LANGUAGE FUNDAMENTALS

**1 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Introduction**

• **Python is a general purpose high level programming language.**

• **Python was developed by Guido Van Rossam in 1989 while working at National Research Institute at Netherlands.**

• **But officially Python was made available to public in 1991. The official Date of Birth for Python is : Feb 20th 1991.**

• **Python is recommended as first programming language for beginners.**

**Eg1: To print Helloworld**

**Java: 1) public class HelloWorld**

**2) { 3) p s v main(String[] args) 4) { 5) SOP("Hello world"); 6) } 7) }**

**C:**

**1) #include<stdio.h> 2) void main() 3) { 4) print("Hello world"); 5) }**

**Python: print("Hello World")**

**Eg2: To print the sum of 2 numbers**

**Java: 1) public class Add**

**2) { 3) public static void main(String[] args) 4) { 5) int a,b;**

**2 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**6) a =10; 7) b=20; 8) System.out.println("The Sum:"+(a+b)); 9) } 10) }**

**C:**

**1) #include <stdio.h> 2) 3) void main() 4) { 5) int a,b; 6) a =10; 7) b=20; 8) printf("The Sum:%d",(a+b)); 9) }**

**Python:**

**1) a=10 2) b=20 3) print("The Sum:",(a+b))**

**The name Python was selected from the TV Show "The Complete Monty Python's Circus", which was broadcasted in BBC from 1969 to 1974.**

**Guido developed Python language by taking almost all programming features from different languages**

**1. Functional Programming Features from C 2. Object Oriented Programming Features from C++ 3. Scripting Language Features from Perl and Shell Script 4. Modular Programming Features from Modula-3**

**Most of syntax in Python Derived from C and ABC languages.**

**Where we can use Python:**

**We can use everywhere. The most common important application areas are**

**1) For developing Desktop Applications 2) For developing web Applications 3) For developing database Applications 4) For Network Programming**

**3 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**5) For developing games 6) For Data Analysis Applications 7) For Machine Learning 8) For developing Artificial Intelligence Applications 9) For IoT**

**... Note:**

• **Internally Google and Youtube use Python coding.**

• **NASA and Nework Stock Exchange Applications developed by Python.**

• **Top Software companies like Google, Microsoft, IBM, Yahoo using Python.**

**Features of Python:**

**1) Simple and easy to learn:**

• **Python is a simple programming language. When we read Python program,we can feel like reading english statements.**

• **The syntaxes are very simple and only 30+ kerywords are available.**

• **When compared with other languages, we can write programs with very less number of lines. Hence more readability and simplicity.**

• **We can reduce development and cost of the project.**

**2) Freeware and Open Source:**

• **We can use Python software without any licence and it is freeware.**

• **Its source code is open,so that we can we can customize based on our requirement.**

• **Eg: Jython is customized version of Python to work with Java Applications.**

**3) High Level Programming language:**

• **Python is high level programming language and hence it is programmer friendly language.**

• **Being a programmer we are not required to concentrate low level activities like memory management and security etc.**

**4) Platform Independent:**

• **Once we write a Python program,it can run on any platform without rewriting once again.**

• **Internally PVM is responsible to convert into machine understandable form.**

**5) Portability:**

**Python programs are portable. ie we can migrate from one platform to another platform very easily. Python programs will provide same results on any paltform.**

**4 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**6) Dynamically Typed:**

• **In Python we are not required to declare type for variables. Whenever we are assigning the value, based on value, type will be allocated automatically.Hence Python is considered as dynamically typed language.**

• **But Java, C etc are Statically Typed Languages b'z we have to provide type at the beginning only.**

• **This dynamic typing nature will provide more flexibility to the programmer.**

**7) Both Procedure Oriented and Object Oriented:**

**Python language supports both Procedure oriented (like C, pascal etc) and object oriented (like C++, Java) features. Hence we can get benefits of both like security and reusability etc**

**8) Interpreted:**

• **We are not required to compile Python programs explcitly. Internally Python interpreter will take care that compilation.**

• **If compilation fails interpreter raised syntax errors. Once compilation success then PVM (Python Virtual Machine) is responsible to execute.**

**9) Extensible:**

• **We can use other language programs in Python.**

• **The main advantages of this approach are:**

▪ **We can use already existing legacy non-Python code**

▪ **We can improve performance of the application**

**10) Embedded:**

**We can use Python programs in any other language programs. i.e we can embedd Python programs anywhere.**

**11) Extensive Library:**

• **Python has a rich inbuilt library.**

• **Being a programmer we can use this library directly and we are not responsible to implement the functionality. Etc.**

**Limitations of Python: 1) Performance wise not up to the mark because it is interpreted language. 2) Not using for mobile Applications.**

**5 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Flavors of Python:**

**1) CPython:**

**It is the standard flavor of Python. It can be used to work with C lanugage Applications.**

**2) Jython OR JPython:**

**It is for Java Applications. It can run on JVM**

**3) IronPython:**

**It is for C#.Net platform**

**4) PyPy:**

**The main advantage of PyPy is performance will be improved because JIT compiler is available inside PVM.**

**5) RubyPython**

**For Ruby Platforms**

**6) AnacondaPython**

**It is specially designed for handling large volume of data processing.**

**Python Versions:**

➢ **Python 1.0V introduced in Jan 1994** ➢ **Python 2.0V introduced in October 2000** ➢ **Python 3.0V introduced in December 2008**

**Note: Python 3 won't provide backward compatibility to Python2 i.e there is no guarantee that Python2 programs will run in Python3.**

