

CLASS : XII SCIENCE

SUBJECT : COMPUTER SCIENCE – PYTHON (083)

**BOOK: COMPUTER SCIENCE (SUMITA ARORA) DHANPATRAI
PUBLICATION.**

ONE TWO LINE REGISTER 200 PAGES.

NOTE:

***FOR NOW STUDENTS NEED NOT WORRY ABOUT THE TEXT BOOK OR
REGISTER.**

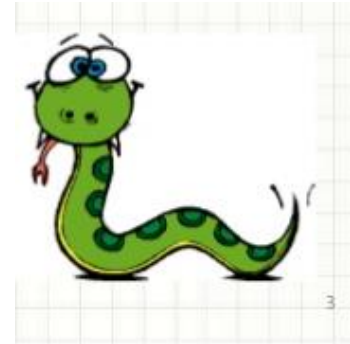


***KINDLY GO THROUGH THE NOTES**

*** THE NOTES SHOULD BE WRITTEN IN ANY ROUGH REGISTER.**

UNIT I – PROGRAMMING AND COMPUTATIONAL THINKING

CHAPTER1 REVIEW OF PYTHON BASICS



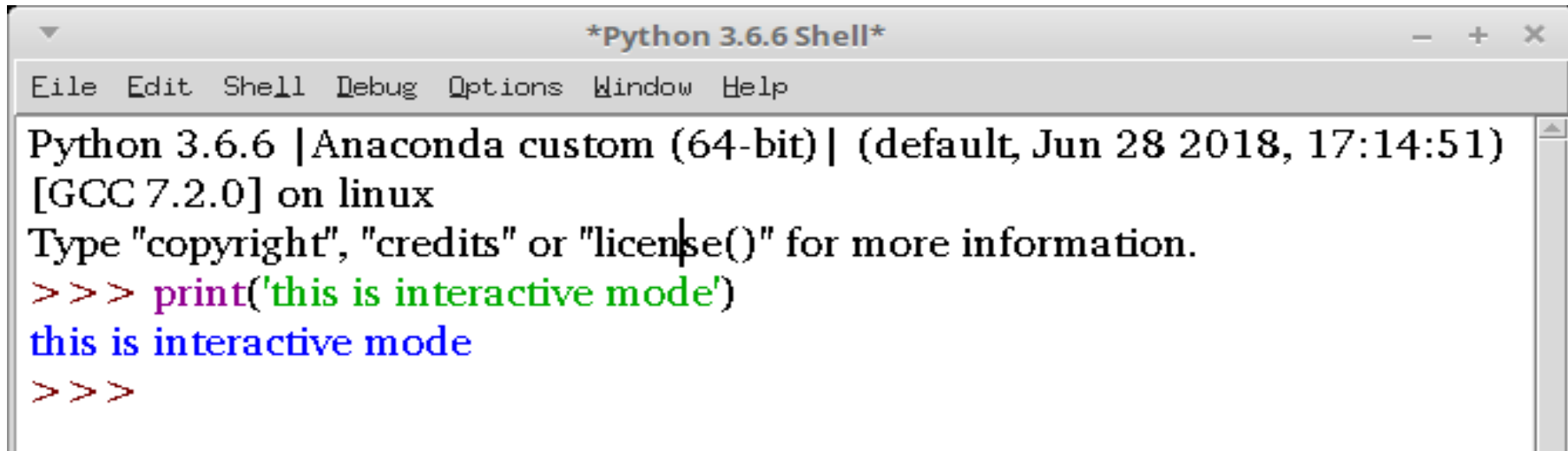
WHAT IS PYTHON ?

- **PYTHON IS PROGRAMMING LANGUAGE ALSO KNOWN AS SCRIPTING LANGUAGE ,DEVELOPED BY GUIDO VAN ROSSUM IN 1991.**
- **PHYTHON IS A HIGH LEVEL GENERAL PURPOSE, OBJECT ORIENTED PROGRAMMING LANGUAGE.**
- **PYTHON IS AN INTERPRETED LANGUAGE.**
- **PYTHON IS A CROSS PLATFORM, FREE, AND OPEN SOURCE LANGUAGE**
- **PYTHON HAS A VARIETY OF USAGE - SCRIPTING, WEB APPLICATION, , GUI PROGRAMS , DATABASE APPLICATIONS ETC.**

PYTHON IDLE (INTEGRATED DEVELOPMENT AND LEARNING ENVIRONMENT)

