PRAKTIKUM 2

1. Buatlah contoh koding untuk Tuple dan Dictionary. (koding tidak boleh sama dengan temannya)
2. Simpan dengan nama Praktikum2_NIM.ipnyb

String Python

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")
```

Mengakses Nilai dalam String

Python tidak menggunakan tipe karakter titik koma ; Ini diperlakukan sebagai string dengan panjang satu, sehingga juga dianggap sebagai substring.

Untuk mengakses substring, gunakan tanda kurung siku untuk mengiris beserta indeks atau indeks untuk mendapatkan substring Anda. Sebagai contoh :

```
name = 'John Doe' message = "John Doe belajar bahasa python di Belajarpython"
print ("name[0]: ", name[0])
print ("message[1:4]: ", message[1:4])
```

Bila kode diatas dieksekusi, maka akan menghasilkan hasil sebagai berikut :

```
name[0]: J message[1:4]: ohn
```

Mengupdate String

Anda dapat "memperbarui" string yang ada dengan (kembali) menugaskan variabel ke string lain. Nilai baru dapat dikaitkan dengan nilai sebelumnya atau ke string yang sama sekali berbeda sama sekali. Sebagai contoh

```
message = 'Hello World'
print ("Updated String :- ", message[:6] + 'Python')
```

Bila kode diatas dieksekusi, maka akan menghasilkan hasil sebagai berikut :

```
Updated String :- Hello Python
```

Escape Characters / Karakter Escape Python

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

Operator Spesial String Python

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

Operator Format String Python

Salah satu fitur Python yang paling keren adalah format string operator %. Operator ini unik untuk string dan membuat paket memiliki fungsi dari keluarga printf C () C. berikut adalah contoh sederhananya : `print ("My name is %s and weight is %d kg!" % ('Zara', 21))`

Berikut adalah daftar lengkap simbol yang bisa digunakan bersamaan dengan % :

Operator	Penjelasan
%c	character
%s	Konversi string melalui str () sebelum memformat
%i	Dianggap sebagai bilangan bulat desimal
%d	Dianggap sebagai bilangan bulat desimal
%u	Unsigned decimal integer
%o	Bilangan bulat oktal
%x	Bilangan bulat heksadesimal (huruf kecil)
%X	Bilangan bulat heksadesimal (huruf besar)
%e	Notasi eksponensial (dengan huruf kecil 'e')
%E	Notasi eksponensial (dengan huruf besar 'E')
%f	Bilangan real floating point
%g	Yang lebih pendek dari% f dan% e
%G	Lebih pendek dari% f dan% E

Triple Quote Python

Python triple quotes digunakan dengan membiarkan string untuk ditulis dalam beberapa baris, termasuk kata kerja NEWLINEs, TABs, dan karakter khusus lainnya. Sintaks untuk triple quotes terdiri dari tiga tanda kutip tunggal atau ganda ditulis berturut-turut : Berikut adalah contohnya :