**Current versions**

**Python 3.6.1 Python 2.7.13**

**6 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

IDENTIFIERS

• **A Name in Python Program is called Identifier.**

• **It can be Class Name OR Function Name OR Module Name OR Variable Name.**

• **a = 10**

**Rules to define Identifiers in Python:**

**1. The only allowed characters in Python are**

• **alphabet symbols(either lower case or upper case)**

• **digits(0 to 9)**

• **underscore symbol(\_)**

**By mistake if we are using any other symbol like $ then we will get syntax error.**

• **cash = 10** √

• **ca$h =20** X

**2. Identifier should not starts with digit**

• **123total** X

• **total123** √

**3. Identifiers are case sensitive. Of course Python language is case sensitive language.**

• **total=10**

• **TOTAL=999**

• **print(total) #10**

• **print(TOTAL) #999**

**7 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Identifier:**

**1) Alphabet Symbols (Either Upper case OR Lower case)**

**2) If Identifier is start with Underscore (\_) then it indicates it is private.**

**3) Identifier should not start with Digits.**

**4) Identifiers are case sensitive.**

**5) We cannot use reserved words as identifiers**

**Eg: def = 10** X

**6) There is no length limit for Python identifiers. But not recommended to use too**

**lengthy identifiers.**

**7) Dollor ($) Symbol is not allowed in Python.**

**Q) Which of the following are valid Python identifiers?**

**1) 123total** X **2) total123** √ **3) java2share** √ **4) ca$h** X **5) \_abc\_abc\_** √ **6) def** X **7) if** X

**Note:**

**1) If identifier starts with \_ symbol then it indicates that it is private 2) If identifier starts with \_\_(Two Under Score Symbols) indicating that strongly private**

**identifier. 3) If the identifier starts and ends with two underscore symbols then the identifier is**

**language defined special name, which is also known as magic methods.**

**4) Eg: \_\_add\_\_**

**8 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

RESERVED WORDS

**In Python some words are reserved to represent some meaning or functionality. Such types of words are called reserved words.**

**There are 33 reserved words available in Python.**

• **True, False, None**

• **and, or ,not,is**

• **if, elif, else**

• **while, for, break, continue, return, in, yield**

• **try, except, finally, raise, assert**

• **import, from, as, class, def, pass, global, nonlocal, lambda, del, with**

**Note:**

**1. All Reserved words in Python contain only alphabet symbols.**

**2. Except the following 3 reserved words, all contain only lower case alphabet symbols.**

• **True**

• **False**

• **None**

**Eg: a= true** X **a=True** √

**>>> import keyword >>> keyword.kwlist ['False', 'None', 'True', 'and', 'as', 'assert', 'break', 'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yield']**

**9 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

DATA TYPES

• **Data Type represents the type of data present inside a variable.**

• **In Python we are not required to specify the type explicitly. Based on value provided, the type will be assigned automatically.Hence Python is dynamically Typed Language.**

**Python contains the following inbuilt data types**

**1) Int 2) Float 3) Complex 4) Bool 5) Str 6) Bytes 7) Bytearray 8) Range 9) List 10) Tuple 11) Set 12) Frozenset 13) Dict 14) None**

**Note: Python contains several inbuilt functions**

**1) type()**

**to check the type of variable**

**2) id()**

**to get address of object**

**10 a = 10 a = 20**

**a 20**

**a = 10 a b = 10**

**10 b**

**10 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**3) print()**

**to print the value**

**In Python everything is an Object.**

**1) int Data Type:**

**We can use int data type to represent whole numbers (integral values) Eg: a = 10**

**type(a) #int**

**Note:**

• **In Python2 we have long data type to represent very large integral values.**

• **But in Python3 there is no long type explicitly and we can represent long values also by using int type only.**

**We can represent int values in the following ways**

**1) Decimal form 2) Binary form 3) Octal form 4) Hexa decimal form**

**I) Decimal Form (Base-10):**

• **It is the default number system in Python**

• **The allowed digits are: 0 to 9**

• **Eg: a =10**

**II) Binary Form (Base-2):**

• **The allowed digits are : 0 & 1**

• **Literal value should be prefixed with 0b or 0B**

• **Eg: a = 0B1111**

**a = 0B123 a = b111**

**11 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**III) Octal Form (Base-8):**

• **The allowed digits are : 0 to 7**

• **Literal value should be prefixed with 0o or 0O.**

• **Eg: a = 0o123 a = 0o786**

**IV) Hexa Decimal Form (Base-16):**

• **The allowed digits are: 0 to 9, a-f (both lower and upper cases are allowed)**

• **Literal value should be prefixed with 0x or 0X**

• **Eg: a = 0XFACE**

**a = 0XBeef a = 0XBeer**

**Note: Being a programmer we can specify literal values in decimal, binary, octal and hexa decimal forms. But PVM will always provide values only in decimal form.**

▪ **a=10**

▪ **b=0o10**

▪ **c=0X10**

▪ **d=0B10**

▪ **print(a)10**

▪ **print(b)8**

▪ **print(c)16**

▪ **print(d)2**

**Base Conversions**

**Python provide the following in-built functions for base conversions**

**1) bin():**

**We can use bin() to convert from any base to binary**

**1) >>> bin(15) 2) '0b1111' 3) >>> bin(0o11) 4) '0b1001' 5) >>> bin(0X10) 6) '0b10000'**

**12 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**2) oct():**

**We can use oct() to convert from any base to octal**

**1) >>> oct(10) 2) '0o12' 3) >>> oct(0B1111) 4) '0o17' 5) >>> oct(0X123)** 6) **'0o443'**

**3) hex():**

**We can use hex() to convert from any base to hexa decimal**

**1) >>> hex(100) 2) '0x64' 3) >>> hex(0B111111) 4) '0x3f' 5) >>> hex(0o12345)** 6) **'0x14e5'**

**2) Float Data Type:**

• **We can use float data type to represent floating point values (decimal values) Eg: f = 1.234**