PYTHON IDLE PROVIDES TWO MODES OF WORKING –

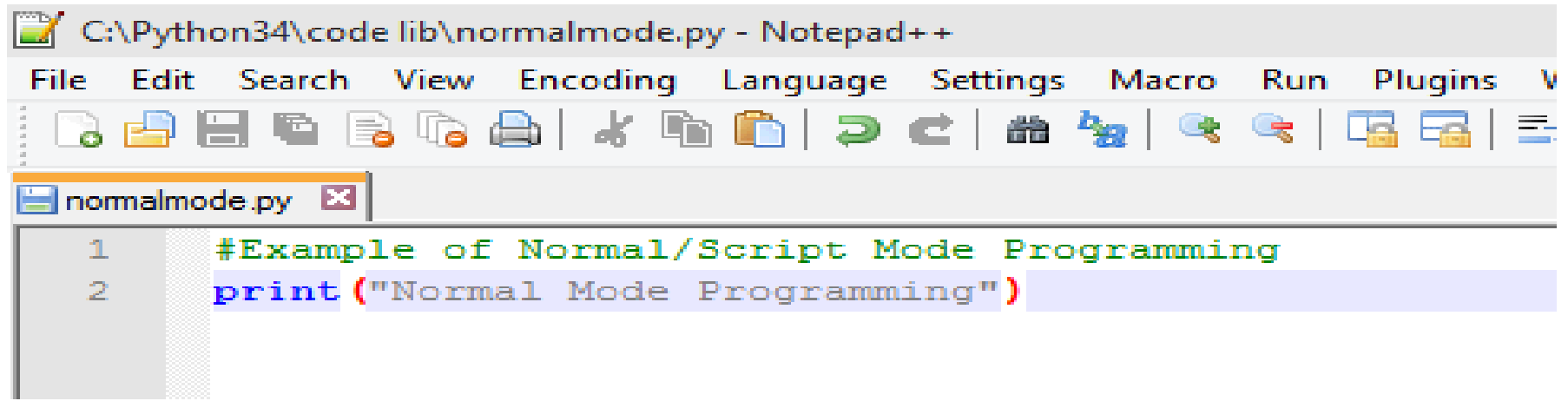
1. INTERACTIVE MODE (PYTHON SHELL)



The screenshot shows a window titled '*Python 3.6.6 Shell*' with a menu bar (File, Edit, Shell, Debug, Options, Window, Help). The main text area displays the following content:

```
Python 3.6.6 |Anaconda custom (64-bit)| (default, Jun 28 2018, 17:14:51)
[GCC 7.2.0] on linux
Type "copyright", "credits" or "license()" for more information.
>>> print('this is interactive mode')
this is interactive mode
>>>
```

