

# ALGORITME BIOINFORMATIKA



Departemen Ilmu Komputer  
Fakultas Matematika dan Ilmu Pengetahuan Alam  
Institut Pertanian Bogor  
2016

Praktikum : Algoritme Bioinformatika

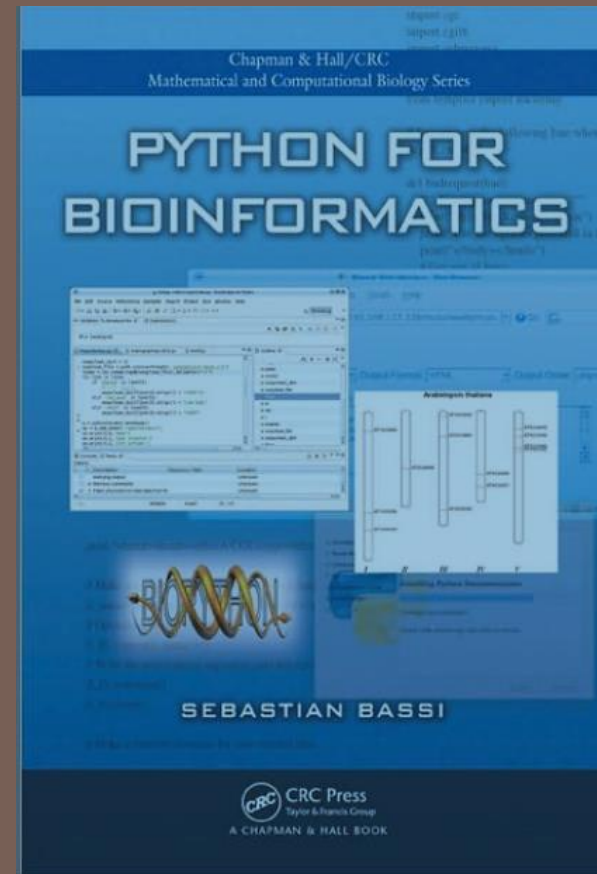
# Praktikum

Pertemuan	Topik
1	Pengantar Python
2	Format data : FASTA, FASTQ, PDB Next Generation Simulator (MetaSIM)
3	Basisdata : NCBI GenBank, SRA, PDB Pengantar mengenai BLAST
4	Pairwise alignment dengan python
5	Aplikasi sekuens alignment dan Multiple Sequence Alignment
6	Ugene
7	phylogenetic tree dengan Clustal
8	Bowtie
9	SAMTools
10	Projek
11	I-Tasser
12	Tassel, R
13	R utk graph mining
14	Presentasi

# PYTHON : PENDAHULUAN



Departemen Ilmu Komputer  
Fakultas Matematika dan Ilmu Pengetahuan Alam  
Institut Pertanian Bogor  
2016



Praktikum : Algoritme Bioinformatika

# Mengapa Python

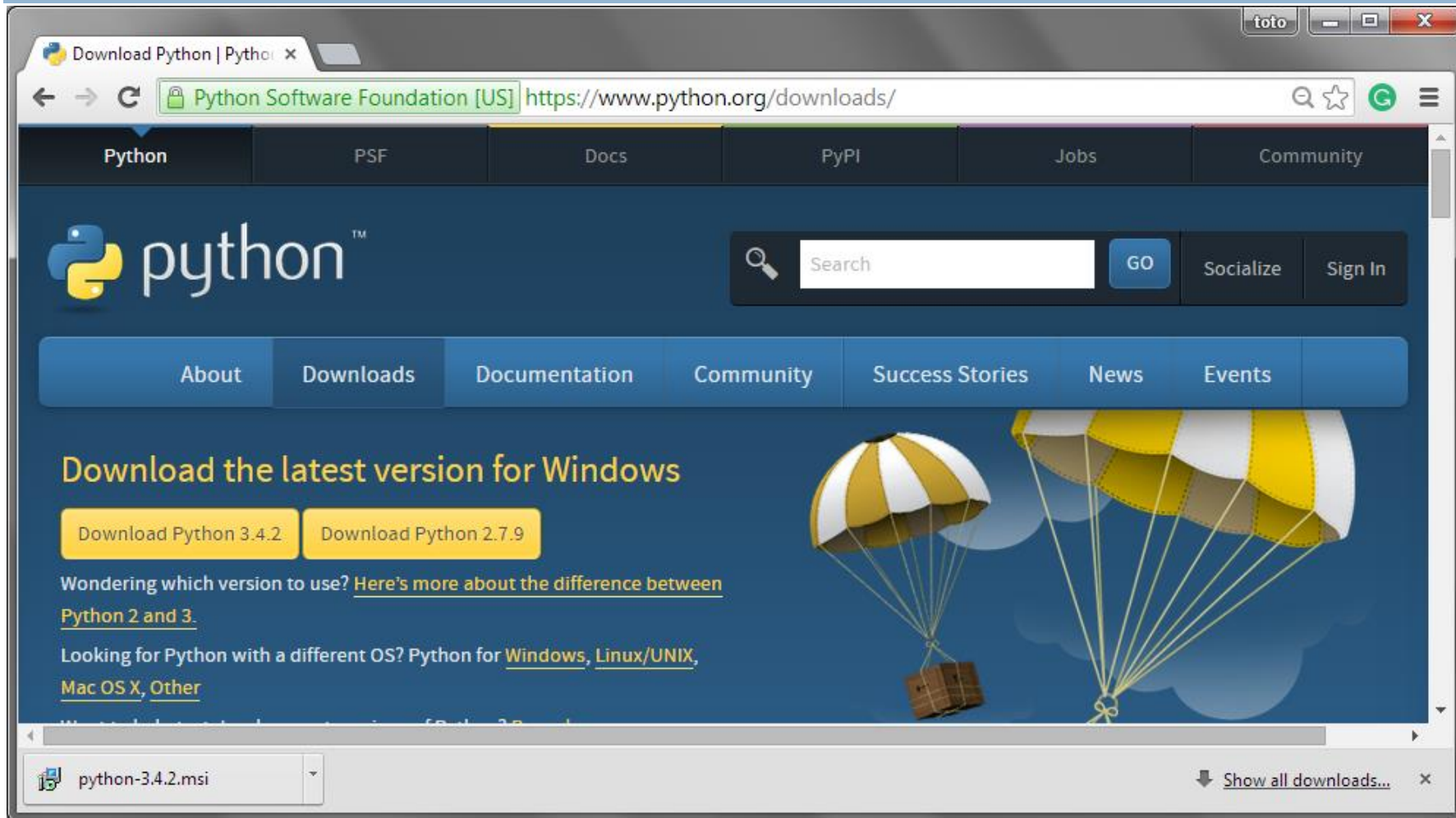
- Readability
- Built-in feature
- Availability of third party modules
- High level built-in data structure
- Multi-paradigm
- Extensibility
- Open Source
- Cross Platform
- Thriving community

# Di mana Python digunakan ?

- Generic OS Service (os,io, time)
- File and directory access (shutil, tempfile,glob)
- Data compression and archiving (zipfile, gzip, bz2)
- Internet data handling (email,mimertools/rfc822)
- Internet Protocol (cgi,urllib,urlparse)
- String service (string, codec, re,unicodedata)