**type(f) float**

• **We can also represent floating point values by using exponential form (Scientific Notation) Eg: f = 1.2e3** → **instead of 'e' we can use 'E'**

**print(f) 1200.0**

• **The main advantage of exponential form is we can represent big values in less memory.**

**\*\*\*Note:**

**We can represent int values in decimal, binary, octal and hexa decimal forms. But we can represent float values only by using decimal form.**

**13 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**1) >>> f=0B11.01 2) File "<stdin>", line 1 3) f=0B11.01 4) ^ 5) SyntaxError: invalid syntax 6) 7) >>> f=0o123.456 8) SyntaxError: invalid syntax 9) 10) >>> f=0X123.456 11) SyntaxError: invalid syntax**

**3) Complex Data Type:**

• **A complex number is of the form**

• **‘a’ and ‘b’ contain Intergers OR Floating Point Values.**

**Eg: 3 + 5j**

**10 + 5.5j 0.5 + 0.1j**

• **In the real part if we use int value then we can specify that either by decimal, octal, binary or hexa decimal form.**

• **But imaginary part should be specified only by using decimal form.**

**1) >>> a=0B11+5j 2) >>> a 3) (3+5j) 4) >>> a=3+0B11j 5) SyntaxError: invalid syntax**

• **Even we can perform operations on complex type values.**

**1) >>> a=10+1.5j 2) >>> b=20+2.5j 3) >>> c=a+b 4) >>> print(c) 5) (30+4j)**

**a + bj**

**j2 j =**

**= -1**

**Real Part Imaginary Part**

**14 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**6) >>> type(c) 7) <class 'complex'>**

**Note: Complex data type has some inbuilt attributes to retrieve the real part and imaginary part**

**c = 10.5+3.6j**

**c.real** → **10.5 c.imag** → **3.6**

**We can use complex type generally in scientific Applications and electrical engineering Applications.**

**4) bool Data Type:**

• **We can use this data type to represent boolean values.**

• **The only allowed values for this data type are:**

• **True and False**

• **Internally Python represents True as 1 and False as 0**

**b = True type(b)** →**bool**

**Eg: a = 10 b = 20 c = a<b print(c)** → **True**

**True+True** → **2 True-False** → **1**

**5) str Data Type:**

• **str represents String data type.**

• **A String is a sequence of characters enclosed within single quotes or double quotes.**

• **s1='durga'**

• **s1="durga"**

**15 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

• **By using single quotes or double quotes we cannot represent multi line string literals.**

• **s1="durga soft"**

• **For this requirement we should go for triple single quotes(''') or triple double quotes(""")**

• **s1='''durga soft'''**

• **s1="""durga soft"""**

• **We can also use triple quotes to use single quote or double quote in our String.**

• **''' This is " character''' ' This i " Character '**

• **We can embed one string in another string**

• **'''This "Python class very helpful" for java students'''**

**Slicing of Strings:**

**1) slice means a piece 2) [ ] operator is called slice operator, which can be used to retrieve parts of String. 3) In Python Strings follows zero based index. 4) The index can be either +ve or -ve. 5) +ve index means forward direction from Left to Right 6) -ve index means backward direction from Right to Left -5 -4 -3 -2 -1 d u r g a 0 1 2 3 4 1) >>> s="durga" 2) >>> s[0] 3) 'd' 4) >>> s[1] 5) 'u' 6) >>> s[-1] 7) 'a' 8) >>> s[40]**

**IndexError: string index out of range**

**16 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**1) >>> s[1:40] 2) 'urga' 3) >>> s[1:] 4) 'urga' 5) >>> s[:4] 6) 'durg' 7) >>> s[:] 8) 'durga' 9) >>> 10) 11) >>> s\*3 12) 'durgadurgadurga' 13) 14) >>> len(s) 15) 5**

**Note:**

**1) In Python the following data types are considered as Fundamental Data types**

• **int**

• **float**

• **complex**

• **bool**

• **str**

**2) In Python, we can represent char values also by using str type and explicitly char type**

**is not available.**

**1) >>> c='a' 2) >>> type(c) 3) <class 'str'>**

**3) long Data Type is available in Python2 but not in Python3. In Python3 long values also**

**we can represent by using int type only.**

**4) In Python we can present char Value also by using str Type and explicitly char Type is**

**not available.**

**17 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

TYPE CASTING

֍ **We can convert one type value to another type. This conversion is called Typecasting**

**or Type coersion.** ֍ **The following are various inbuilt functions for type casting.**

**1) int() 2) float() 3) complex() 4) bool() 5) str()**

֍ **int():**

**We can use this function to convert values from other types to int**

**1) >>> int(123.987) 2) 123 3) >>> int(10+5j) 4) TypeError: can't convert complex to int 5) >>> int(True) 6) 1 7) >>> int(False) 8) 0 9) >>> int("10") 10) 10 11) >>> int("10.5") 12) ValueError: invalid literal for int() with base 10: '10.5' 13) >>> int("ten") 14) ValueError: invalid literal for int() with base 10: 'ten' 15) >>> int("0B1111") 16) ValueError: invalid literal for int() with base 10: '0B1111'**

**Note:**

**1) We can convert from any type to int except complex type. 2) If we want to convert str type to int type, compulsary str should contain only integral**

