

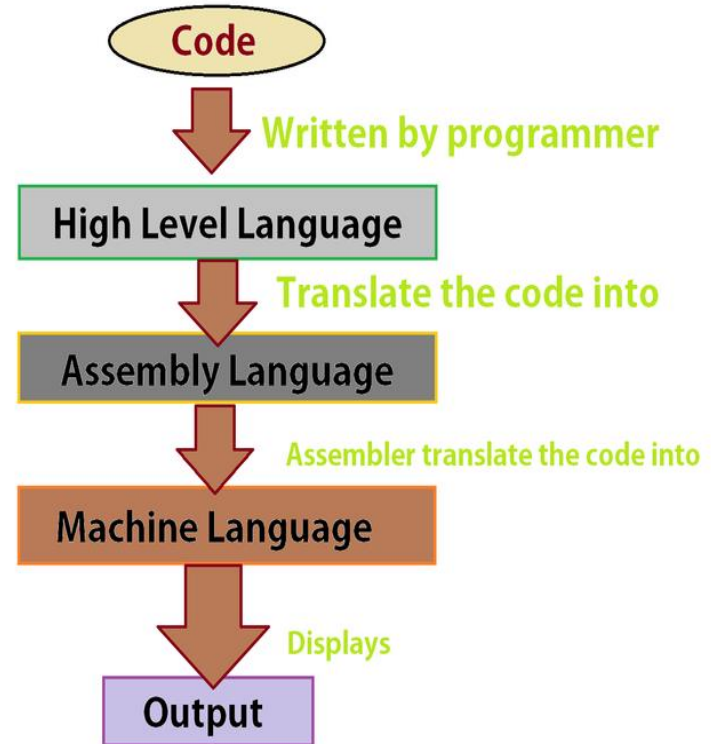
# Programming Fundamentals and Python

---

# Computer Programming

2

- The act of writing program which a computer can understand and take appropriate action, is known as computer programming.
- Generally, it is a common set of sequenced instructions which defines specific tasks.
- There are a large number of high level languages, some of them are BASIC, C, C++, Java, Python, PHP, Pascal, Perl, Ruby, FORTRAN, and Visual Basic etc.
- These all comes under high level programming, which is then translated into assembly language to machine language, to be implemented on hardware and fetch output.



- Python is an open-source (free) programming language that is used in web programming, data science, artificial intelligence, and many scientific applications.
- It is an interpreted, object oriented, high level programming language with dynamic semantics.
- It is very attractive for Rapid Application development.
- Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

---

## Programs With Python IDLE

- Python Integrated Development and Learning Environment allows you to enter, edit, save, retrieve, compile, link, and run a python program.

For Installation

<https://www.python.org/downloads/>

## Programs With Python IDLE

- Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).
- Python has a simple syntax similar to the English language.
- Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
- Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
- Python can be treated in a procedural way, an object-oriented way or a functional way.

- Python was designed for readability, and has some similarities to the English language with influence from mathematics.
- Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.
- Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.

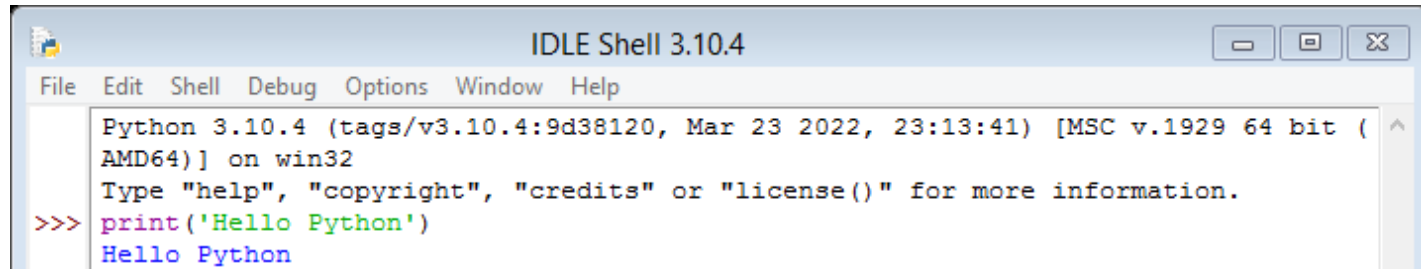
# Approach-1

7

## Hello Python

After installation of Python, follow following steps to develop and execute python program

- Create a python script file by replacing the extension of text file (.txt) with (.py).
- Right click on the file (say first.py) and select —Edit with IDLE.
- Type the program and click on —run > run module —or press F5 to execute the program. The window (on the next page) will appear showing the output of program.
- Although, simple programs can easily carried in 'IDLE shell', as shown below.



