



PLAN 396

Lecture 8

Dr. Hossen Asiful Mustafa

<https://hossenmustafa.buet.ac.bd>



History of Python

- Created in 1989 by Guido van Rossum
 - Created as a scripting language for administrative tasks
 - Based on *All Basic Code (ABC)* and *Modula-3*
 - Added extensibility
 - Named after comic troupe Monty Python
- Released publicly in 1991
 - Growing community of Python developers
 - Evolved into well-supported programming language



History of Python

- Modules

- Reusable pieces of software
- Can be written by any Python developer
- Extend Python's capabilities

- Python Web site at www.python.org

- Primary distribution center for Python source code, modules and documentation



History of Python

○ Python

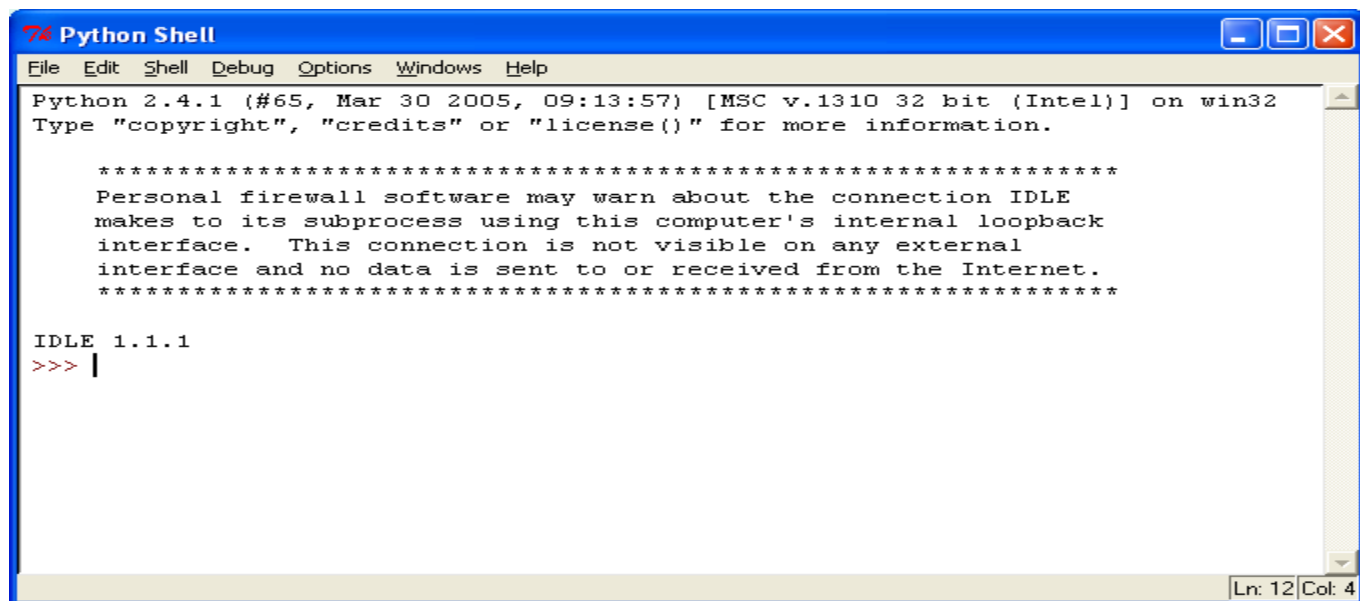
- Designed to be portable and extensible
 - Originally implemented on UNIX
 - Programs often can be ported from one operating system to another without any change
- Syntax and design promote good programming practices and surprisingly rapid development times
 - Simple enough to be used by beginning programmers
 - Powerful enough to attract professionals



Setting Up Python on Windows

- Go to <http://www.python.org> and get the latest distribution
 - Online tutorials
 - Python related websites
- Use all the defaults when installing
- Use online compiler
 - https://www.onlinegdb.com/online_python_compiler

Python IDLE



The screenshot shows a window titled "Python Shell" with a menu bar containing "File", "Edit", "Shell", "Debug", "Options", "Windows", and "Help". The main text area displays the following text:

```
Python 2.4.1 (#65, Mar 30 2005, 09:13:57) [MSC v.1310 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.

*****
Personal firewall software may warn about the connection IDLE
makes to its subprocess using this computer's internal loopback
interface.  This connection is not visible on any external
interface and no data is sent to or received from the Internet.
*****

IDLE 1.1.1
>>> |
```

The status bar at the bottom right indicates "Ln: 12|Col: 4".