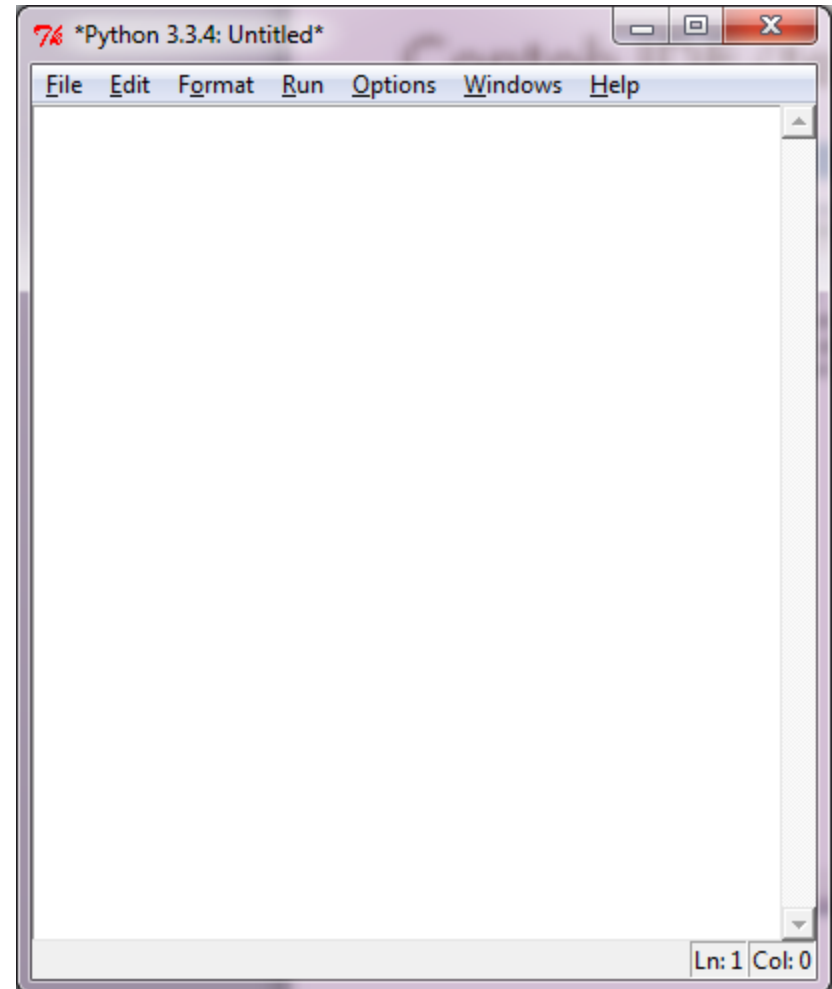
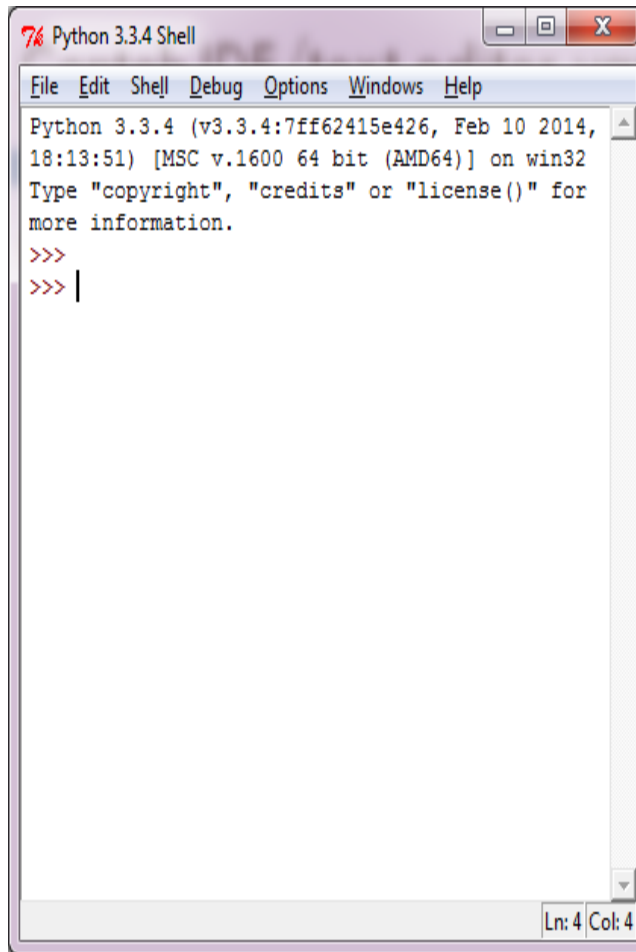
# Instalasi Python

- Website : (<http://www.python.org/download>)