The screenshot shows the IDLE Shell 3.10.4 window. The title bar reads "IDLE Shell 3.10.4". The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main text area displays the following text:

```
Python 3.10.4 (tags/v3.10.4:9d38120, Mar 23 2022, 23:13:41) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print('Hello Python')
Hello Python
```

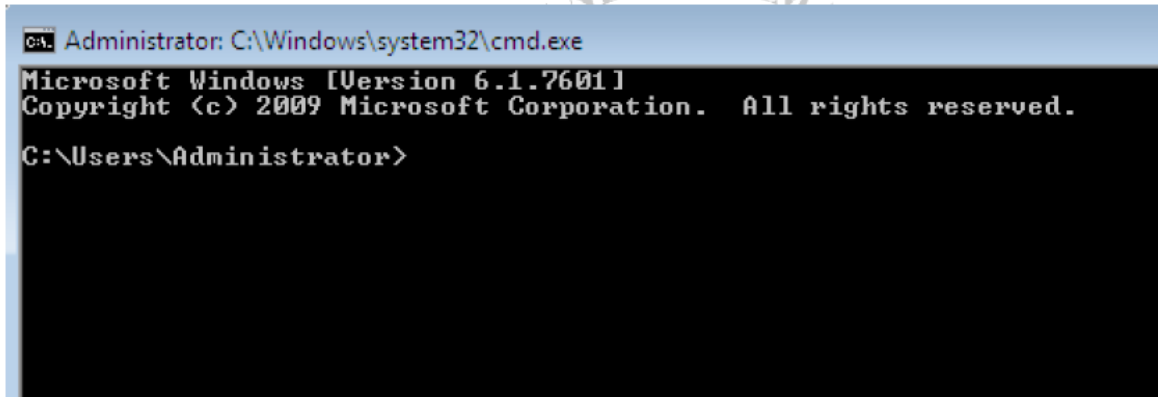
# Approach-2

8

## Hello world Program

After installation of Python, perform the following steps to develop and execute the python program.

- Create a python script file by replacing the extension of text file (.txt) with (.py).
- Open command prompt by clicking on command prompt from the start> All Programs > Accessories

A screenshot of a Windows Command Prompt window. The title bar at the top reads "Administrator: C:\Windows\system32\cmd.exe". The main text area shows the following: "Microsoft Windows [Version 6.1.7601]", "Copyright (c) 2009 Microsoft Corporation. All rights reserved.", and the current directory "C:\Users\Administrator>".

```
Administrator: C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.
C:\Users\Administrator>
```



- Change the path of DOS to the folder containing First.py with 'cd' command

```
C:\> Administrator: C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Administrator>cd..
C:\Users>cd..
C:\>cd Code
C:\code>
```

- Run the script by using command `python First.py`

```
C:\> Administrator: C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Administrator>cd..
C:\Users>cd..
C:\>cd Code
C:\code>python First.py
hello world
C:\code>
```

## RUN Menu

### 1. Python Shell

- Open or wake up the Python Shell window.

### 2. Check Module

- Check the syntax of the module currently open in the Editor window. If the module has not been saved, IDLE will either prompt the user to save or auto-save.

### 3. Run Module

- Do Check Module (above). If no error, restart the shell to clean the environment then execute the module. Output is displayed in the Shell window. Note that output requires use of `print` or `write`.

