



**MAHIDOL
UNIVERSITY**
Wisdom of the Land

[SCPY204]

Computer Programing

for Physicists

Class 05: 15 Feb 2018

Content: Program's input/output, introduction to Python programming

Instructor: Puwis Amatyakul



2018

**“Valentine’s after
party”**

Today's Goals

Part I: Review, Q&A

Part II: Introduction to Python

Part III: Again! Exercises

Part IV: File I/O

C-Array: pitfalls

Today's Goals

Part I.I: File I/O in C

Part II: Introduction to Python

Part III: Again! Exercises

Part IV: File I/O

C: File I/O

Step 1: Opening file(s)

```
FILE *fopen( const char * filename, const char * mode );
```

Step 2: Write / Read

```
fprintf(fp, "This is testing for fprintf...\n");
```

```
fputs("This is testing for fputs...\n", fp);
```

```
fscanf(fp, "%s", buff);
```

```
fgets(buff, 255, (FILE*)fp);
```

Step 3: Closing file(s)

```
fclose( FILE *fp );
```

C: File I/O

Writing

```
#include<stdio.h>
int main(){
    FILE*fp;
    fp=fopen("test.txt","w");
    fprintf(fp,"This is testing for fprintf...\n");
    fputs("This is testing for fputs...\n",fp);
    fclose(fp);
    return 1;
}
```

C: File I/O

Reading

```
#include <stdio.h>

main() {

    FILE *fp;
    char buff[255];

    fp = fopen("test.txt", "r");
    fscanf(fp, "%s", buff);
    printf("1 : %s\n", buff );

    fgets(buff, 255, (FILE*)fp);
    printf("2: %s\n", buff );

    fgets(buff, 255, (FILE*)fp);
    printf("3: %s\n", buff );
    fclose(fp);

}
```


C: File I/O

Sr.No.	Mode & Description
1	r Opens an existing text file for reading purpose.
2	w Opens a text file for writing. If it does not exist, then a new file is created. Here your program will start writing content from the beginning of the file.
3	a Opens a text file for writing in appending mode. If it does not exist, then a new file is created. Here your program will start appending content in the existing file content.
4	r+ Opens a text file for both reading and writing.
5	w+ Opens a text file for both reading and writing. It first truncates the file to zero length if it exists, otherwise creates a file if it does not exist.
6	a+ Opens a text file for both reading and writing. It creates the file if it does not exist. The reading will start from the beginning but writing can only be appended.

C: File I/O

Ex 1: Can you write 100 random number into a file?

Hint: `#include <stdio.h>`
`#include <stdlib.h>`

```
int main () {
    int i, n;
    time_t t;

    n = 5;

    /* Intializes random number generator */
    srand((unsigned) time(&t));

    /* Print 5 random numbers from 0 to 49 */
    for( i = 0 ; i < n ; i++ ) {
        printf("%d\n", rand() % 50);
    }

    return(0);
}
```

Today's Goals

Part I: Review, Q&A

Part II: Introduction to Python

Part III: Again! Exercises

Part IV: File I/O

Python: Introduction



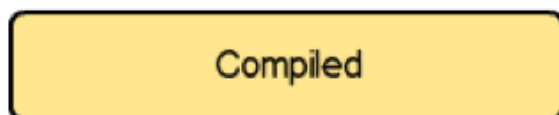
Guido van Rossum
Dutch programmer

About Python

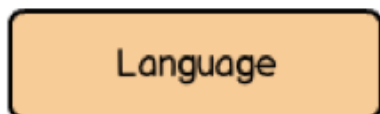
- Python is a high-level programming language created by Guido van Rossum.
- First released in 1991.
- It can be classified as an interpreted language used for general-purpose programming.
- Python emphasizes its code readability by using **whitespace indentation** to delimit code blocks rather than **curly braces** or **keywords**).