2. SCRIPT MODE

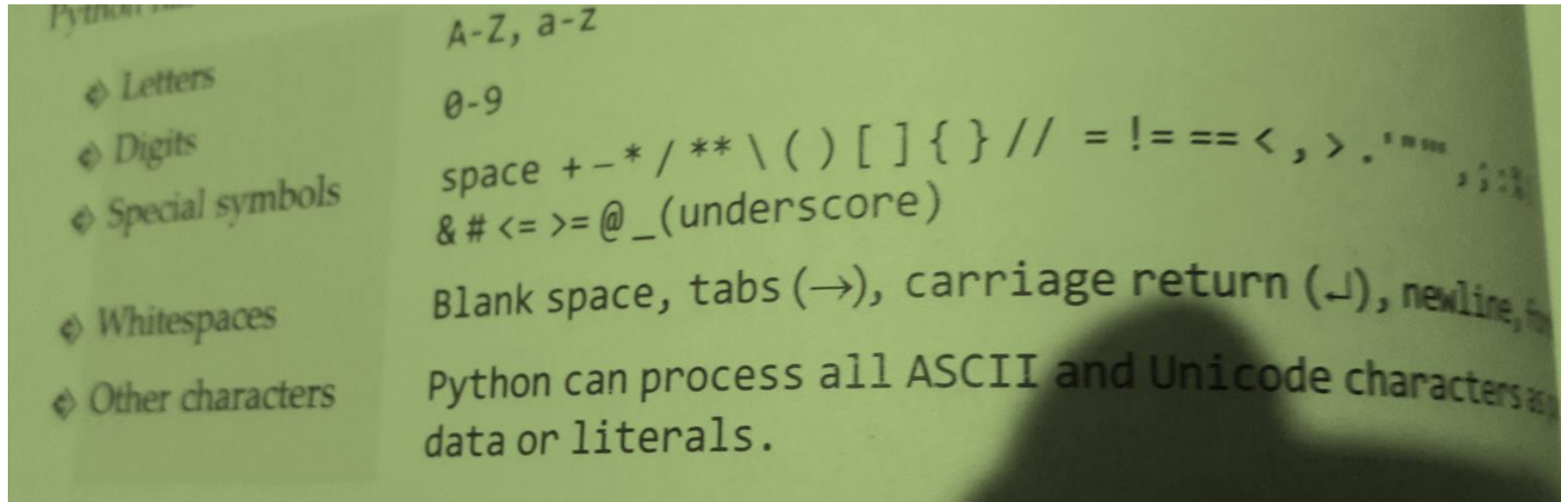


The screenshot shows a Notepad++ window titled 'C:\Python34\code lib\normalmode.py - Notepad++'. The menu bar includes File, Edit, Search, View, Encoding, Language, Settings, Macro, Run, and Plugins. The toolbar contains various icons for file operations and editing. The main text area shows the following code:

```
1 #Example of Normal/Script Mode Programming
2 print("Normal Mode Programming")
```