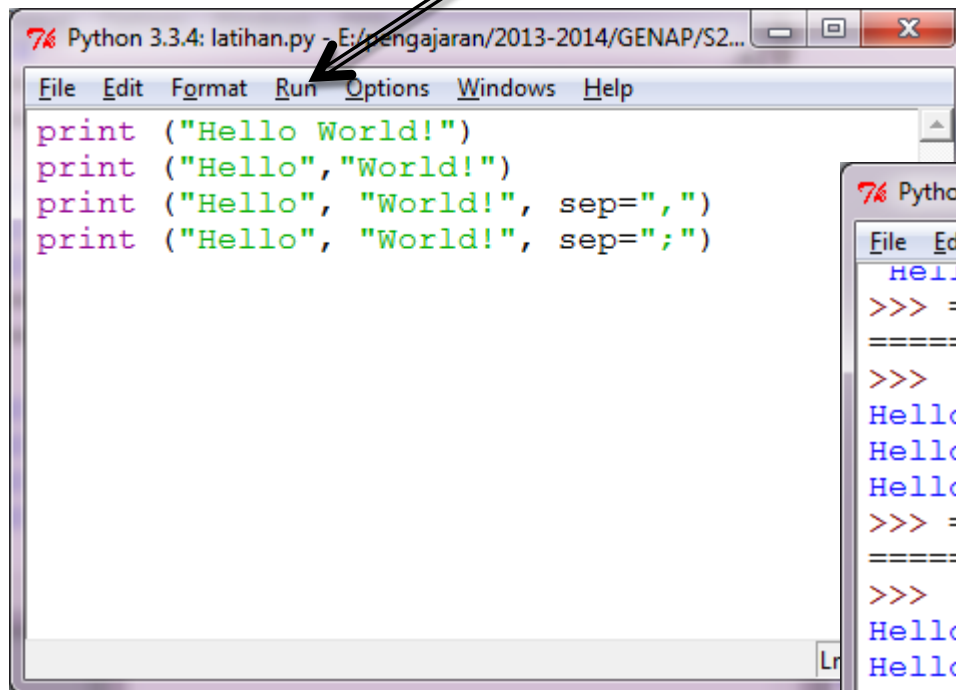
# Contoh IDE/text editor yang digunakan

## □ IDLE



# Python : First Use

## □ Mencetak Hello World

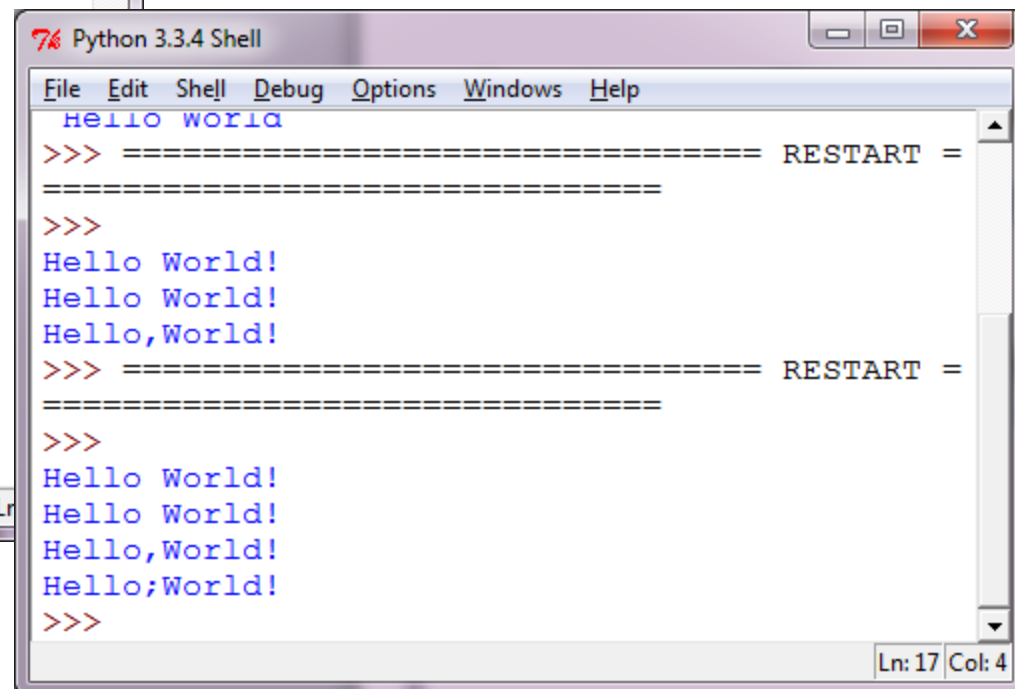


A screenshot of a Python 3.3.4 editor window. The title bar reads "Python 3.3.4: latihan.py - E:/pengajaran/2013-2014/GENAP/S2...". The menu bar includes File, Edit, Format, Run, Options, Windows, and Help. The code editor contains the following Python code:

```
print ("Hello World!")
print ("Hello", "World!")
print ("Hello", "World!", sep=",")
print ("Hello", "World!", sep=";")
```

An arrow points from the "Run" menu item to the "Python 3.3.4 Shell" window.

editor



A screenshot of a Python 3.3.4 Shell window. The title bar reads "Python 3.3.4 Shell". The menu bar includes File, Edit, Shell, Debug, Options, Windows, and Help. The shell displays the output of the code from the editor window:

```
Hello World!
>>> ===== RESTART =
=====
>>>
Hello World!
Hello World!
Hello,World!
>>> ===== RESTART =
=====
>>>
Hello World!
Hello World!
Hello,World!
Hello;World!
>>>
```

The status bar at the bottom right shows "Ln: 17 Col: 4".

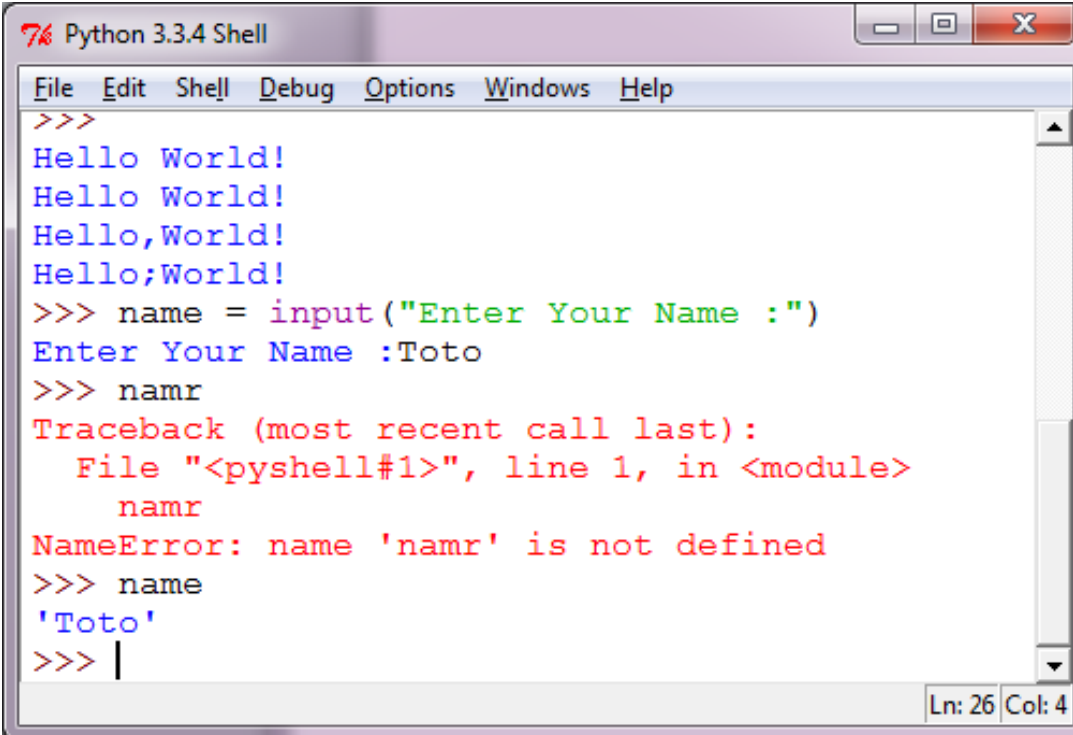
Compiler/shell



# Raw Input

- Membaca masukan dari standar input pada Python versi 3.x

```
name = input ("Enter Your Name:")
```