```
kutipantiga = """this is a long string that is made up of
several lines and non-printable characters such as
TAB ( \t ) and they will show up that way when displayed.
NEWLINEs within the string, whether explicitly given like
this within the brackets [ \n ], or just a NEWLINE within
the variable assignment will also show up.
"""
print (kutipantiga)
```

String Unicode Python

Pada Python 3, semua string diwakili dalam Unicode. Sedangkan pada Python 2 disimpan secara internal sebagai 8-bit ASCII, maka diperlukanlampiran 'u' untuk membuatnya menjadi Unicode. Tetapi hal ini tidak lagi diperlukan sekarang. :

Metode String Built-in

Python menyertakan metode built-in berikut untuk memanipulasi string

Metode	Penjelasan
`capitalize()`	Meng-kapitalkan huruf pertama string
`center(width, fillchar)`	Mengembalikan string yang dilapisi dengan fillchar dengan string asli yang dipusatkan pada total width kolom.
`count(str, beg = 0,end = len(string))`	Menghitung berapa kali str yang terjadi dalam string atau dalam substring string jika memulai indeks beg dan end index end diberikan.
`decode(encoding = 'UTF-8',errors = 'strict')`	Dekode string menggunakan codec yang terdaftar untuk pengkodean. Encoding default ke pengkodean string default.
`encode(encoding = 'UTF-8',errors = 'strict')`	Mengembalikan versi string yang dikodekan string; Pada kesalahan, default adalah menaikkan ValueError kecuali jika kesalahan diberikan dengan 'ignore' atau 'replace'.
`endswith(suffix, beg = 0, end = len(string))`	Menentukan apakah string atau substring string (jika memulai indeks memohon dan mengakhiri akhir indeks diberikan) berakhir dengan akhiran; Mengembalikan nilai true jika benar dan salah.

`expandtabs(tabsize = 8)`	Memperluas tab dalam string ke banyak ruang; Default ke 8 spasi per tab jika tabsize tidak tersedia.
`find(str, beg = 0 end = len(string))`	Tentukan jika str terjadi dalam string atau dalam substring string jika memulai indeks beg dan end index end diberikan return index jika ditemukan dan -1 sebaliknya.
`index(str, beg = 0, end = len(string))`	Sama seperti find (), namun menimbulkan pengecualian jika str tidak ditemukan.
`isalnum()`	Mengembalikan true jika string memiliki minimal 1 karakter dan semua karakternya alfanumerik dan false sebaliknya.
`isalpha()`	Mengembalikan true jika string memiliki minimal 1 karakter dan semua karakter adalah abjad dan false sebaliknya.
`isdigit()`	Mengembalikan true jika string hanya berisi digit dan false sebaliknya.
`islower()`	Mengembalikan true jika string memiliki setidaknya 1 karakter casing dan semua karakter casing dalam huruf kecil dan false sebaliknya.
`isnumeric()`	Mengembalikan true jika string unicode hanya berisi karakter numerik dan false sebaliknya.
`isspace()`	Mengembalikan true jika string hanya berisi karakter spasi dan false sebaliknya.
`istitle()`	Mengembalikan true jika string benar "titlecased" dan false sebaliknya.
`isupper()`	Mengembalikan true jika string memiliki setidaknya satu karakter casing dan semua karakter casing ada dalam huruf besar dan false sebaliknya.
`join(seq)`	Merges (concatenates) representasi string elemen dalam urutan seq menjadi string, dengan string pemisah.
`len(string)`	Mengembalikan panjang string
`ljust(width[, fillchar])`	Mengembalikan string berlapis ruang dengan string asli dibiarkan dibenarkan ke kolom lebar total.
`lower()`	Mengonversi semua huruf besar dalam bentuk string menjadi huruf kecil.
`lstrip()`	Menghapus semua spasi utama dalam string.
`maketrans()`	Mengembalikan tabel terjemahan untuk digunakan dalam fungsi terjemahan.
`max(str)`	Mengembalikan karakter alfabetik dari string str.
`min(str)`	Mengembalikan min karakter abjad dari string str.
`replace(old, new [, max])`	Menggantikan semua kemunculan lama dalam string dengan kejadian baru atau paling maksimal jika max diberikan.
`rfind(str, beg = 0,end = len(string))`	Sama seperti find (), tapi cari mundur dalam string.
`rindex( str, beg = 0, end = len(string))`	Sama seperti index (), tapi cari mundur dalam string.

`rindex( str, beg = 0, end = len(string))`	Sama seperti index (), tapi cari mundur dalam string.
`rjust(width,[, fillchar])`	Mengembalikan string berlapis ruang dengan senar asli benar-dibenarkan untuk total kolom lebar.
`rstrip()`	Menghapus semua spasi spasi string.
`split(str="", num=string.count(str))`	Membagi string sesuai dengan pemisah str (ruang jika tidak disediakan) dan mengembalikan daftar substring; Terpecah menjadi paling banyak substring jika diberikan.
`splitlines( num=string.count('\n'))`	Membagi string sama sekali (atau num) NEWLINEs dan mengembalikan daftar setiap baris dengan NEWLINEs dihapus.
`startswith(str, beg=0,end=len(string)`	Determines if string or a substring of string (if starting index beg and ending index end are given) starts with substring str; returns true if so and false otherwise.
`strip([chars])`	Lakukan kedua lstrip () dan rstrip () pada string
`swapcase()`	Kasus invers untuk semua huruf dalam string.
`title()`	Mengembalikan versi string "titlecased", yaitu, semua kata diawali dengan huruf besar dan sisanya huruf kecil.
`translate(table, deletechars="")`	Menerjemahkan string sesuai dengan tabel terjemahan str (256 karakter), menghapus string del.

Tuple Python

Sebuah tupel adalah urutan objek Python yang tidak berubah. Tupel adalah urutan, seperti daftar. Perbedaan utama antara tupel dan daftarnya adalah bahwa tupel tidak dapat diubah tidak seperti List Python. Tupel menggunakan tanda kurung, sedangkan List Python menggunakan tanda kurung siku.

Membuat tuple semudah memasukkan nilai-nilai yang dipisahkan koma. Secara opsional, Anda dapat memasukkan nilai-nilai yang dipisahkan koma ini di antara tanda kurung juga. Sebagai contoh :