FUNDAMENTALS OF PYTHON

CHARACTER SET



TOKENS IN PYTHON

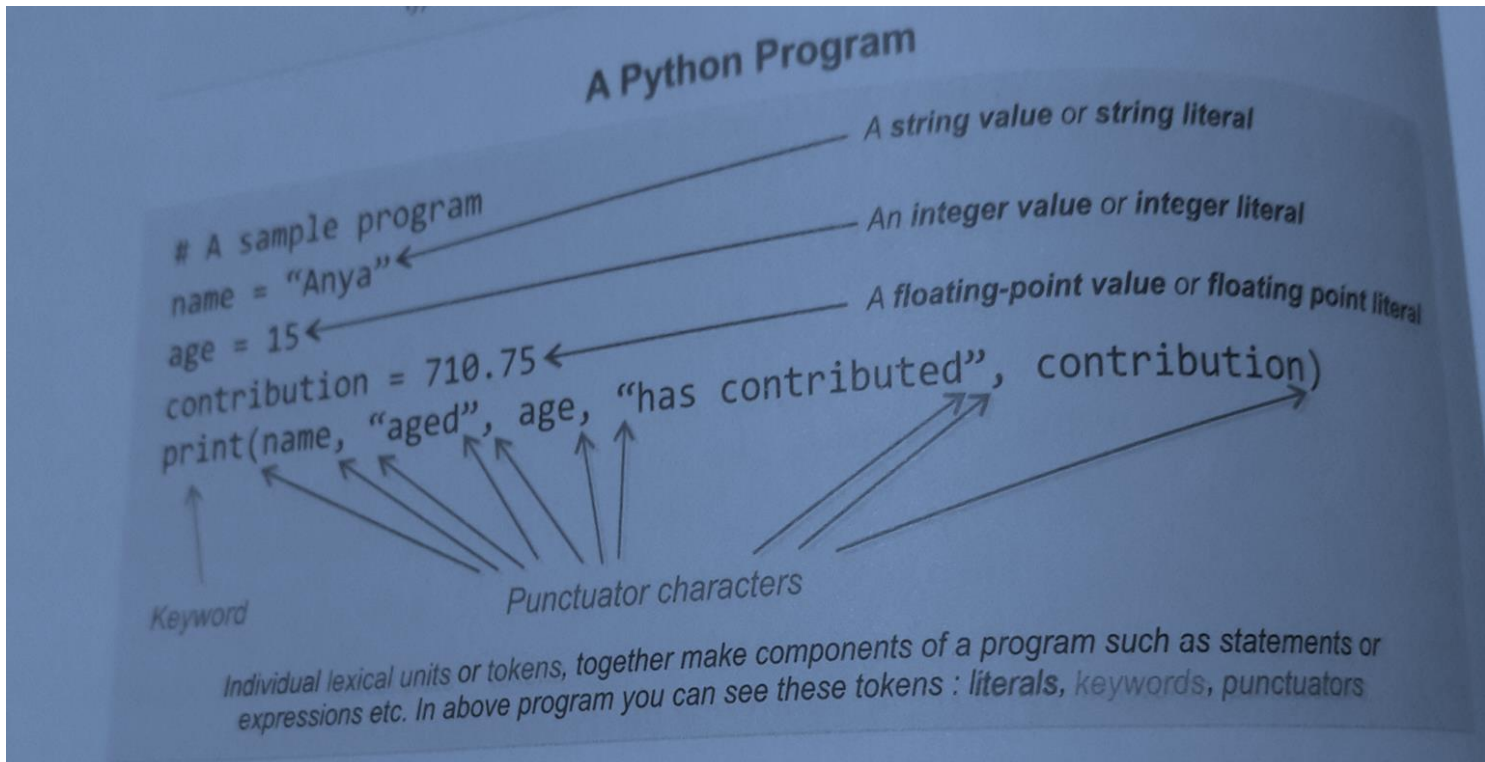


Figure 6.1 Tokens are smallest individual units in a program.

Python has following tokens :

- (i) Keywords
- (ii) Identifiers (Names)
- (iii) Literals
- (iv) Operators
- (v) Punctuators

TOKENS

The smallest indiv
program is know

OPERATORS IN PYTHON

1. ARITHMETIC OPERATORS:

Arithmetic operators are used to perform mathematical operations like addition subtraction, multiplication and division.

THE DIFFERENT ARITHMETIC OPERATORS ARE :-

OPERATOR	DESCRIPTION	SYNTAX
+	Addition: adds two operands	$x + y$
-	Subtraction: subtracts two operands	$x - y$
*	Multiplication: multiplies two operands	$x * y$
/	Division (float): divides the first operand by the second	x / y
//	Division (floor): divides the first operand by the second	$x // y$
%	Modulus: returns the remainder when first operand is divided by the second	$x \% y$

EXAMPLE OF ARITHMETIC OPERATORS

```
# Examples of Arithmetic Operator
a = 9
b = 4

# Addition of numbers
add = a + b
# Subtraction of numbers
sub = a - b
# Multiplication of number
mul = a * b
# Division(float) of number
div1 = a / b
# Division(floor) of number
div2 = a // b
# Modulo of both number
mod = a % b

# print results
print(add)
print(sub)
print(mul)
print(div1)
print(div2)
print(mod)
```

Output:

13

5

36

2.25

2

1

2. ASSIGNMENT OPERATORS IN PYTHON

Assignment operators are used to assign values to variables:

Operator	Example	Same As
<code>=</code>	<code>x = 5</code>	<code>x = 5</code>
<code>+=</code>	<code>x += 3</code>	<code>x = x + 3</code>
<code>-=</code>	<code>x -= 3</code>	<code>x = x - 3</code>
<code>*=</code>	<code>x *= 3</code>	<code>x = x * 3</code>
<code>/=</code>	<code>x /= 3</code>	<code>x = x / 3</code>
<code>%=</code>	<code>x %= 3</code>	<code>x = x % 3</code>
<code>//=</code>	<code>x //= 3</code>	<code>x = x // 3</code>
<code>**=</code>	<code>x **= 3</code>	<code>x = x ** 3</code>

3. RELATIONAL OR COMPARISON OPERATORS IN PYTHON

Operator	Name	Example
==	Equal	x == y
!=	Not equal	x != y
>	Greater than	x > y
<	Less than	x < y
>=	Greater than or equal to	x >= y
<=	Less than or equal to	x <= y

4. LOGICAL OPERATORS IN PYTHON

Logical operators are used to combine conditional statements:

Operator	Description	Example
and	Returns True if both statements are true	x < 5 and x < 10
or	Returns True if one of the statements is true	x < 5 or x < 4
not	Reverse the result, returns False if the result is true	not (A > B) IF A = 5, B = 7 not (F) T

5. PYTHON IDENTITY OPERATORS

IDENTITY OPERATORS ARE USED TO COMPARE THE OBJECTS, NOT IF THEY ARE EQUAL BUT IF THEY ARE ACTUALLY THE SAME OBJECT, WITH THE SAME MEMORY LOCATION:

Operator	Description	Example
is	Returns True if both variables are the same object	x is y
is not	Returns True if both variables are not the same	x is not y

6. PYTHON MEMBERSHIP OPERATORS

MEMBERSHIP OPERATORS ARE USED TO TEST IF A SEQUENCE IS PRESENTED IN AN OBJECT:

Operator	Description	Example
in	Returns True if a sequence with the specified value is present in the object	x in y
not in	Returns True if a sequence with the specified value is not present in the object	x not in y

7. Python Bitwise Operators

Bitwise operators are used to compare (binary) numbers.