**value and should be specified in base-10.**

**18 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

֍ **float():**

**We can use float() function to convert other type values to float type.**

**1) >>> float(10) 2) 10.0 3) >>> float(10+5j) 4) TypeError: can't convert complex to float 5) >>> float(True) 6) 1.0 7) >>> float(False) 8) 0.0 9) >>> float("10") 10) 10.0 11) >>> float("10.5") 12) 10.5 13) >>> float("ten") 14) ValueError: could not convert string to float: 'ten' 15) >>> float("0B1111") 16) ValueError: could not convert string to float: '0B1111'**

**Note: 1) We can convert any type value to float type except complex type. 2) Whenever we are trying to convert str type to float type compulsary str should be either integral or floating point literal and should be specified only in base-10.**

֍ **complex():**

**We can use complex() function to convert other types to complex type.**

**Form-1: complex(x) We can use this function to convert x into complex number with real part x and imaginary part 0. Eg: 1) complex(10)==>10+0j**

**2) complex(10.5)===>10.5+0j 3) complex(True)==>1+0j 4) complex(False)==>0j 5) complex("10")==>10+0j 6) complex("10.5")==>10.5+0j 7) complex("ten") 8) ValueError: complex() arg is a malformed string**

**19 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Form-2: complex(x,y)**

**We can use this method to convert x and y into complex number such that x will be real part and y will be imaginary part.**

**Eg: complex(10, -2)** → **10-2j**

**complex(True, False)** → **1+0j**

֍ **bool():**

**We can use this function to convert other type values to bool type.**

**1) bool(0)** → **False 2) bool(1)** → **True 3) bool(10)** → **True 4) bool(10.5)** → **True 5) bool(0.178)** → **True 6) bool(0.0)** → **False 7) bool(10-2j)** → **True 8) bool(0+1.5j)** → **True 9) bool(0+0j)** → **False 10) bool("True")** → **True 11) bool("False")** → **True 12) bool("")** → **False**

**20 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

֍ **str():**

**We can use this method to convert other type values to str type.**

**1) >>> str(10) 2) '10' 3) >>> str(10.5) 4) '10.5' 5) >>> str(10+5j) 6) '(10+5j)' 7) >>> str(True)** 8) **'True'**

**Fundamental Data Types vs Immutability:**

֍ **All Fundamental Data types are immutable. i.e once we creates an object,we cannot**

**perform any changes in that object. If we are trying to change then with those changes a new object will be created. This non-chageable behaviour is called immutability.**

֍ **In Python if a new object is required, then PVM won’t create object immediately. First**

**it will check is any object available with the required content or not. If available then existing object will be reused. If it is not available then only a new object will be created. The advantage of this approach is memory utilization and performance will be improved.**

֍ **But the problem in this approach is, several references pointing to the same object, by using one reference if we are allowed to change the content in the existing object then the remaining references will be effected. To prevent this immutability concept is required. According to this once creates an object we are not allowed to change content. If we are trying to change with those changes a new object will be created.**

**1) >>> a=10 2) >>> b=10 3) >>> a is b 4) True 5) >>> id(a) 6) 1572353952 7) >>> id(b) 8) 1572353952 9) >>>**

**21 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**>>> a=True**

**>>> b=True**

**>>> a is b**

**True**

**>>> id(a)**

**1572172624**

**>>> id(b)**

**1572172624**

**>>> a='durga'**

**>>> b='durga'**

**>>> a is b**

**True**

**>>> id(a)**

**16378848**

**>>> id(b)**

**16378848**

**>>> a=10**

**>>> b=10**

**>>> id(a)**

**1572353952**

**>>> id(b)**

**1572353952**

**>>> a is b**

**True**

**>>> a=10+5j**

**>>> b=10+5j**

**>>> a is b**

**False**

**>>> id(a)**

**15980256**

**>>> id(b)**

**15979944**

**6) bytes Data Type:**

**bytes data type represens a group of byte numbers just like an array.**

**1) x = [10,20,30,40] 2) b = bytes(x) 3) type(b)** → **bytes 4) print(b[0])** → **10 5) print(b[-1])** → **40 6) >>> for i in b : print(i) 7) 8) 10 9) 20 10) 30 11) 40**

**Conclusion 1: The only allowed values for byte data type are 0 to 256. By mistake if we are trying to provide any other values then we will get value error.**

**Conclusion 2: Once we creates bytes data type value, we cannot change its values,otherwise we will get TypeError.**

**22 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Eg:**

**1) >>> x=[10,20,30,40] 2) >>> b=bytes(x) 3) >>> b[0]=100 4) TypeError: 'bytes' object does not support item assignment**

**7) bytearray Data Type:**

**bytearray is exactly same as bytes data type except that its elements can be modified.**

**Eg 1: 1) x=[10,20,30,40] 2) b = bytearray(x) 3) for i in b : print(i) 4) 10 5) 20 6) 30 7) 40 8) b[0]=100 9) for i in b: print(i) 10) 100 11) 20 12) 30 13) 40**

**Eg 2: 1) >>> x =[10,256]**

**2) >>> b = bytearray(x) 3) ValueError: byte must be in range(0, 256)**

**8) List Data Type:**

**If we want to represent a group of values as a single entity where insertion order required to preserve and duplicates are allowed then we should go for list data type.**

**1) Insertion Order is preserved 2) Heterogeneous Objects are allowed 3) Duplicates are allowed 4) Growable in nature 5) Values should be enclosed within square brackets.**