Python: Introduction

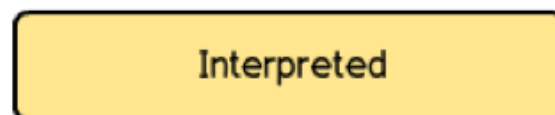
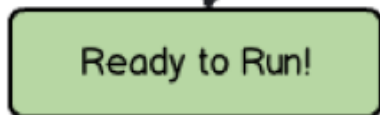
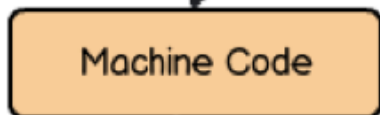
Compiled VS Interpreted Language



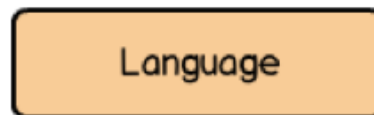
C, C++, Go, Fortran, Pascal



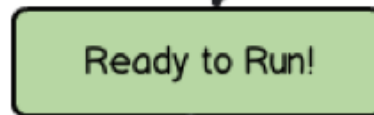
"Compiling"



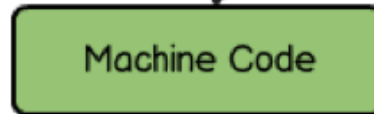
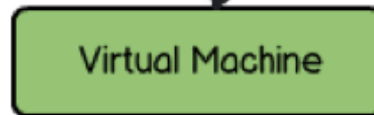
Python, PHP, Ruby, JavaScript



Ready to Run!



"Interpreting"



Python: Introduction

Compiled VS Interpreted Language

Compiled		Interpreted	
PROS	CONS	PROS	CONS
ready to run	not cross platform	cross-platform	interpreter required
often faster	inflexible	simpler to test	often slower
source code is private	extra step	easier to debug	source code is public

Python: Introduction

Code blocks

```
1 #!/usr/bin/python
2
3 print "Hello, World!";
4
```

"Hello, World!" program
in Python

```
1 #include <stdio.h>
2
3 int main()
4 {
5     printf("Hello, World! \n");
6     return 0;
7 }
8
```

"Hello, World!" program
in C

Python: Introduction

□ Comparison with other languages (1)

Matrix multiplication test: $C = AB$

Language	Option	n=1500	n=1750	n=2000
Python	intrinsic	0.49	0.80	0.95
Python + Numba (loops)		3.6	6.28	13.4
Matlab	intrinsic	0.77	1.02	0.99
Fortran	gfortran (matmul)	1.58	2.52	4.34
	gfortran -O3 (matmul)	1.28	2.05	3.68
	ifort (loop)	1.55	2.01	4.48
	ifort -O3 (loop)	0.51	0.81	1.24
	ifort -O3 (matmul)	0.52	0.82	1.25
	ifort (DGEMM)	0.19	0.23	0.33
C	gcc (loop)	13.33	21.18	31.77
	gcc -Ofast (loop)	1.34	2.35	4.30
	icc (loop)	1.25	2.19	3.99
	icc -Ofast (loop)	1.23	1.72	2.62

Source: <https://modelingguru.nasa.gov/docs/DOC-2625>

Python: Introduction

□ Comparison with other languages (2)

Solving 2-D Laplace's equation : $u_{xx} + u_{yy} = 0$

Language	Option	n=100	n=150	n=200
Python		144.54	715.96	2196.97
Python + Numba		1.23	5.37	16.34
Matlab		5.06	12.50	23.40
Fortran	gfortran	1.21	5.56	15.64
	gfortran -O3	0.668	3.072	8.897
	ifort	0.38	2.15	6.10
	ifort -O3	0.536	2.46	7.15
C	gcc	0.51	2.47	7.85
	gcc -Ofast	0.21	1.04	3.18
	icc	0.45	2.23	6.78
	icc -Ofast	0.32	1.60	4.87

Source: <https://modelingguru.nasa.gov/docs/DOC-2625>

Python: Basic

- Syntax**
- Data type**
- Operation**
- Control flows**
- Function**
- Sequence** (array)

Python: Basic

□ Syntax

Identifier (naming)

- An identifier starts with a letter A to Z or a to z or an underscore (_) followed by zero or more letters, underscores and digits (0 to 9).