```
#Contoh sederhana pembuatan tuple pada bahasa pemrograman python

tup1 = ('fisika', 'kimia', 1993, 2017)
tup2 = (1, 2, 3, 4, 5 )
tup3 = "a", "b", "c", "d"
```

Tupel kosong ditulis sebagai dua tanda kurung yang tidak berisi apa-apa, contohnya : tup1 = (); Untuk menulis tupel yang berisi satu nilai, Anda harus memasukkan koma, meskipun hanya ada satu nilai, contohnya : tup1 = (50,) Seperti indeks String, indeks tuple mulai dari 0, dan mereka dapat diiris, digabungkan, dan seterusnya

Akses Nilai Dalam Tuple Python

Untuk mengakses nilai dalam tupel, gunakan tanda kurung siku untuk mengiris beserta indeks atau indeks untuk mendapatkan nilai yang tersedia pada indeks tersebut. Sebagai contoh :

```
#Cara mengakses nilai tuple

tup1 = ('fisika', 'kimia', 1993, 2017)
tup2 = (1, 2, 3, 4, 5, 6, 7 )

print ("tup1[0]: ", tup1[0])
print ("tup2[1:5]: ", tup2[1:5])
```

Setelah Anda mengeksekusi kode diatas, hasilnya akan seperti dibawah ini :

```
tup1[0]: fisika
```
```
tup2[1:5]: (2, 3, 4, 5)
```

Update Nilai Dalam Tuple Python

Tuple tidak berubah, yang berarti Anda tidak dapat memperbarui atau mengubah nilai elemen tupel. Anda dapat mengambil bagian dari tupel yang ada untuk membuat tupel baru seperti ditunjukkan oleh contoh berikut.

```
tup1 = (12, 34.56)
tup2 = ('abc', 'xyz')

# Aksi seperti dibawah ini tidak bisa dilakukan pada tuple python
# Karena memang nilai pada tuple python tidak bisa diubah
# tup1[0] = 100;

# Jadi, buatlah tuple baru sebagai berikut
tup3 = tup1 + tup2
print (tup3)
```

Hapus Nilai Dalam Tuple Python

Menghapus elemen tuple individual tidak mungkin dilakukan. Tentu saja, tidak ada yang salah dengan menggabungkan tupel lain dengan unsur-unsur yang tidak diinginkan dibuang.

Untuk secara eksplisit menghapus keseluruhan tuple, cukup gunakan del statement. Sebagai contoh

```
tup = ('fisika', 'kimia', 1993, 2017)
print(tup)

# hapus tuple dengan statement del
del tup

# lalu buat kembali tuple yang baru dengan elemen yang diinginkan
tup = ('Bahasa', 'Literasi', 2020)
print("Setelah menghapus tuple :", tup)
```

Operasi Dasar Pada Tuple Python

Tupel merespons operator + dan * sama seperti String; Mereka berarti penggabungan dan pengulangan di sini juga berlaku, kecuali hasilnya adalah tupel baru, bukan string.

Sebenarnya, Tuple merespons semua operasi urutan umum yang kami gunakan pada String di bab sebelumnya. Dibawah ini adalah tabel daftar operasi dasar pada Tuple python

Python Expression	Hasil	Penjelasan
len((1, 2, 3))	3	Length
(1, 2, 3) + (4, 5, 6)	(1, 2, 3, 4, 5, 6)	Concatenation
('Halo!',) * 4	('Halo!', 'Halo!', 'Halo!', 'Halo!')	Repetition
3 in (1, 2, 3)	True	Membership
for x in (1,2,3) : print (x, end = ' ')	1 2 3	Iteration

Indexing, Slicing dan Matrix Pada Tuple Python

Karena tupel adalah urutan, pengindeksan dan pengiris bekerja dengan cara yang sama untuk tupel seperti pada String, dengan asumsi masukan berikut

Dengan asumsi input berikut : `T = ('C++', 'Java', 'Python')`

Python Expression	Hasil	Penjelasan
`T[2]`	`'Python'`	Offset mulai dari nol
`T[-2]`	`'Java'`	Negatif: hitung dari kanan
`T[1:]`	`('Java', 'Python')`	Slicing mengambil bagian

Fungsi Build-in Pada Tuple Python

Python menyertakan fungsi built-in sebagai berikut

Python Function	Penjelasan
`cmp(tuple1, tuple2)`	# Tidak lagi tersedia dengan Python 3
`len(tuple)`	Memberikan total panjang tuple.
`max(tuple)`	Mengembalikan item dari tuple dengan nilai maks.
`min(tuple)`	Mengembalikan item dari tuple dengan nilai min.
`tuple(seq)`	Mengubah seq menjadi tuple.

Dictionary Python

Dictionary Python berbeda dengan List ataupun Tuple. Karena setiap urutanya berisi key dan value. Setiap key dipisahkan dari value-nya oleh titik dua (:), item dipisahkan oleh koma, dan semuanya tertutup dalam kurung kurawal. Dictionary kosong tanpa barang ditulis hanya dengan dua kurung kurawal, seperti ini: {}.

Nilai kamus bisa berupa tipe apa pun, namun key harus berupa tipe data yang tidak berubah seperti string, angka, atau tupel.

Akses Nilai Dalam Dictionary Python

Untuk mengakses elemen Dictionary, Anda dapat menggunakan tanda kurung siku yang sudah dikenal bersama dengan key untuk mendapatkan nilainya. Berikut adalah contoh sederhananya :