Your First Python Program

- At the prompt (>>>) type:

```
print("Hello Python!")
```

- Press [Enter]

- A sequence of characters surrounded by “ “ or ‘ ‘

```
print('Hello Python!')
```



Your First Python Program

- Python is “case-sensitive”
 - `print("Hello Python!")`
 - `Print("Hello Python!")`
 - `PRINT("Hello Python!")`



Your First Python Program

- In Python, this computer instruction is called a “statement”
 - Command (like a verb) (print)
 - Expression (like a value) (“Hello Python”)
- More specifically, “Hello Python” is called a string expression
 - A series of characters between “ “



Syntax Errors

- When the computer does not recognize the statement to be executed, a syntax error is generated
- Analogous to a misspelled word in a programming language
 - Bug

```
>>> print("Hello Python!")  
SyntaxError: invalid syntax
```



Program Documentation

- Comment lines provide documentation about your program
 - Anything after the “#” symbol is a comment
 - Ignored by the computer

```
# Ima P Programmer
```

```
# First Python Program
```

```
# September 1, 2005
```



Programming in Script Mode

- Interactive mode gives you immediate feedback
- Not designed to create programs to save and run later
- Script Mode
 - Write, edit, save, and run (later)
 - Word processor for your code
- Save your file using the “.py” extension



Escape Sequences with Strings

- An **escape sequence** is a special sequence of characters that provide more functionality to the displayed text.
- `\\` Backslash, prints one backslash
- `\'` Single quote, prints one single quote
- `\"` Double quote, prints one double quote
- `\a` Bell, sounds the system bell
- `\b` Backspace, moves the cursor back one space
- `\n` Newline, moves the cursor to the beginning of next line
- `\t` Horizontal tab, Moves cursor forward one tab stop

Escape Sequences and Strings

- ASCII – American Standard Code for Information Interchange

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2	SP	!	"	#	\$	%	&	'	()	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL



Concatenating Strings

- String **concatenation** means to join two or more strings into a single string.
- The plus sign (+) is the string concatenation operator.
- Example

```
print('1'+"2"+"3")
```



Repeating Strings

- Strings can be repeated; multiple copies of the same string concatenated together
- The multiplication sign (*) is the string repetition operator
- Example:

```
print("1"*10)
```




Working With Numbers

- Two number types
 - Integers (e.g. -3, -2, 0, 100123)
 - Floating point (e.g. -23.2, 3.14159)
- Integers and floats act differently under certain circumstances (shown later)

Working With Numbers

<u>Python Operation</u>	<u>Arithmetic Operator</u>	<u>Algebraic Expression</u>	<u>Python Expression</u>	<u>Order of Evaluation</u>
Exponentiation	**	x^y	<code>x ** y</code>	Right -> left
Division	/	$x \div y$	<code>x / y</code>	Left -> right
Integer Division	// from __future__ import division	$x \div y$	<code>x // y</code>	Left -> right
Modulus	%	$x \bmod y$	<code>x % y</code>	Left -> right
Multiplication	*	xy	<code>x * y</code>	Left -> right
Addition	+	$x + y$	<code>x + y</code>	Left -> right
Subtraction	-	$x - y$	<code>x - y</code>	Left -> right



Working With Numbers

- Be careful of the order of evaluation!
 - Do what is in parentheses first!
 - ****** right to left
 - *****, **/**, **%** left to right
 - **+**, **-** left to right



Class Assignment

- Write a program named classassignment11.py
- The program should generate the following:

```
I LOVE PROGRAMMING!!!  
  *       *       *       *       *  
 *     *   *     *   *     *   *     *  
* * * * * * * * * * * * * * * *
```