OPERATOR	NAME	DESCRIPTION
&	AND	Sets each bit to 1 if both bits are 1
 	OR	Sets each bit to 1 if one of two bits is 1
^	XOR	Sets each bit to 1 if only one of two bits is 1
~	NOT	Inverts all the bits

CONTROL STATEMENTS IN PYTHON

Control statements are used to control the flow of execution depending upon the specified condition/logic. There are three types of control statements

1. Decision Making Statements (if, elif, else)
2. Iteration Statements (while and for Loops)
3. Jump Statements (break, continue, pass)

Decision Making Statements (if, elif, else) Syntax: if(logic): Statement/s elif(logic): Statement/s else: Statement/s	Program a=int(input("Enter any integer number :")) if(a==0): print("Number is Zero") elif(a>0): print("Number is +ve") else: print("Number is -ve")	Output Enter any integer number :5 Number is Positive
Iteration Statements (while loop) Syntax: while(condition): Statement/s	Program n=1 while(n<4): print("Govind ", end=" ") n=n+1	Output Govind Govind Govind
Iteration Statements (for loop) Syntax: for value in sequence: Statement/s	Program for i in range(1, 6): print(i, end=' ')	Output 1 2 3 4 5
Jump Statements (break, continue, pass) Syntax for val in sequence: If (val== i): break if (val== j): continue if (val== k): pass	Program for i in range(1, 11): if(i==3): print("hello", end=' ') continue if(i==8): break if(i==5): pass else: print(i, end=' ');	Output 1 2 hello 4 6 7

LISTS IN PYTHON

In Python programming, a list is created by placing all the items (elements) inside a square bracket [], separated by commas.

It can have any number of items and they may be of different types (integer, float, string etc.).

```
# empty list          mylist = []  
# list of integers    mylist = [1, 2, 3]  
# list with mixed datatypes mylist = [1, "Hello", 3.4]
```

Also, a list can even have another list as an item. This is called nested list.

```
# nested list        mylist = ['mouse', [8, 4, 6], ['a']]
```

TUPLE IN PYTHON

It is a sequence of immutable objects. It is just like a list.

Difference between a tuple and a list

- tuple cannot be changed like a list.
- List uses square bracket whereas tuple use parentheses.
- L=[1,2,3,4,5] Mutable Elements of list can be changed
- T=(1,2,3,4,5) Immutable Elements of tuple can not be changed

DICTIONARY IN PYTHON

Dictionary in Python is an unordered collection of data values, used to store data values along with the keys.

Dictionary holds key: value pair. Key value is provided in the dictionary to make it more optimized. Each key-value pair in a Dictionary is separated by a colon:, whereas each key is separated by a ‘comma’.

```
dict={ "a": "alpha", "o": "omega", "g": "gamma" }
```

STRINGS IN PYTHON

A string in Python is a sequence of characters. It is a derived data type.

Strings are immutable. This means that once defined, they cannot be changed. Many Python methods, such as `replace()`, `join()`, or `split()` modify strings. They do not modify the original string. They create a copy of a string which they modify and return to the caller.

Python strings can be created with single quotes, double quotes, or triple quotes. When we use triple quotes, strings can span several lines without using the escape character.

```
# string_literals.py
```

```
a = "proximity alert"
```

```
b = 'evacuation'
```

```
c = """ requiem
```

```
for a tower """
```

```
print(a)
```

```
print(b)
```

```
print(c)
```

In our example we assign three string literals to `a`, `b`, and `c` variables. And we print them to the console.

SOME STRING FUNCTIONS-

Syntax	Description
<code>isalnum()</code>	Returns True if the string contains only letters and digit. It returns False. If the string contains any special character like <code>_</code> , <code>@</code> , <code>#</code> , <code>*</code> , etc.
<code>isalpha()</code>	Returns True if the string contains only letters. Otherwise return False.
<code>isdigit()</code>	Returns True if the string contains only numbers. Otherwise it returns False.
<code>lower()</code>	Returns the exact copy of the string with all the letters in lowercase.
<code>islower()</code>	Returns True if the string is in lowercase.
<code>isupper()</code>	Returns True if the string is in uppercase.
<code>upper()</code>	Returns the exact copy of the string with all letters in uppercase.
<code>title()</code>	Returns a string in title case
<code>swapcase()</code>	It will change case of every character to its opposite case vice-versa.