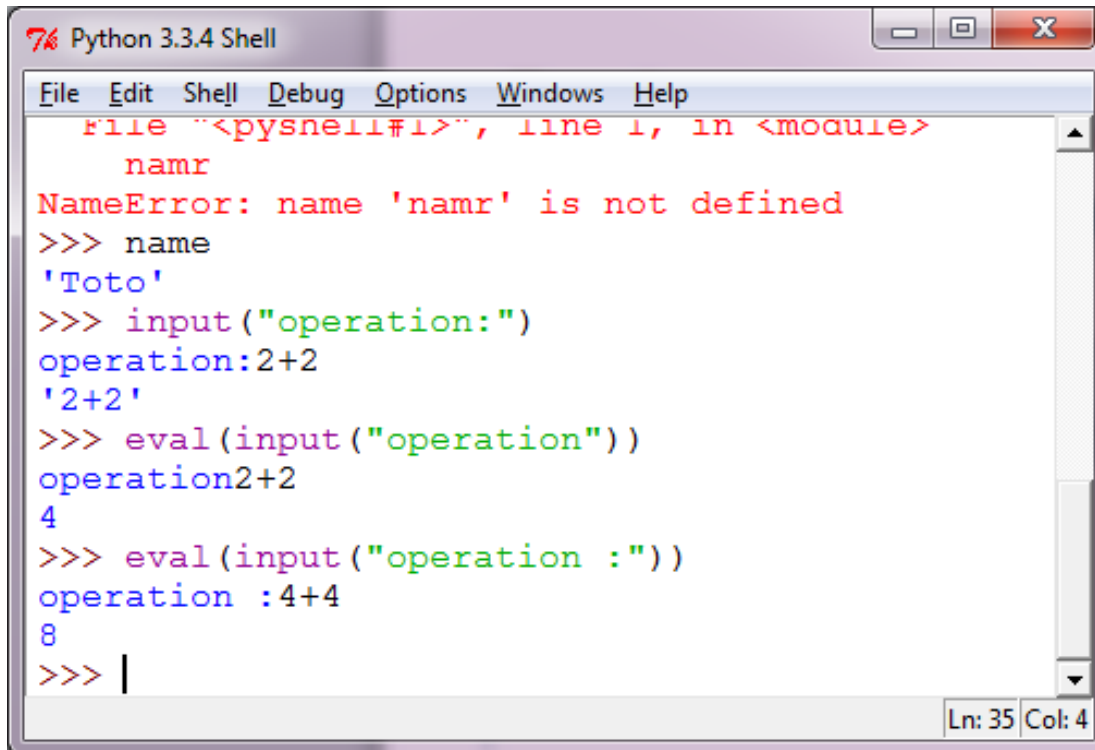
```
Python 3.3.4 Shell
File Edit Shell Debug Options Windows Help
>>>
Hello World!
Hello World!
Hello,World!
Hello;World!
>>> name = input("Enter Your Name :")
Enter Your Name :Toto
>>> namr
Traceback (most recent call last):
  File "<pyshell#1>", line 1, in <module>
    namr
NameError: name 'namr' is not defined
>>> name
'Toto'
>>> |
Ln: 26 Col: 4
```

# Evaluasi suatu ekspresi

## Contoh

```
input(" Operation :")
```

```
Eval (input("operation"))
```

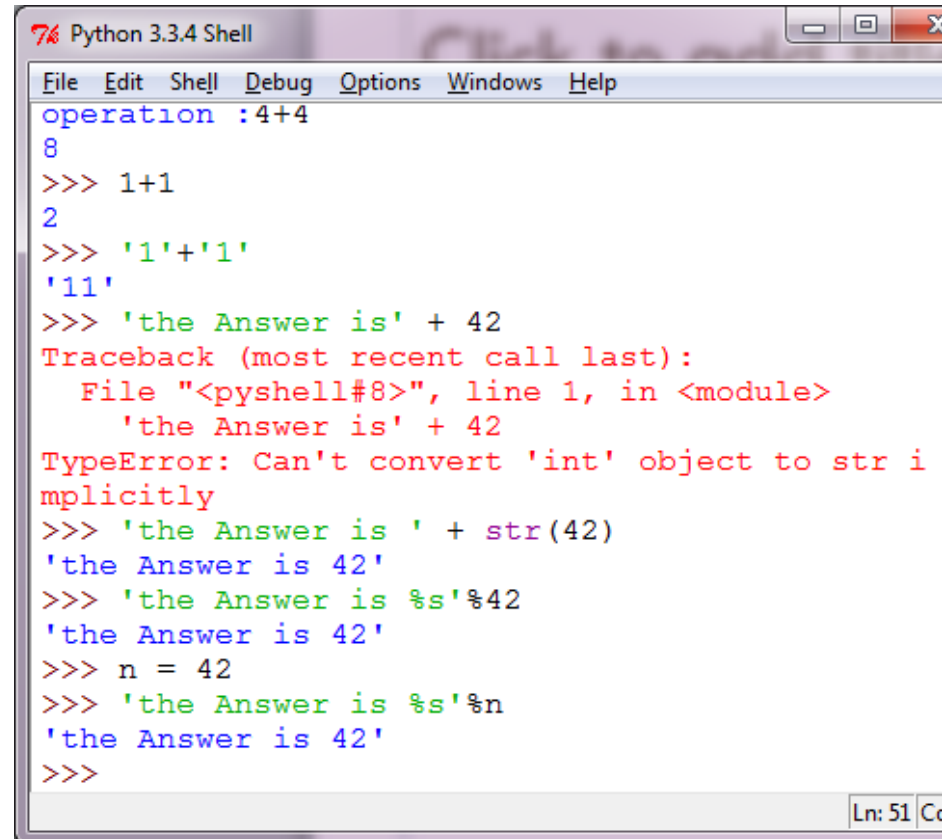


```
Python 3.3.4 Shell
File Edit Shell Debug Options Windows Help
File "<pysnelli#1>", line 1, in <module>
    namr
NameError: name 'namr' is not defined
>>> name
'Toto'
>>> input("operation:")
operation:2+2
'2+2'
>>> eval(input("operation"))
operation2+2
4
>>> eval(input("operation :"))
operation :4+4
8
>>> |
```

Ln: 35 Col: 4

# example

```
>> 1 + 1
>> '1'+'1'
>> 'The Answer is ' + 42
>> 'The Answer is ' + str(42)
>> 'the Answer is %s' %42
>> n = 42
>> 'the Answer is %s', %n
```



```
Python 3.3.4 Shell
File Edit Shell Debug Options Windows Help
operation :4+4
8
>>> 1+1
2
>>> '1'+'1'
'11'
>>> 'the Answer is' + 42
Traceback (most recent call last):
  File "<pyshell#8>", line 1, in <module>
    'the Answer is' + 42
TypeError: Can't convert 'int' object to str implicitly
>>> 'the Answer is ' + str(42)
'the Answer is 42'
>>> 'the Answer is %s'%42
'the Answer is 42'
>>> n = 42
>>> 'the Answer is %s'%n
'the Answer is 42'
>>>
```

Ln: 51 Co

# Operasi Matematika

## □ Tabel Sintaks Operasi Matematika

Symbol	Deskripsi
+	Penjumlahan
-	Pengurangan
*	Perkalian
/	Pembagian
**	Pangkat
%	Sisa bagi

# Beberapa Ketentuan

- Parentheses : Menunjukkan urutan operator yang akan dievaluasi oleh kompiler.  
>>>  $2 * (3-2)$
- Exponensial (Pangkat) :  $2^{**}2+1$  akan menghasilkan 5 bukan 8  
>>>  $2^{**}2 + 1$
- Multiplication (perkalian) dan Division (pembagian) memiliki derajat yang sama :  $2*2-1$  akan menghasilkan 3 bukan 2
- Addition (penambahan) dan Substraction (pengurangan) memiliki derajat yang sama. Dibandingkan operasi lainnya, operasi ini yang paling akhir dievaluasi
- Disingkat : **PEMDAS**