**23 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Eg: 1) list=[10,10.5,'durga',True,10]**

**2) print(list) # [10,10.5,'durga',True,10]**

**Eg: 1) list=[10,20,30,40]**

**2) >>> list[0] 3) 10 4) >>> list[-1] 5) 40 6) >>> list[1:3] 7) [20, 30] 8) >>> list[0]=100 9) >>> for i in list:print(i) 10) ... 11) 100 12) 20 13) 30 14) 40**

**list is growable in nature. i.e based on our requirement we can increase or decrease the size. 1) >>> list=[10,20,30]**

**2) >>> list.append("durga") 3) >>> list 4) [10, 20, 30, 'durga'] 5) >>> list.remove(20) 6) >>> list 7) [10, 30, 'durga'] 8) >>> list2=list\*2 9) >>> list2 10) [10, 30, 'durga', 10, 30, 'durga']**

**Note: An ordered, mutable, heterogenous collection of eleemnts is nothing but list, where duplicates also allowed.**

**24 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**9) Tuple Data Type:**

• **tuple data type is exactly same as list data type except that it is immutable.i.e we cannot chage values.**

• **Tuple elements can be represented within parenthesis.**

**Eg: 1) t=(10,20,30,40)**

**2) type(t) 3) <class 'tuple'> 4) t[0]=100 5) TypeError: 'tuple' object does not support item assignment 6) >>> t.append("durga") 7) AttributeError: 'tuple' object has no attribute 'append' 8) >>> t.remove(10) 9) AttributeError: 'tuple' object has no attribute 'remove'**

**Note: tuple is the read only version of list**

**10) Range Data Type:**

• **range Data Type represents a sequence of numbers.**

• **The elements present in range Data type are not modifiable. i.e range Data type is immutable.**

**Form-1: range(10) generate numbers from 0 to 9**

**Eg: r = range(10) for i in r : print(i)** → **0 to 9**

**Form-2: range(10, 20) generate numbers from 10 to 19**

**Eg: r = range(10,20) for i in r : print(i)** →**10 to 19**

**25 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Form-3: range(10, 20, 2) 2 means increment value**

**Eg: r = range(10,20,2) for i in r : print(i)** → **10,12,14,16,18**

**We can access elements present in the range Data Type by using index.**

**Eg: r = range(10,20) r[0]** → **10 r[15]** → **IndexError: range object index out of range**

**We cannot modify the values of range data type**

**Eg: r[0] = 100 TypeError: 'range' object does not support item assignment**

**We can create a list of values with range data type**

**Eg: 1) >>> l = list(range(10))**

**2) >>> l 3) [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]**

**11) set Data Type:**

֍ **If we want to represent a group of values without duplicates where order is not**

**important then we should go for set Data Type.**

**1) Insertion order is not preserved 2) Duplicates are not allowed 3) Heterogeneous objects are allowed 4) Index concept is not applicable 5) It is mutable collection 6) Growable in nature**

**26 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Eg: 1) s={100,0,10,200,10,'durga'} 2) s # {0, 100, 'durga', 200, 10} 3) s[0]** → **TypeError: 'set' object does not support indexing**

֍ **set is growable in nature, based on our requirement we can increase or decrease the**

**size.**

**1) >>> s.add(60) 2) >>> s 3) {0, 100, 'durga', 200, 10, 60} 4) >>> s.remove(100) 5) >>> s 6) {0, 'durga', 200, 10, 60}**

**12) frozenset Data Type:**

֍ **It is exactly same as set except that it is immutable.** ֍ **Hence we cannot use add or remove functions.**

**1) >>> s={10,20,30,40} 2) >>> fs=frozenset(s) 3) >>> type(fs) 4) <class 'frozenset'> 5) >>> fs 6) frozenset({40, 10, 20, 30}) 7) >>> for i in fs:print(i) 8) ... 9) 40 10) 10 11) 20 12) 30 13) 14) >>> fs.add(70) 15) AttributeError: 'frozenset' object has no attribute 'add' 16) >>> fs.remove(10) 17) AttributeError: 'frozenset' object has no attribute 'remove'**

**27 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**13) dict Data Type:**

֍ **If we want to represent a group of values as key-value pairs then we should go for**

**dict data type.**

֍ **Eg: d = {101:'durga',102:'ravi',103:'shiva'}**

֍ **Duplicate keys are not allowed but values can be duplicated. If we are trying to**

**insert an entry with duplicate key then old value will be replaced with new value.**

**Eg:**

**1) >>> d={101:'durga',102:'ravi',103:'shiva'} 2) >>> d[101]='sunny' 3) >>> d 4) {101: 'sunny', 102: 'ravi', 103: 'shiva'} 5) 6) We can create empty dictionary as follows 7) d={ } 8) We can add key-value pairs as follows 9) d['a']='apple' 10) d['b']='banana' 11) print(d)**

**Note: dict is mutable and the order won’t be preserved.**

**Note:**

**1) In general we can use bytes and bytearray data types to represent binary information**

**like images, video files etc 2) In Python2 long data type is available. But in Python3 it is not available and we can**

**represent long values also by using int type only. 3) In Python there is no char data type. Hence we can represent char values also by using**

**str type.**

**28 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Summary of Datatypes in Python 3 Datatype Description Is Immutable? Example Int We can use to represent the whole/integral numbers**