## Option Menu

### 1. Configure IDLE

- Open a configuration dialog and change preferences for the following: Fonts, indentation, key bindings, text color themes, startup windows and size, additional help sources, and extensions. To use a new built-in color theme (IDLE Dark) with older IDLEs, save it as a new custom theme.

### 2. Code Context (Editor Window only)

- Open a pane at the top of the edit window which shows the block context of the code which has scrolled above the top of the window.

- There are different notions, symbols and parenthesis style to assign different data type:

Example	Data Type
<code>x = "Hello World"</code>	str
<code>x = 20</code>	int
<code>x = 20.5</code>	float
<code>x = 1j</code>	complex
<code>x = ["apple", "banana", "cherry"]</code>	list
<code>x = ("apple", "banana", "cherry")</code>	tuple

<code>x = range(6)</code>	range
---------------------------	-------

<code>x = {"name" : "John", "age" : 36}</code>	dict
--	------

<code>x = {"apple", "banana", "cherry"}</code>	set
--	-----

<code>x = frozenset({"apple", "banana", "cherry"})</code>	frozenset
---	-----------

<code>x = True</code>	bool
-----------------------	------

<code>x = b"Hello"</code>	bytes
---------------------------	-------

<code>x = bytearray(5)</code>	bytearray
-------------------------------	-----------

<code>x = memoryview(bytes(5))</code>	memoryview
---------------------------------------	------------

<code>x = None</code>	NoneType
-----------------------	----------

# How to set a Data Type

14

- There are different notions, symbols and parenthesis style to assign different data type:

<code>x = str("Hello World")</code>	<code>str</code>
-------------------------------------	------------------

<code>x = int(20)</code>	<code>int</code>
--------------------------	------------------

<code>x = float(20.5)</code>	<code>float</code>
------------------------------	--------------------

<code>x = complex(1j)</code>	<code>complex</code>
------------------------------	----------------------

<code>x = list(("apple", "banana", "cherry"))</code>	<code>list</code>
--	-------------------

<code>x = tuple(("apple", "banana", "cherry"))</code>	<code>tuple</code>
---	--------------------

<code>x = range(6)</code>	<code>range</code>
---------------------------	--------------------

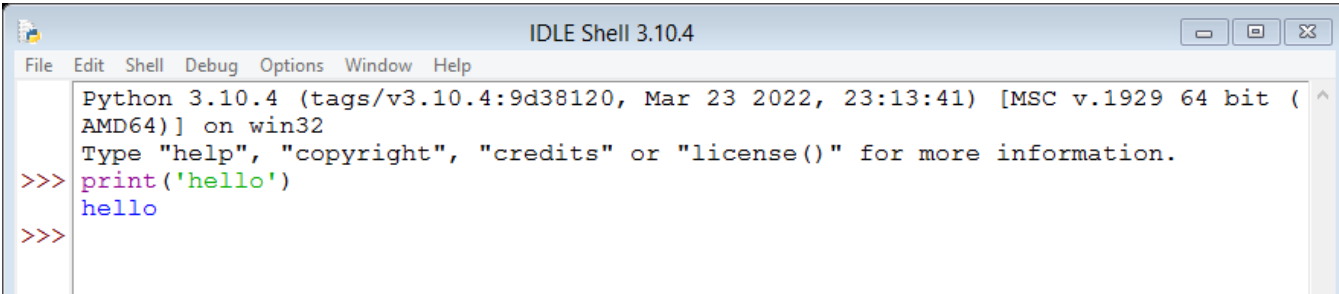
<code>x = dict(name="John", age=36)</code>	<code>dict</code>
--	-------------------

---

<code>x = set(("apple", "banana", "cherry"))</code>	set
<code>x = frozenset(("apple", "banana", "cherry"))</code>	frozenset
<code>x = bool(5)</code>	bool
<code>x = bytes(5)</code>	bytes
<code>x = bytearray(5)</code>	bytearray
<code>x = memoryview(bytes(5))</code>	memoryview

## Syntax

Python syntax can be executed directly in command line.