# Latihan Operasi Matematika

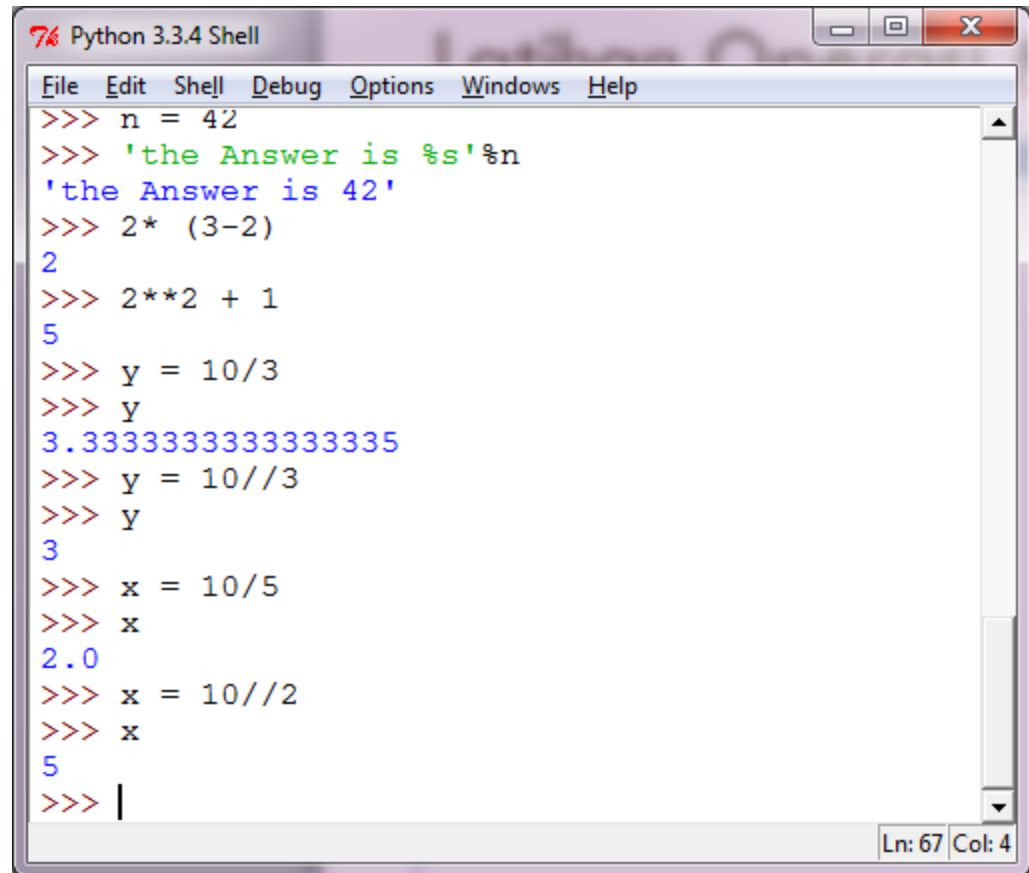
```
>> y = 10/3
```

```
>> x = 10/2
```

Bedakan

```
>> y = 10//3
```

```
>> x = 10//2
```



```
Python 3.3.4 Shell
File Edit Shell Debug Options Windows Help
>>> n = 42
>>> 'the Answer is %s'%n
'the Answer is 42'
>>> 2* (3-2)
2
>>> 2**2 + 1
5
>>> y = 10/3
>>> y
3.3333333333333335
>>> y = 10//3
>>> y
3
>>> x = 10/5
>>> x
2.0
>>> x = 10//2
>>> x
5
>>> |
```

Ln: 67 Col: 4

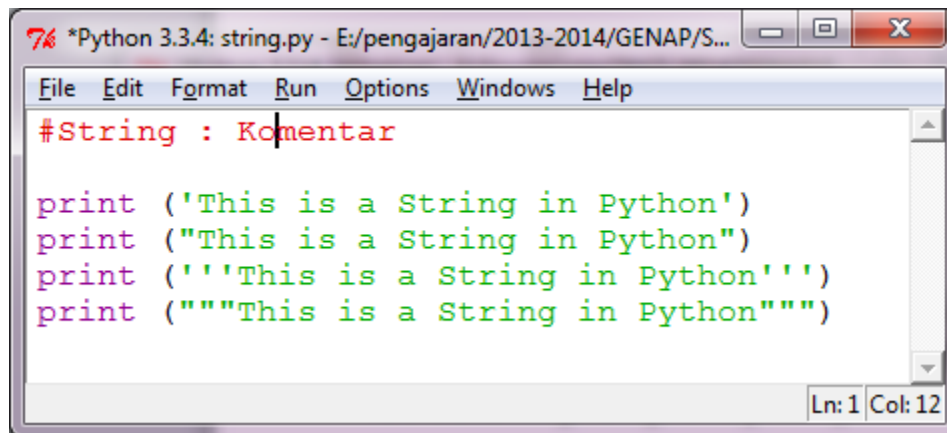
# Basic Programming : Data Types

## STRING

- Merupakan serangkaian simbol yang dibatasi dengan tanda single quote ('), double quote ("), single triple quote(''') atau double triple quote('' ''')

- Ilustrasi

'This is a String in Python'

A screenshot of a Python 3.3.4 IDLE window. The title bar reads '\*Python 3.3.4: string.py - E:/pengajaran/2013-2014/GENAP/S...'. The menu bar includes File, Edit, Format, Run, Options, Windows, and Help. The code editor contains the following text: 

```
#String : Komentar

print ('This is a String in Python')
print ("This is a String in Python")
print ('''This is a String in Python''')
print ("""This is a String in Python""")
```

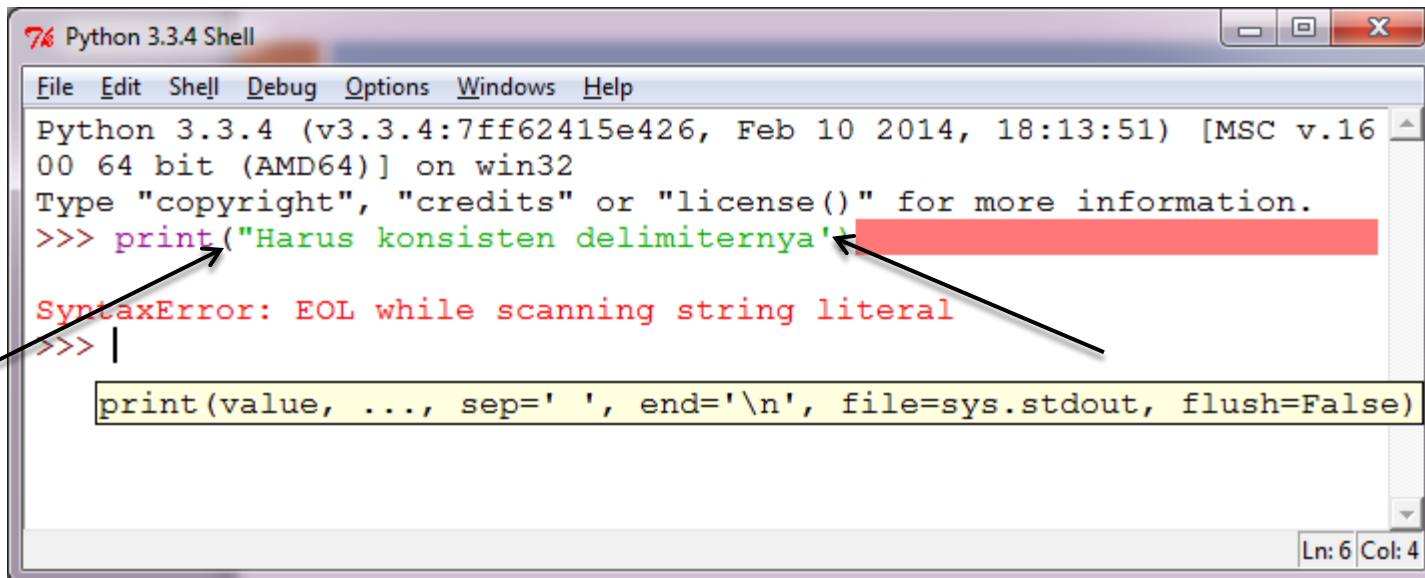
The status bar at the bottom right shows 'Ln:1 Col:12'.