```
#Contoh cara membuat Dictionary pada Python

dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}
print ("dict['Name']: ", dict['Name'])
print ("dict['Age']: ", dict['Age'])
```

Update Nilai Dalam Dictionary Python

Anda dapat memperbarui Dictionary dengan menambahkan entri baru atau pasangan nilai kunci, memodifikasi entri yang ada, atau menghapus entri yang ada seperti ditunjukkan pada contoh sederhana yang diberikan di bawah ini.

```
#Update dictionary python

dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}
dict['Age'] = 8; # Mengubah entri yang sudah ada
dict['School'] = "DPS School" # Menambah entri baru

print ("dict['Age']: ", dict['Age'])
print ("dict['School']: ", dict['School'])
```

Hapus Elemen Dictionary Python

Anda dapat menghapus elemen Dictionary individual atau menghapus keseluruhan isi Dictionary. Anda juga dapat menghapus seluruh Dictionary dalam satu operasi.

Untuk menghapus seluruh Dictionary secara eksplisit, cukup gunakan del statement. Berikut adalah contoh sederhana :

```
#Contoh cara menghapus pada Dictionary Python

dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}

del dict['Name'] # hapus entri dengan key 'Name'
dict.clear()     # hapus semua entri di dict
del dict         # hapus dictionary yang sudah ada

print ("dict['Age']: ", dict['Age'])
print ("dict['School']: ", dict['School'])
```

Fungsi Build-in Pada Dictionary Python

Python menyertakan fungsi built-in sebagai berikut :

Fungsi Python	Penjelasan
cmp(dict1, dict2)	Membandingkan unsur keduanya.
len(dict)	Memberikan panjang total Dictionary. Ini sama dengan jumlah item dalam Dictionary.
str(dict)	Menghasilkan representasi string yang dapat dicetak dari Dictionary
type(variable)	Mengembalikan tipe variabel yang lulus. Jika variabel yang dilewatkan adalah Dictionary, maka akan mengembalikan tipe Dictionary.

Method Build-in Pada Dictionary Python

Python menyertakan method built-in sebagai berikut :

Method Python	Penjelasan
dict.clear()	Menghapus semua elemen Dictionary
dict.copy()	Mengembalikan salinan Dictionary
dict.fromkeys()	Buat Dictionary baru dengan kunci dari seq dan nilai yang disetel ke nilai.
dict.get(key, default=None)	For key, nilai pengembalian atau default jika tombol tidak ada dalam Dictionary
dict.has_key(key)	Mengembalikan true jika key dalam Dictionary, false sebaliknya
dict.items()	Mengembalikan daftar dari pasangan tuple dictionary (key, value)
dict.keys()	Mengembalikan daftar key dictionary
dict.setdefault(key, default=None)	Mirip dengan get (), tapi akan mengatur dict [key] = default jika kunci belum ada di dict
dict.update(dict2)	Menambahkan pasangan kunci kata kunci dict2 ke dict
dict.values()	Mengembalikan daftar nilai dictionary