Statement

- Semicolon (;) is not needed to end a statement. It can be used to omit display output and also for multiple statements on a single line.

```
x = 2.5; y = 4.5; z = 45  
d = x/y
```

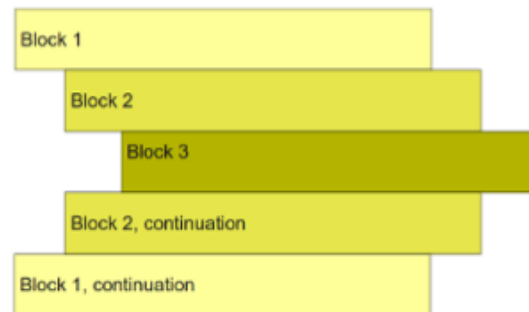
Python: Basic

□ Syntax

Lines and Indentation

- Blocks of code are denoted by line indentation
- The number of spaces in the indentation is variable, but all statements within the block must be indented the same amount.

```
if True:
    print "Answer"
    print "True"
else:
    print "Answer"
    print "False" XXXXXXXXX
```



This is very important!

Python: Basic

□ Syntax

Multi-Line Statements

Python allow multiline coding for both assignment and operation statement.

For example,

```
total = item_one + \  
        item_two + \  
        item_three
```

and,

```
days = ['Monday', 'Tuesday', 'Wednesday',  
        'Thursday', 'Friday']
```

Python: Basic

□ Syntax

Comment

'#' will be used for commenting. All characters after the # and up to the end of the physical line are part of the comment.

```
x = 2.5 # comments  
y = 4.5
```

Input and Output

Waiting for user input (keyboard)

```
input_str = raw_input("Enter input string:")
```

Display output

```
print( "Input = ",input_str,"\n") # Python 3
```

See: http://sebastianraschka.com/Articles/2014_python_2_3_key_diff.html

Today's Goals

Part I: Review, Q&A

Part II: Introduction to Python

Part III: Again! Exercises

Part IV: File I/O

Python: Exercise

□ Ex 1:

Giving a series $\mathbf{s} = \{-1, 4, -9, 16, -25, \dots\}$.

- 1) Can you find a formula of this series?
- 2) Write a Python program to sum the first 20 terms.
- 3) Write a program to find how many percent of the first 100 term that $|s| < 1000$.

Today's Goals

Part I: Review, Q&A

Part II: Introduction to Python

Part III: Again! Exercises

Part IV: File I/O

* I/O == input and output

File I/O: Python

How to simply WRITE and READ

First, create file object

```
file_object = open("filename", "mode")
```

File Mode

- '*r*' – Read mode which is used when the file is only being read
- '*w*' – Write mode which is used to edit and write new information to the file (any existing files with the same name will be erased when this mode is activated)
- '*a*' – Appending mode, which is used to add new data to the end of the file; that is new information is automatically amended to the end
- '*r+*' – Special read and write mode, which is used to handle both actions when working with a file

Try 1

```
file = open("testfile.txt", "w")
file.write("Hello World")
file.write("Second line.")
file.write("and the third line.")
file.close()
```

File I/O: Python

How to simply WRITE and READ

Try 2

```
file = open("testfile.txt","r")

# Try these commands
print(file.read())
print(file.read(5))
print(file.readline():)
print(file.readline(1):)

file.close()
```

File I/O: Python

How to simply WRITE and READ

Try 3

```
file = open("testfile.txt", "r")
for line in file:
    print(line)
```

Try 4

```
with open("testfile.txt") as f:
    for line in f:
        print(line)
```

File I/O: Python

How to simply WRITE and READ

Try 5: Splitting

```
with open("hello.text", "r") as f:  
    data = f.readlines()  
  
for line in data:  
    words = line.split()  
    print(words)
```

File I/O: Python

Try reading this tutorial:

1. http://www.python-course.eu/python3_file_management.php
2. http://www.python-course.eu/python3_formatted_output.php

Exercise 1: Try creating a simple file containing numbers in each line. Read those number into a list.

Exercise 2: Create a text file containing numbers in array format. Try reading it into a list.