# Kombinasi antar jenis string

```
>>> print (' Kami memiliki "string double quote" di dalam string single quote')
Kami memiliki "string double quote" di dalam string single quote
>>> |
```

Ln: 77 Col: 4

```
>> print (" Harus konsisten delimiternya ")
```



```
Python 3.3.4 Shell
File Edit Shell Debug Options Windows Help
Python 3.3.4 (v3.3.4:7ff62415e426, Feb 10 2014, 18:13:51) [MSC v.16
00 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> print('Harus konsisten delimiternya')
SyntaxError: EOL while scanning string literal
>>> |

print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)
```

Ln: 6 Col: 4



# Manipulasi String

- String are immutable. Once a string is created, it can't be modified.
- If we need change a string, what we can do is to make a derivated string
- To become string as parameter in function and get the return value

# Manipulasi String: Contoh

```
>> signal_peptide = "MASKATILLAFTLLFACTIA"  
>> signal_peptide.lower()  
>> signal_peptide  
>> signal_peptide = signal_peptide.lower()  
>> signal_peptide
```

# Teknik Asosiasi String

- ❑ replace
- ❑ count
- ❑ find
- ❑ Index
- ❑ Split
- ❑ Join
- ❑ Etc...

# Replace

- Bentuk umum : `replace(old,new[,count])`
- Untuk melakukan bagian di dalam string (old) dengan suatu nilai baru (new)

```
>> DNA_Seq = "TTGCTAGTTT"
```

```
>> mRNA_Seq = DNA_Seq.replace("T","U")
```

```
>> mRNA_Seq
```

# Count

- Bentuk umum : `count(sub[,start[,end]])`
- Menghitung berapa kali substring itu muncul yang terletak antara start dan end

```
>> DNA_Seq = "TTGCTAGTTT"
```

```
>> t=DNA_Seq.count("T")
```

```
>> g=DNA_Seq.count("G")
```

```
>> float(t+g)/len(DNA_Seq)*100
```

# Find

- Bentuk umum : `find(sub[,start[,end]])`
- Mengembalikan informasi posisi substring *sub* yang terletak antara *start* dan *end*
- `start` : default = 0, `end` : length of string
- Jika tidak ada maka bernilai -1

```
str1 = "ATGTCC";  
str2 = "C";  
print (str1.find(str2));  
print (str1.find(str2, 1));  
print (str1.find(str2, 2));
```

# index

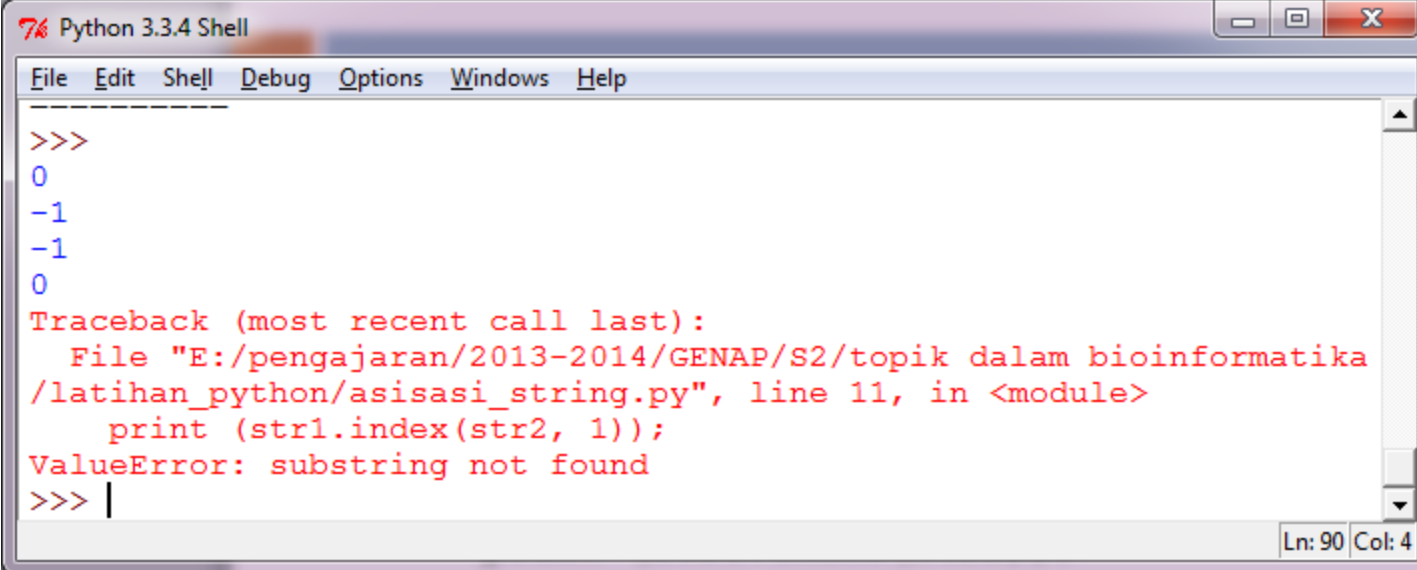
- Sama dengan find, tapi jika tidak ditemukan compiler akan memberikan peringatan, **BUKAN dengan -1**

- Contoh

```
str1 = "ACGTCC";  
str2 = "A";  
print (str1.index(str2));  
print (str1.index(str2, 1));  
print (str1.index(str2, 2));
```

# Ilustrasi index

```
str1 = "ACGTCC";  
str2 = "A";  
print (str1.index(str2));  
print (str1.index(str2, 1));  
print (str1.index(str2, 2));
```



The screenshot shows a Python 3.3.4 Shell window with a menu bar (File, Edit, Shell, Debug, Options, Windows, Help) and a toolbar. The main text area contains the following text:

```
>>>  
0  
-1  
-1  
0  
Traceback (most recent call last):  
  File "E:/pengajaran/2013-2014/GENAP/S2/topik dalam bioinformatika  
/latihan_python/asisasi_string.py", line 11, in <module>  
    print (str1.index(str2, 1));  
ValueError: substring not found  
>>> |
```

The status bar at the bottom right indicates "Ln: 90 Col: 4".