A screenshot of the IDLE Shell 3.10.4 window. The window has a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Window', and 'Help'. The main text area shows the Python 3.10.4 startup message: 'Python 3.10.4 (tags/v3.10.4:9d38120, Mar 23 2022, 23:13:41) [MSC v.1929 64 bit (AMD64)] on win32'. Below this, it says 'Type "help", "copyright", "credits" or "license()" for more information.' The prompt '>>>' is followed by the code 'print('hello')' and the output 'hello'. Another prompt '>>>' is shown on the next line.

```
Python 3.10.4 (tags/v3.10.4:9d38120, Mar 23 2022, 23:13:41) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print('hello')
hello
>>>
```



- Indentation refers to the spaces at the beginning of a code line.
- Where in other programming languages the indentation in code is for readability only, the indentation in Python is very important.
- Python uses indentation to indicate a block of code.
- Where in other programming languages the indentation in code is for readability only, the indentation in Python is very important.
- Python will show error if you miss the indent.

```
if 5 > 2:  
    print("Five is greater than two!")
```

Syntax Error:

```
if 5 > 2:  
print("Five is greater than two!")
```

- The number of spaces is up to you as a programmer, the most common use is four, but it has to be at least one.

```
if 5 > 2:  
    print("Five is greater than two!")  
if 5 > 2:  
    print("Five is greater than two!")
```

- You have to use the same number of spaces in the same block of code, otherwise Python will give you an error:

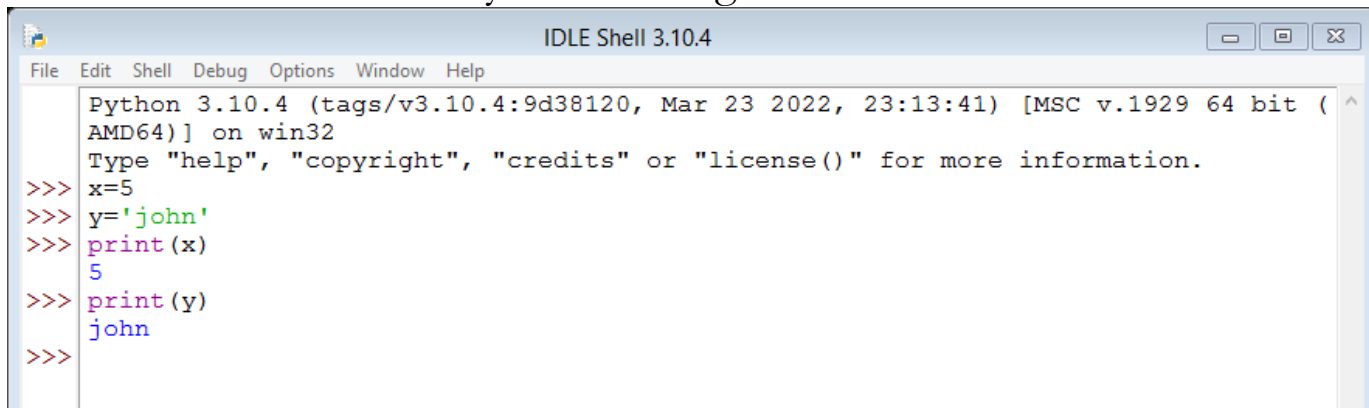
Syntax Error:

```
if 5 > 2:  
    print("Five is greater than two!")  
    print("Five is greater than two!")
```

# Python Variables

19

- Python has no command for declaring a variable.
- A variable is created the moment you first assign a value to it.

A screenshot of the IDLE Shell 3.10.4 window. The window has a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Window', and 'Help'. The main text area shows the Python 3.10.4 shell prompt and the following code and output:

```
Python 3.10.4 (tags/v3.10.4:9d38120, Mar 23 2022, 23:13:41) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> x=5
>>> y='john'
>>> print(x)
5
>>> print(y)
john
>>>
```

- Variables do not need to be declared with any particular type, and can even change type after they have been set.

```
x = 4          # x is of type int
x = "Sally"    # x is now of type str
print(x)
```

# Casting

20

- The method to declare any variable as any particular type, is called casting.
- The type can also be changed after it has been set.

```
x = str(3)    # x will be '3'  
y = int(3)    # y will be 3  
z = float(3)  # z will be 3.0
```

## Get The Type

- The type can be seen by using

```
>>> x=5  
>>> y='john'  
>>> print(x)  
5  
>>> print(y)  
john  
>>> print(type(x))  
<class 'int'>  
>>> print(type(y))  
<class 'str'>  
>>>
```

---

## Case Sensitive

- Python is case sensitive; you can declare a variable with 'x' and a different variable with 'X'.