Exercise 3: Score of 100 students is prepared in the course website. Try reading it into a list and do the following tasks.

- a) Find max, min, mean, median, mode and SD.
- b) Make a histogram inside a terminal and into a file.
- c) Write a file with grade after score in each line.

File I/O: Python

How to read numbers from file?

For a simple 1-D list, try using append.

For a 2-D array formatted file.

Way I:

```
file = open ('input.txt' , 'r')
arr = [ map(int,line.split(',')) for line in file ]
print(arr)
```

Way II:

```
arr = []
with open('input.txt', 'r') as file:
    for line in file:
        line = line.strip()
        if len(line) > 0:
            arr.append(map(int, line.split(',')))
print(arr)
```

Way III:

```
from numpy import loadtxt
lines = loadtxt("input.txt", delimiter="," , dtype="i")
```

File I/O: Python

How to write formatted string to file?

Try these tricks:

```
# Assume you had strings variable: filename, type, size and modified

f.write('%-40s %6s %10s %2s\n' % (filename, type, size, modified))

# or

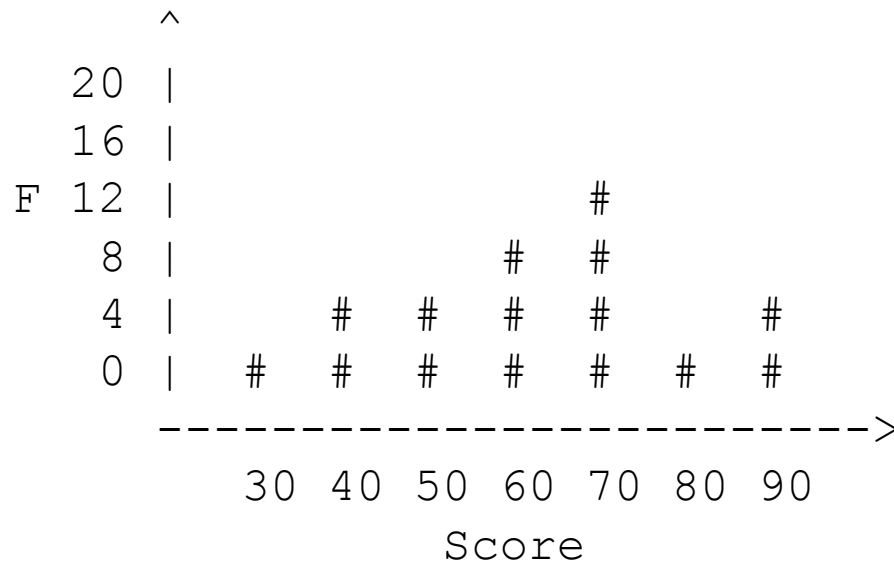
f.write(
    "{0} {1} {2} {3}".format(
        filename.ljust(40),
        type.rjust(6),
        size.rjust(10),
        modified.rjust(2)
    )
)
```


File I/O: Python

Exercise 3: Output example

Graph: histogram of student scores

=====



Average = xxxx

Mean = xxxx, SD = xxxx

Mode = xxxx, Median = xxxx

Python: Modules

Try reading the manual from <http://matplotlib.org/>

Exercise 1: Making a sine curve from 0 to 4π .

Exercise 2: Plot a histogram of a previous exercise.

Today's Goals

Part I: Review, Q&A

Part II: Introduction to Python

Part III: Again! Exercises

Part IV: File I/O

Something Interesting

Interesting Stuffs



NASA modeling Guru

> <https://modelingguru.nasa.gov>

File I/O: Python

Try reading this tutorial:

1. http://www.python-course.eu/python3_file_management.php
2. http://www.python-course.eu/python3_formatted_output.php

Exercise 1: Try creating a simple file containing numbers in each line. Read those number into a list.

Exercise 2: Score of 100 students is prepared in the course website. Try reading it into a list and do the following tasks.

- a) Find max, min, mean, median, mode and SD.
- b) Make a histogram inside a terminal!
- c) Write a file with grade after score in each line.