebelah kanan.
Bagi sama dengan /=	**a /= 4**	Memberikan nilai variabel dengan nilai variabel itu sendiri dibagi dengan nilai di sebelah kanan.
Sisa bagi sama dengan %=	**a %= 3**	Memberikan nilai variabel dengan nilai variabel itu sendiri dibagi dengan nilai di sebelah kanan. Yang diambil nantinya adalah sisa baginya.
Operator	**Contoh**	**Penjelasan**

Pangkat sama dengan **=	**a **= 3**	Memberikan nilai variabel dengan nilai variabel itu sendiri dipangkatkan dengan nilai di sebelah kanan.
Pembagian bulat sama dengan //=	**a //= 3**	Membagi bulat operan sebelah kiri operator dengan operan sebelah kanan operator kemudian hasilnya diisikan ke operan sebelah kiri.

Logical Operator

Operator	Contoh	Penjelasan
and	a, b = True, True # hasil akan True print a and b	Jika kedua operan bernilai True, maka kondisi akan bernilai True. Selain kondisi tadi maka akan bernilai False.
or	a, b = True, False # hasil akan True print a or b print b or a print a or a # hasil akan False print b or b	Jika salah satu atau kedua operan bernilai True maka kondisi akan bernilai True. Jika keduanya False maka kondisi akan bernilai False.
not	a, b = True, False # hasil akan True print not a print not b	Membalikkan nilai kebeneran pada operan misal jika asalnya True akan menjadi False dan begitupun sebaliknya.

Bitwise Operator

Operator	Contoh	Penjelasan
&	a, b = 13, 37 # a akan bernilai '0000 1101' # b akan bernilai '0010 0101' c = a & b # c akan bernilai 5 = '0000 0101' print c	Operator biner AND, memeriksa apakah operan di sebelah kiri dan operan sebelah kanan mempunyai angka biner 1 di setiap bit. Jika keduanya bernilai 1 maka bit hasil operasi akan bernilai 1.

|	a, b = 13, 37 # a akan bernilai '0000 1101' # b akan bernilai '0010 0101' c = a | b	Operator biner OR, memeriksa apakah operan di sebelah kiri dan operan sebelah kanan mempunyai angka biner 1 di setiap bit. Jika salah
Operator	**Contoh**	**Penjelasan**
	# c akan bernilai 45 = '0010 1101' print c	satunya bernilai 1 maka bit hasil operasi akan bernilai 1.
^	a, b = 13, 37 # a akan bernilai '0000 1101' # b akan bernilai '0010 0101' c = a ^ b # c akan bernilai 40 = '0010 1000' print c	Operator biner XOR, memeriksa apakah operan di sebelah kiri dan operan sebelah kanan mempunyai angka biner 1 di setiap bit. Jika keduanya bernilai 1 maka bit hasil operasi akan bernilai 0.
Kali sama dengan *=	**a *= 2**	Operator biner Negative, membalik nilai bit. Misal dari 1 menjadi 0, dari 0 menjadi 1.
~	a, b = 13, 37 # a akan bernilai '0000 1101' # b akan bernilai '0010 0101'	Memberikan nilai variabel dengan nilai variabel itu sendiri dibagi dengan nilai di sebelah kanan.
<<	a, b = 13, 37 # a akan bernilai '0000 1101' # b akan bernilai '0010 0101' # hasil bernilai 52 = "0011 0100' print a << 2 # hasil bernilai 148 = '1001 0100' print b << 2	Memberikan nilai variabel dengan nilai variabel itu sendiri dibagi dengan nilai di sebelah kanan. Yang diambil nantinya adalah sisa baginya.
>>	a, b = 13, 37 # a akan bernilai '0000 1101' # b akan bernilai '0010 0101' # hasil bernilai 3 = '0000 0011' print a >> 2	Operator penggeser biner ke kiri, deret bit akan digeser ke kiri sebanyak n kali.