## Quotes

- Single and double quotes are same.
- It is usually used to declare string variables.

There are several techniques you can use to make them more readable:

## Camel case

```
myVariableName = "Johnny"
```

## Pascal case

```
MyVariableName = "Johnny"
```

## Snake Case

```
my_variable_name = "Johnny"
```

## Assigning many values to multiple variable

- Python allows us to assign multiple variable in a line.
- Including, print option assess to print multiple variable.

```
>>> a,b,c='i','love','pakistan'  
>>> print(a,b,c)  
i love pakistan  
>>>
```

## One value to multiple variable

- One value can be assigned to multiple variables.

```
>>> d=e=f='world'  
>>> print(d)  
world  
>>> print(e)  
world  
>>> print(f)  
world  
>>>
```

# Collection Unpack

24

- If you have a collection of values (i.e. list or tuple), python allows to extract values into variables.

```
>>> fruits = ["apple", "banana", "cherry"]
>>> x, y, z = fruits
>>> print(x)
apple
>>> print(y)
banana
>>> print(z)
cherry
>>>
```

## Output Differences

- Different print command makes different output, see the example

```
>>> x='python'
>>> y='is'
>>> z='awesome'
>>> print(x+y+z)
pythonisawesome
>>> print(x,y,z)
python is awesome
>>> |
```



## Print () Function

- Print function allows you to add the same type of input.
- It shows error when we try to add or subtract different data type.

```
>>> a=5
>>> b=6
>>> print(a+b)
11
>>> d='Pakistan'
>>> e=13
>>> print(d+e)
Traceback (most recent call last):
  File "<pyshell#10>", line 1, in <module>
    print(d+e)
TypeError: can only concatenate str (not "int") to str
>>> print(d,e)
Pakistan 13
>>> |
```

- In computer programming language, data type is an important concept.
- Different data types can be stored in variable.
- Built-in data types in python are:

Text Type:	str
Numeric Types:	int, float, complex
Sequence Types:	list, tuple, range
Mapping Type:	dict
Set Types:	set, frozenset
Boolean Type:	bool
Binary Types:	bytes, bytearray, memoryview
None Type:	NoneType

Three different number type is present in python.

- Int – Int, or integer, is a whole number, positive or negative, without decimals, of unlimited length.
- Float – Float, or "floating point number" is a number, positive or negative, containing one or more decimals. Float can also be scientific numbers with an "e" to indicate the power of 10.
- Complex – Complex numbers are written with a "j" as the imaginary part.

Assign Numbers

```
x=2
y=7.9
z=complex(4,5)
print(x)
print(y)
print(z)|
print(type(x))
print(type(y))
print(type(z))
```

Output

```
==== RESTART: C:/Users/Hera Noor/AppData/Local/
2
7.9
(4+5j)
<class 'int'>
<class 'float'>
<class 'complex'>
>>> |
```

# Conversion of Types

29

- You can convert from one type to another with the `int()`, `float()`, and `complex()` methods.
- Considering the same example:

## Conversion

```
x=2
y=7.9
z=complex(4,5)
print(x)
print(y)
print(z)
a=float(x)
b=int(y)
c=complex(x)
print(type(a))
print(type(b))
print(type(c))
```

## Output

```
2
7.9
(4+5j)
<class 'float'>
<class 'int'>
<class 'complex'>
>>> |
```

- Python does not have a `random()` function to make a random number, but Python has a built-in module called `random` that can be used to make random numbers.
- You can import `random` module up to any number.

```
>>> import random
>>> print(random.randrange(3,7))
4
>>> print(random.randrange(2,9))
5
>>> print(random.randrange(13,100))
47
>>> print(random.randrange(1,100))
34
>>> print(random.randrange(1,100))
80
```