**Immutable >>> a=10**

**>>> type(a) <class 'int'>**

**Float We can use to represent the decimal/floating point numbers**

**Immutable >>> b=10.5**

**>>> type(b) <class 'float'>**

**Complex We can use to represent the complex numbers**

**Immutable >>> c=10+5j**

**>>> type(c) <class 'complex'> >>> c.real 10.0 >>> c.imag 5.0 Bool We can use to**

**represent the logical values (Only allowed values are True and False)**

**Immutable >>> flag=True >>> flag=False >>> type(flag) <class 'bool'>**

**Str To represent**

**sequence of Characters**

**Immutable >>> s='durga'**

**>>> type(s) <class 'str'> >>> s="durga" >>> s='''Durga Software Solutions... Ameerpet''' >>> type(s) <class 'str'> bytes To represent a**

**sequence of byte values from 0-255**

**Immutable >>> list=[1,2,3,4] >>> b=bytes(list) >>> type(b) <class 'bytes'> bytearray To represent a**

**sequence of byte values from 0-255**

**Mutable >>> list=[10,20,30]**

**>>> ba=bytearray(list) >>> type(ba) <class 'bytearray'> range To represent a range**

**of values**

**Immutable >>> r=range(10)**

**>>> r1=range(0,10) >>> r2=range(0,10,2)**

**29 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Mutable list To represent an**

**>>> l=[10,11,12,13,14,15] ordered collection of**

**>>> type(l) objects**

**<class 'list'> tuple To represent an**

**ordered collections of objects**

**Immutable >>> t=(1,2,3,4,5)**

**>>> type(t) <class 'tuple'> set To represent an**

**unordered collection of unique objects**

**Mutable >>> s={1,2,3,4,5,6}**

**>>> type(s) <class 'set'> frozenset To represent an**

**unordered collection of unique objects**

**Immutable >>> s={11,2,3,'Durga',100,'Ramu'}**

**>>> fs=frozenset(s) >>> type(fs) <class 'frozenset'> dict To represent a group**

**of key value pairs**

**Mutable >>>**

**d = {101:'durga', 102:'ramu', 103:'hari'} >>> type(d) <class 'dict'>**

**14) None Data Type:**

• **None means nothing or No value associated.**

• **If the value is not available, then to handle such type of cases None introduced.**

• **It is something like null value in Java.**

**Eg: def m1():**

**a=10**

**print(m1())**

**None**

**30 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Escape Characters:**

**In String literals we can use esacpe characters to associate a special meaning.**

**1) >>> s="durga\nsoftware" 2) >>> print(s) 3) durga 4) software 5) >>> s="durga\tsoftware" 6) >>> print(s) 7) durga software 8) >>> s="This is " symbol" 9) File "<stdin>", line 1 10) s="This is " symbol" 11) ^ 12) SyntaxError: invalid syntax 13) >>> s="This is \" symbol" 14) >>> print(s) 15) This is " symbol**

**The following are various important escape characters in Python**

**1) \n**

→ **New Line 2) \t**

→ **Horizontal Tab 3) \r**

→ **Carriage Return 4) \b**

→ **Back Space 5) \f**

→ **Form Feed 6) \v**

→ **Vertical Tab 7) \'**

→ **Single Quote 8) \"**

→ **Double Quote 9) \\** → **Back Slash Symbol**

**....**

**Constants:**

• **Constants concept is not applicable in Python.**

• **But it is convention to use only uppercase characters if we don’t want to change value.**

• **MAX\_VALUE = 10**

• **It is just convention but we can change the value.**

**31 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

OPERATORS

**32 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

• **Operator is a symbol that performs certain operations.**

• **Python provides the following set of operators**

**1) Arithmetic Operators 2) Relational Operators OR Comparison Operators 3) Logical operators 4) Bitwise oeprators 5) Assignment operators 6) Special operators**

**1) Arithmetic Operators:**

**1) +**

**2) –**

**3) \***

**4) /**

**5) %**

**6) //**

**7) \*\***

**Eg: test.py**

**1) a=10 2) b=2 3) print('a+b=',a+b) 4) print('a-b=',a-b) 5) print('a\*b=',a\*b) 6) print('a/b=',a/b) 7) print('a//b=',a//b) 8) print('a%b=',a%b) 9) print('a\*\*b=',a\*\*b)**

**Output: Python test.py OR py test.py a+b = 12 a-b= 8 a\*b= 20 a/b= 5.0**

→ **Addition**

→ **Subtraction**

→ **Multiplication**

→ **Division Operator**

→ **Modulo Operator**

→ **Floor Division Operator**

→ **Exponent Operator OR Power Operator**

**33 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**a//b= 5 a%b= 0 a\*\*b= 100**

**Eg: 1) a = 10.5 2) b=2 3) 4) a+b= 12.5 5) a-b= 8.5 6) a\*b= 21.0 7) a/b= 5.25 8) a//b= 5.0 9) a%b= 0.5 10) a\*\*b= 110.25**

**Eg: 10/2** → **5.0 10//2** → **5 10.0/2** → **5.0 10.0//2** → **5.0**

**Note:** ֍ **/ operator always performs floating point arithmetic. Hence it will always returns float**

**value.**

֍ **But Floor division (//) can perform both floating point and integral arithmetic. If**

**arguments are int type then result is int type. If atleast one argument is float type then result is float type.**

**Note:** ֍ **We can use +,\* operators for str type also.** ֍ **If we want to use + operator for str type then compulsory both arguments should be**

**str type only otherwise we will get error.**

**1) >>> "durga"+10 2) TypeError: must be str, not int 3) >>> "durga"+"10" 4) 'durga10'**

֍ **If we use \* operator for str type then compulsory one argument should be int and**

**other argument should be str type.**

**34 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

֍ **2\*"durga" "durga"\*2 2.5\*"durga"** → **TypeError: can't multiply sequence by non-int of type 'float' "durga"\*"durga"** → **TypeError: can't multiply sequence by non-int of type 'str'**