	# hasil bernilai 9 = '0000 1001' print b >> 2	

Membership Operator

Operator	Contoh	Penjelasan
in	sebuah_list = [1, 2, 3,4 ,5] print 5 in sebuah_list	Memeriksa apakah nilai yang dicari berada pada list atau struktur data python lainnya. Jika nilai tersebut ada maka kondisi akan bernilai True.
not in	sebuah_list = [1, 2, 3,4 ,5] print 10 not in sebuah_list	Memeriksa apakah nilai yang dicari tidak ada pada list atau struktur data python lainnya. Jika nilai tersebut tidak ada maka kondisi akan bernilai True.

Identity Operator

Operator	Contoh	Penjelasan
is	a, b = 10, 10 # hasil akan True print a is b	Memeriksa apakah nilai di sebelah kiri operan memiliki identitas memori yang sama dengan nilai di sebelah kanan operan. Jika sama maka kondisi bernilai True.
is not	a, b = 10, 5 # hasil akan True	Memeriksa apakah nilai di sebelah kiri operan memiliki identitas memori yang berbeda dengan nilai di sebelah kanan operan. Jika berbeda maka kondisi bernilai True.

	print a is not b	

5.7 Konfisi If

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 Kondisi if digunakan untuk mengeksekusi kode jika kondisi bernilai benar. Jika kondisi bernilai salah maka statement/kondisi if tidak akan di-eksekusi. Dibawah ini adalah contoh penggunaan kondisi if pada Python

```
#Kondisi if adalah kondisi yang akan dieksekusi oleh program jika bernilai
benar atau TRUE nilai = 9 #jika kondisi benar/TRUE maka program akan
mengeksekusi perintah dibawahnya if(nilai > 7):     print("Selamat Anda Lulus")
#jika kondisi salah/FALSE maka program tidak akan mengeksekusi perintah
dibawahnya if(nilai > 10):
print("Selamat Anda Lulus")
```

Dari contoh diatas, jika program dijalankan maka akan mencetak string "Selamat Anda Lulus Ujian" sebanyak 1 kali yaitu pada if pertama. Di if kedua statement bernilai salah, jadi perintah print("Selamat Anda Lulus") tidak akan dieksekusi.

5.8 If Else

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 (true) maka kode dalam if akan dieksekusi, tetapi jika bernilai salah (false) maka akan mengeksekusi kode di dalam else. Dibawah ini adalah contoh penggunaan kondisi if else pada Python

```
#Kondisi if else adalah jika kondisi bernilai TRUE maka akan dieksekusi pada
if, tetapi jika bernilai FALSE maka akan dieksekusi kode pada else
nilai = 3 #Jika pernyataan pada if bernilai TRUE maka if akan dieksekusi, tetapi jika
FALSE kode pada else yang akan
dieksekusi. if(nilai > 7):     print("Selamat
Anda Lulus") else:     print("Maaf Anda
Tidak Lulus")
```

Pada contoh diatas, jika program dijalankan maka akan mencetak string "Maaf Anda Tidak Lulus" karena pernyataan pada if bernilai FALSE

5.9 Kondisi Elif

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. Dibawah ini adalah contoh penggunaan kondisi elif pada Python

```
#Contoh penggunaan kondisi elif
hari_ini = "Minggu" if(hari_ini == "Senin"):
print("Saya akan kuliah") elif(hari_ini == "Selasa"):     print("Saya akan kuliah")
elif(hari_ini == "Rabu"):
print("Saya akan kuliah") elif(hari_ini == "Kamis"):     print("Saya akan kuliah")
elif(hari_ini == "Jumat"):     print("Saya akan kuliah") elif(hari_ini == "Sabtu"):
print("Saya akan kuliah") elif(hari_ini == "Minggu"):     print("Saya akan libur")
```

Pada contoh diatas, jika program dijalankan maka akan mencetak string "Saya akan libur".

5.10 Pengulangan “Loop”

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.