# Escape Sequences

---

# Python Escape Character

---

- An escape sequence is a sequence of characters that, when used inside a character or string, does not represent itself but is converted into another character or series of characters that may be difficult or impossible to express directly.
- In the escape sequence, a character is preceded by a backslash (\) followed by the character.
- We can use the Escape Sequences concept to specify actions such as tab movements in the terminal and also for the representation of non-printing characters like new line (\n) or characters that have special meaning like double quotation marks (").
- Talking about prevention, we have two methods: either use the double backslash method, which is not possible for a larger string or use the concept of raw string, i.e., adding r or R before the string.



- In order to insert characters that are illegal in a string, we use an escape character.
- An escape character is a backslash \ followed by the character you want to insert.
- An example of an illegal character is a double quote inside a string that is surrounded by double quotes.
- To fix this problem, use the escape character \":
- The escape character allows you to use double quotes when you normally would not be allowed:

```
>>> txt = "We are the so-called "Muslim" from the middle east."  
SyntaxError: invalid syntax  
>>> txt = "We are the so-called \"Muslims\" from the middle east."  
>>> txt  
'We are the so-called "Muslims" from the middle east.'  
>>> |
```

# Escape Characters

---

34

<u>Escape Sequence</u>	<u>Meaning</u>
\'	Single quote
\"	Double quote
\\	Backslash
\n	Newline
\r	Carriage Return
\t	Horizontal Tab
\b	Backspace
\f	Formfeed
\v	Vertical Tab
\0	Null Character

# Escape Characters

---

35

<code>\uxxxxxxxx</code>	Unicode character with a 16-bit hex value
<code>\Uxxxxxxx</code>	Unicode character with a 32-bit hex value
<code>\ooo</code>	Character with octal value ooo

# Examples

- New line
- Backslash
- Tab space between word
- Escape single quotes
- Escape sequence for hexa value
- Escape sequence for octal value

```
>>> print('daniel\nahmed')
daniel
ahmed
>>> print('daniel\\ahmed')
daniel\ahmed
>>> print('daniel\tahmed')
daniel  ahmed
>>> print('who\'s this')
who's this
```

```
>>> print("\x48\x45\x4C\x4C\x4F\x20\x57\x4F\x52\x4C\x44")
HELLO WORLD
>>> print("\110\105\114\114\117\040\127\117\122\114\104")
HELLO WORLD
```

- To ignore all the escape sequences in the string, we have to make a string as a raw string using 'r' before the string.

```
>>> print(r'daniel\nahmed')  
daniel\nahmed
```

- ASCII Character Chart with Decimal, Binary and Hexadecimal Conversions
- <https://www.eso.org/~ndelmott/ascii.html>

- 
- Task for you: install python (latest version)
  - Try all the examples from the lecture in python.
  - Apply remaining escape characters.
  - Check out different options in python IDLE and change the theme and color.

# Python – Basic Operations

---

# Number and Types

40

## STRING

- Any data type surrounded by quotation mark (either single or double quotation) is coined as string.

## MULTILINE STRING

- Multiline string need multi quotation.

## STRING LENGTH

- String length includes the alphabets and spaces as well.

## CHECK STRING

- You can check string if it's present or not by using keyword 'in'

```
>>> a='string'
>>> print(a)
string
>>> a='''I
        love
                Pakistan'''
>>> print(a)
I
        love
                Pakistan
>>> a='I Love Pakistan'
>>> print(len(a))
15
>>> print('love' in a)
False
>>> print('Love' in a)
True
>>> |
```



# String Operator

---

41

```
>>> s = 'a\nb\tc'
```

```
>>> s
```

```
'a\nb\tc'
```

```
>>> print(s)
```

```
a
```

```
b c
```

```
>>> S = 'Spam' # Make a 4-character  
string, and assign it to a name
```

```
>>> len(S) # Length
```

```
4
```

```
>>> S[0] # The first item in S, indexing  
by zero-based position  
      'S'
```