# split

- Bentuk umum : `split([sep [,maxsplit]])`
- Membagi string dengan separator dan mengembalikan dalam bentuk LIST
- Default separator : white space

```
>> "this string has words separated by space".split()  
>> "Toto Haryanto, Ilmu Komputer, FMIPA, IPB ".split(",")  
>> "Toto Novianto; Teknik Kimia; FT; UNDIP".split(";")
```

# Join

- Bentuk umum : `join(seq)`
- Inverse dari `split`
- Kegunaan : menggabungkan  
`' '.join(['A', 'C', 'G', 'T'])`

# TUGAS

- Download file format .FASTA dari NCBI
- Lakukan ekstraksi informasi dari file tersebut.

Contoh file.fasta

```
>4LFU:A|PDBID|CHAIN|SEQUENCE
```

```
MQDKDFFSWRRTMLLRFORMETAEEVYHEIELQAQQLEYDYYSLCVRHPVPF  
TRPKVAFYTNYPEAWVSYYQAKNFLAIDPVLNPENFSQGHLMWNDDLFSEAQ  
PLWEAARAHGLRRGVTQYLMLPNRALGFLSFSRCSAREIPILSDELQLKMQL  
LVRESLMALMRLNDEIVMTPEMNFSKREKEILRWTAEGKTSAEIAMILSISE  
NTVNFHQKNMQKKINAPNKTQVACYAAATGLILEHHHHHH
```

# List : Tipe data pada Python

- List merupakan tipe data yang paling banyak digunakan di dalam bidang bioinformatika
- List direpresentasikan dalam dengan format: [,]
- List dapat dihasilkan dari penggunaan fungsi split
- contoh :

```
>>> "Toto Haryanto, Ilmu Komputer, FMIPA,IPB".split(",")
```

# List

- `>>> first_list = [1,2,3,4,5]`

- **Contoh list dengan tipe data yang berbeda**

```
>>> list_lain = [1,"dua",3,4,"akhir"]
```

- **List Bersarang**

```
>>> list_bersarang = [1,"dua",first_list,3,4,"akhir"]
```

- **List Kosong** : terkadang dibutuhkan untuk inisialisasi saat akan menambah data

```
list_kosong = []
```

# Inisialisasi List

- Kalau kita mengetahui list akan memiliki 5 element, maka dapat didefinisikan sbb.

```
>>> codons = [None] * 5
```

```
>>> codons
```

# Cara Lain Inisialisasi List

- List dapat dibuat dari list lain

Dianalogikan sebagai suatu set himpunan dalam notasi matematika

Misalkan A adalah suatu himpunan dengan anggota

1,2,3,4,5  $\rightarrow A = \{0,1,2,3,4,5\}$ . Kemudian B

didefinisikan  $B = \{3*x \mid x \in A\}$ . Maka  $B = \{0,3,6,9,12,15\}$

```
>>> A = [1, 2, 3, 4, 5]
```

```
>>> B = [3*x for x in A]
```

# Mengakses Elemen List

```
>>> first_list = [1,2,3,4,5]
```

```
>>> first_list[0]
```

```
>>> first_list[1]
```

- List dapat diakses dari kanan menggunakan bilangan negatif

```
>>> first_list = [1,2,3,4,5]
```

```
>>> first_list[-1]
```

```
>>> first_list[-4]
```



# Menjadikan Objek Sebagai List

- Apabila kita memiliki sekuens yang belum memiliki tipe data list dapat dibuat dengan menggunakan fungsi **list()**

```
>>> sequences = "accgtaaaccttt"
```

```
>>> make_list = list(sequence)
```

# Copy Reference List

```
>>> a = [1, 2, 3, 4]
```

```
>>> b = a
```

```
>>> b.pop()
```

```
>>> a
```

□ Keterangan:

tanda “=” tidak berarti men-copy value dari a,  
namun yang di-copy adalah reference/pointer

# 2 Cara Meng-Copy Value List

## □ Cara 1 : dengan menggunakan module copy

```
>>> import copy
>>> a = [1, 2, 3, 4]
>>> b=copy.copy(a)
>>> b.pop()
>>> a
```

## □ Cara 2 : tanpa menggunakan modul copy

```
>>> a = [1, 2, 3, 4]
>>> b=a[:]
>>> b.pop()
>>> a
```

# Modifikasi List

Nama Fungsi	Deskripsi Fungsi
s.append(x)	menambahkan elemen x pada list s
s.insert(position,x)	menyisipkan pada posisi <i>posisi</i> suatu elemet x
s.count(x)	menghitung berapa banyak elemen x pada list s
s.index(x)	mengembalikan posisi elemen x pada list s
s.remove(x)	menghapus elemen x pada list s
s.reverse(x)	membalik list s
s.sort(x)	mengurutkan list s

# Latihan : Teladan Modifikasi List

```
>>> first_list = [1,2,2,2,3,3,3,3,4]
```

```
>>> first_list.pop()
```

```
>>> first_list = [1,2,2,2,3,3,3,3,4]
```

```
>>> first_list.remove(3)
```

```
>>> first_list = [1,2,2,3,3,3,3,4]
```

```
>>> first_list.reverse()
```

```
>>> first_list = [ 6,7,3,5,1,2,8,7,9,3]
```

```
>>> first_list.sort()
```

# Latihan : Teladan Modifikasi List

```
>>> first_list = [1,2,3,4]
```

```
>>> first_list.append(98)
```

```
>>> first_list = [1,2,3,4]
```

```
>>> first_list.insert(2,55)
```

```
>>> first_list = [1,2,2,3,3,3,3,4]
```

```
>>> first_list.count(3)
```

```
>>> first_list = [1,2,2,3,3,3,3,4]
```

```
>>> first_list.index(2)
```

```
>>> first_list.index(3)
```

```
>>> first_list.index(4)
```

# Common Property of Sequence

## Slicing

- Notasi (:)
- *Slicing* digunakan untuk memilih sebagian dari sequence yang kita miliki

```
>>> my_seq = "Python"  
>>> part = my_seq[0:2]  
>>> part = my_seq[:2]  
>>> part = my_seq[4:6]  
>>> part = my_seq[4:]  
>>> part = my_seq[1:5:2]
```

# Common Property of Sequence

## MEMBERSHIP TEST

- Untuk memeriksa apakah dalam suatu elemen ada di dalam sequence atau tidak

```
>>> point = (23,56,11)
```

```
>>> 11 in point
```

```
>>> my_seq = "MVALLLLASSTTAA"
```

```
>>> "X" in my_seq
```



# Common Property of Sequence

## CONCATENATION

- Untuk menggabungkan dua atau lebih sequence

```
>>> point = (23, 56, 11)
```

```
>>> point2 = (2, 6, 7)
```

```
>>> gabung = point+point2
```

```
>>> DNASeq = "ATGCTGTAGTAGCTGGATTA"
```

```
>>> TATAbox = "TATAA"
```

```
>>> gabung = DNASeq+TATAbox
```

# Common Property of Sequence

## **LEN, MAX, MIN**

```
>>> point = (55, 23, 11)
```

```
>>> len(point)
```

```
>>> max(point)
```

```
>>> min(point)
```

## **MAX AND MIN FOR STRING**

```
>>> MySeq = "ACCTAGGTTATATAGG"
```

```
>>> max(MySeq)
```

```
>>> min(MySeq)
```



# Pengelolaan File (I/O) dengan Python

# MENGELOLA FILE

---

- Terminologi : Parsing
- Memahami unit data dari file teks
- Dua aktivitas :
  - ▣ Baca (Read)
  - ▣ Tulis (Write)