֍ **+** → **String Concatenation Operator** ֍ **\*** → **String Multiplication Operator**

**Note: For any number x, x/0 and x%0 always raises "ZeroDivisionError"**

**10/0 10.0/0 ..... 2) Relational Operators: >, >=, <, <=**

**1) a=10 2) b=20 3) print("a > b is ",a>b) 4) print("a >= b is ",a>=b) 5) print("a < b is ",a<b) 6) print("a <= b is ",a<=b) 7) 8) a > b is False 9) a >= b is False 10) a < b is True** 11) **a <= b is True**

**We can apply relational operators for str types also.**

**Eg 2: 1) a="durga" 2) b="durga" 3) print("a > b is ",a>b) 4) print("a >= b is ",a>=b) 5) print("a < b is ",a<b) 6) print("a <= b is ",a<=b) 7) 8) a > b is False 9) a >= b is True 10) a < b is False 11) a <= b is True**

**35 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Eg: 1) print(True>True) False 2) print(True>=True) True 3) print(10 >True) True 4) print(False > True) False 5) 6) print(10>'durga') 7) TypeError: '>' not supported between instances of 'int' and 'str'**

**Eg: 1) a=10 2) b=20 3) if(a>b): 4) print("a is greater than b") 5) else: 6) print("a is not greater than b")**

**Output: a is not greater than b**

**Note: Chaining of relational operators is possible. In the chaining, if all comparisons returns True then only result is True. If atleast one comparison returns False then the result is False**

**1) 10<20** → **True 2) 10<20<30** → **True 3) 10<20<30<40** → **True 4) 10<20<30<40>50** → **False**

**3) Equality Operators: ==, !=**

**We can apply these operators for any type even for incompatible types also.**

**1) >>> 10==20 2) False 3) >>> 10!= 20 4) True 5) >>> 10==True 6) False 7) >>> False==False 8) True 9) >>> "durga"=="durga" 10) True 11) >>> 10=="durga"**

**36 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

12) **False**

**Note: Chaining concept is applicable for equality operators. If atleast one comparison returns False then the result is False. Otherwise the result is True.**

**1) >>> 10==20==30==40 2) False 3) >>> 10==10==10==10 4) True**

**4) Logical Operators: and, or, not**

**We can apply for all types.**

❖ **For boolean Types Behaviour:**

**and** → **If both arguments are True then only result is True or** → **If atleast one arugemnt is True then result is True not** → **Complement**

**True and False** → **False True or False** → **True not False** → **True**

❖ **For non-boolean Types Behaviour:**

**0 means False non-zero means True empty string is always treated as False**

**x and y: If x is evaluates to false return x otherwise return y Eg: 10 and 20 0 and 20**

**If first argument is zero then result is zero otherwise result is y**

**x or y: If x evaluates to True then result is x otherwise result is y**

**10 or 20** → **10 0 or 20** → **20**

**37 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**not x: If x is evalutates to False then result is True otherwise False**

**not 10** → **False not 0** → **True**

**Eg: 1) "durga" and "durgasoft" ==>durgasoft**

**2) "" and "durga" ==>"" 3) "durga" and "" ==>"" 4) "" or "durga" ==>"durga" 5) "durga" or ""==>"durga" 6) not ""==>True 7) not "durga" ==>False**

**5) Bitwise Operators:**

֍ **We can apply these operators bitwise.** ֍ **These operators are applicable only for int and boolean types.** ֍ **By mistake if we are trying to apply for any other type then we will get Error.**

֍ **&, |, ^, ~, <<, >>**

֍ **print(4&5)** → **Valid** ֍ **print(10.5 & 5.6)**

→ **TypeError: unsupported operand type(s) for &: 'float' and 'float'**

֍ **print(True & True)** → **Valid**

֍ **&** → **If both bits are 1 then only result is 1 otherwise result is 0** ֍ **|**→ **If atleast one bit is 1 then result is 1 otherwise result is 0** ֍ **^** → **If bits are different then only result is 1 otherwise result is 0** ֍ **~** → **bitwise complement operator** ֍ **1** → **0 & 0** → **1** ֍ **<<** → **Bitwise Left Shift** ֍ **>>** → **Bitwise Right Shift**

֍ **print(4&5)** → **4** ֍ **print(4|5)** → **5** ֍ **print(4^5)** → **1**

**38 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Operator Description**

**& If both bits are 1 then only result is 1 otherwise result is 0 | If atleast one bit is 1 then result is 1 otherwise result is 0 ^ If bits are different then only result is 1 otherwise result is 0 ~ bitwise complement operator i.e 1 means 0 and 0 means 1 >> Bitwise Left shift Operator << Bitwise Right shift Operator**

**Bitwise Complement Operator (~): We have to apply complement for total bits.**

**Eg: print(~5)**→ **-6**

**Note:** ֍ **The most significant bit acts as sign bit. 0 value represents +ve number where as 1**

**represents -ve value.** ֍ **Positive numbers will be repesented directly in the memory where as -ve numbers will**

**be represented indirectly in 2's complement form.**

**6) Shift Operators:**

**<< Left Shift Operator After shifting the empty cells we have to fill with zero**

**print(10<<2)** → **40**

**0 0 0 0 1 0 1 0**

**0 0 1 0 1 0 0 0**

**>> Right Shift Operator After shifting the empty cells we have to fill with sign bit.( 0 for +ve and 1 for -ve)**

**print(10>>2)** → **2**

**0 0 0 0 1 0 1 0**

**0 0 0 0 0 0 1 0**

**39 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**We can apply bitwise operators for boolean types also**

֍ **print(True & False)** → **False** ֍ **print(True | False)** → **True** ֍ **print(True ^ False)** → **True** ֍ **print(~True)** → **-2** ֍ **print(True<<2)** → **4** ֍ **print(True>>2)** → **0**