# Membaca File

- ▶ Terdapat tiga tahap dalam membaca file pada python
  1. Membuka File :
    - Fungsi yang digunakan : open
    - Sebagai *filehandle* yang merujuk pada file yang sedang kita buka
    - Memiliki dua parameter, yaitu **file\_name** dan **mode**
  2. Membaca File (Terdapat tiga cara):
    - Read() : membaca file secara keseluruhan
    - Readline() : membaca per baris dalam bentuk string
    - Readlines() : membaca keseluruhan dan menjadikan suatu list dengan element setiap barisnya
  3. Menutup file
    - Sintaks : `filehandle.close()`

# Teladan Membuka File

# operasi file

# membuka file lengkap dengan direktori (path)

```
buka_file = open('E:/Latihan_Pyhton/helloworld.py')
```

```
buka_file
```

# membuka file di dalam folder yang sama

```
buka_file1 = open('coba.txt')
```

```
buka_file1
```

# Membaca *file* dalam format FASTA

- ▶ Salah satu format *file* yang banyak digunakan untuk menyimpan data biologi adalah format Fasta.
- ▶ Secara umum *file* dengan format Fasta adalah
  - ▶ Pada baris pertama diawali dengan tanda “>” lalu identifier
  - ▶ Baris berikutnya adalah sequence
- ▶ contoh file dalam format fasta dapat diunduh di :
  - ▶ <http://www.rcsb.org/pdb/explore/remediatedSequence.do?structureId=2LZP&bionumber=1>

```
>2LZP:A | PDBID | CHAIN | SEQUENCE  
DTEIIGGLTIPPVVALVVMRSRFGFFAHLPR
```

# Teladan

Diketahui File dalam format FASTA sebagai berikut

Nama file : seqA.fas

```
>O00626|HUMAN Small inducible cytokine A22.  
MARLQTALLVVLVLLAVALQATEAGPYGANMEDSVCCRDYVRYRL  
PLRVVKHIFYWTSDS<=  
CPRPGVVLLTFRDKEICADPR  
VPWVKMILNKLSQ
```



# Instruksi

---

- ❑ Baca file tersebut
- ❑ Pisahkan Nama Identifier dan Sequence-nya
- ❑ Gabungkan sequencenya

# JAWAB 1: Teladan

```
# MEMBACA FASTA DENGAN READ
```

```
# membuka file seqA.fas
```

```
fh = open('seqA.fas')
```

```
myfile = fh.read() # membaca keseluruhan file
```

```
# mengambil list indeks 0
```

```
# lalu di dalam list indeks 0 tsb diambil dari indeks 1 s.d newline
```

```
name = myfile.split('\n')[0][1:]
```

```
# melakukan penggabungan sequence mulai indeks 1 s.d EOF yang terpisah oleh  
newline
```

```
sequence = ''.join(myfile.split('\n')[1:])
```

```
print("The name is : %s " %name)          # cetak inisial
```

```
print("The sequence is %s" %sequence)     # cetak sequence
```

```
fh.close()                                # tutup
```

# JAWAB 2: Teladan

```
# MEMBACA FASTA DENGAN READLINE
# ingat sifat dari readline
# mengembalikan nilai string hanya 1 baris dalam 1 file
fh = open('seqA.fas')
baris_pertama = fh.readline()
# mendefinisikan baris pertama dari indeks 1 sampai paling
kanan (-1)
name = baris_pertama[1:-1]
sequence = ""      # inisialisasi string kosong
while True :
    baris = fh.readline()
    if baris == "":
        break
    else :
        sequence += baris.replace('\n','')
print("The name is %s " %name)          # cetak name
print("The sequence is %s " %sequence)  # cetak sequence
fh.close()
```

# JAWAB 3: Teladan

# Menggunakan fungsi FOR untuk looping

```
fh = open("seqA.fas")
name = fh.readline()[1:-1]
sequence = ""
for baris in fh:
    sequence += baris.replace('\n', '')

print("The name is %s " %name)           # cetak name
print("The sequence is %s " %sequence)   # cetak sequence
fh.close()
```

# Membaca file CSV (Comma Separated Values)

## Contoh File CSV

# Membaca file csv dengan python

# Format data sbb

```
MarkerID,LenAmpForSeq,MotifAmpForSeq
```

```
TKO001,119,AG(12)
```

```
TKO002,255,TC(16)
```

```
TKO003,121,AG(5)
```

```
TKO004,220,AG(9)
```

```
TKO005,238,TC(17)
```

# Jawab 1: Teladan 2

## Program Membaca *file* CSV

```
# inisialisasi
tlen = 0; n=0
fh = open('seqB.csv')
fh.readline()
for line in fh :
    data = line.split(",")
    tlen += int(data[1])
    n += 1

print("Rata-rata panjang adalah ", tlen/float(n))
fh.close()
```

# Jawab 2: Teladan 2

## # Menggunakan modul CSV

```
import csv
tlen = 0
data = list(csv.reader(open('seqB.csv'))))

for x in range(1, len(data)):
    tlen += int(data[x][1])

print (float(tlen) / (len(data) - 1))
```

# Menulis File

- ▶ Secara umum, sama dengan proses membaca *file*
  1. Membuka File :
    - Fungsi yang digunakan : `open`
    - Sebagai *filehandle* yang merujuk pada file yang sedang kita buka
    - Memiliki dua parameter, yaitu `file_name` dan `mode`
  2. Menulis ke dalam File (Terdapat tiga cara):
    - `Filehandle.write(string)`
  3. Menutup file
    - Sintaks : `filehandle.close()`



# Teladan 3

```
# MENULIS FILE
```

```
fh = open('newfile.txt', 'w')
```

```
fh.write(">4LFU:A|PDBID|CHAIN|SEQUENCE  ")
```

```
fh.close()
```

```
fh = open('newfile.txt', 'a')
```

```
fh.write("DTEIIGGLTIPPVVALVMSRFGFFAHLLPR  ")
```

```
fh.close()
```

# Pertanyaan

1. Lakukan pembacaan file fasta di atas ?
2. Ambil informasi nama protein dari file tersebut
3. fragmen asam amino dari file fasta tersebut
4. Hitung prosentasi setiap asam amino yang terdapat pada fragment tersebut
5. dikumpulkan pekan depan dalam softcopy .doc sebagai laporan dan .py sebagai sourcecode

# TUGAS

- ▶ Download file fasta dari PDB ID pada database <http://www.rcsb.org/pdb/home/home.do>
  - ▶ 2mk2
  - ▶ 3w6a
  - ▶ 3beq
  - ▶ 3smb
  - ▶ 2lw7
  - ▶ 2lxh
- ▶ Lakukan ekstraksi sequence setiap kelas
- ▶ Hitung distribusi asam amino setiap kelas dan simpan hasilnya dalam suatu file

# SELESAI



*Semangtlah terhadap segala sesuatu yang bermanfaat bagi mu  
Mintalah pertolongan kepada Robmu  
Janganlah merasa lemah*