**7) Assignment Operators:**

֍ **We can use assignment operator to assign value to the variable.**

**Eg: x = 10**

֍ **We can combine asignment operator with some other operator to form compound**

**assignment operator. Eg: x += 10** → **x = x+10**

**The following is the list of all possible compound assignment operators in Python.**

▪ **+=**

▪ **-=**

▪ **\*=**

▪ **/=**

▪ **%=**

▪ **//=**

▪ **\*\*=**

▪ **&=**

▪ **|=**

▪ **^=**

▪ **>>=**

▪ **<<=**

**Eg: 1) x=10**

**2) x+=20 3) print(x)** → **30**

**Eg: 1) x=10 2) x&=5 3) print(x)** → **0**

**40 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**8) Ternary Operator OR Conditional Operator**

**Syntax: x = firstValue if condition else secondValue**

**If condition is True then firstValue will be considered else secondValue will be considered.**

**Eg 1: 1) a,b=10,20**

**2) x=30 if a<b else 40 3) print(x) #30**

**Eg 2: Read two numbers from the keyboard and print minimum value**

**1) a=int(input("Enter First Number:")) 2) b=int(input("Enter Second Number:")) 3) min=a if a<b else b 4) print("Minimum Value:",min)**

**Output: Enter First Number:10 Enter Second Number:30 Minimum Value: 10**

**Note: Nesting of Ternary Operator is Possible.**

**Q) Program for Minimum of 3 Numbers**

**1) a=int(input("Enter First Number:")) 2) b=int(input("Enter Second Number:")) 3) c=int(input("Enter Third Number:")) 4) min=a if a<b and a<c else b if b<c else c 5) print("Minimum Value:",min)**

**Q) Program for Maximum of 3 Numbers**

**1) a=int(input("Enter First Number:")) 2) b=int(input("Enter Second Number:")) 3) c=int(input("Enter Third Number:")) 4) max=a if a>b and a>c else b if b>c else c 5) print("Maximum Value:",max)**

**41 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Eg: 1) a=int(input("Enter First Number:"))**

**2) b=int(input("Enter Second Number:")) 3) print("Both numbers are equal" if a==b else "First Number is Less than Second Nu**

**mber" if a<b else "First Number Greater than Second Number")**

**Output: D:\python\_classes>py test.py Enter First Number:10 Enter Second Number:10 Both numbers are equal**

**D:\python\_classes>py test.py Enter First Number:10 Enter Second Number:20 First Number is Less than Second Number**

**D:\python\_classes>py test.py Enter First Number:20 Enter Second Number:10 First Number Greater than Second Number**

**9) Special Operators:**

**Python defines the following 2 special operators**

**1) Identity Operators 2) Membership operators**

**1) Identity Operators**

• **We can use identity operators for address comparison.**

• **There are 2 identity operators are available**

**1) is 2) is not**

• **r1 is r2 returns True if both r1 and r2 are pointing to the same object.**

• **r1 is not r2 returns True if both r1 and r2 are not pointing to the same object.**

**Eg: 1) a=10 2) b=10 3) print(a is b) True 4) x=True**

**42 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**5) y=True 6) print( x is y) True**

**Eg: 1) a="durga" 2) b="durga" 3) print(id(a)) 4) print(id(b)) 5) print(a is b)**

**Eg: 1) list1=["one","two","three"] 2) list2=["one","two","three"] 3) print(id(list1)) 4) print(id(list2)) 5) print(list1 is list2) False 6) print(list1 is not list2) True** 7) **print(list1 == list2) True**

**Note: We can use is operator for address comparison where as == operator for content comparison.**

**2) Membership Operators:**

• **We can use Membership operators to check whether the given object present in the given collection. (It may be String, List, Set, Tuple OR Dict)**

• **In** → **Returns True if the given object present in the specified Collection**

• **not in** → **Retruns True if the given object not present in the specified Collection**

**Eg: 1) x="hello learning Python is very easy!!!"**

**2) print('h' in x) True 3) print('d' in x) False 4) print('d' not in x) True 5) print('Python' in x) True**

**Eg: 1) list1=["sunny","bunny","chinny","pinny"]**

**2) print("sunny" in list1) True 3) print("tunny" in list1) False 4) print("tunny" not in list1) True**

**43 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Operator Precedence:**

**If multiple operators present then which operator will be evaluated first is decided by operator precedence.**

**Eg: print(3+10\*2)** → **23 print((3+10)\*2)** → **26**

**The following list describes operator precedence in Python**

**1) ()** → **Parenthesis 2) \*\*** → **Exponential Operator 3) ~, -** → **Bitwise Complement Operator, Unary Minus Operator 4) \*, /, %, //** → **Multiplication, Division, Modulo, Floor Division 5) +, -** → **Addition, Subtraction 6) <<, >>** → **Left and Right Shift 7) &** → **Bitwise And 8) ^** → **Bitwise X-OR 9) |** → **Bitwise OR 10) >, >=, <, <=, ==, !=** → **Relational OR Comparison Operators 11) =, +=, -=, \*=...** → **Assignment Operators 12) is , is not** → **Identity Operators 13) in , not in** → **Membership operators 14) not** → **Logical not 15) and** → **Logical and 16) or** → **Logical or**

**1) a=30 2) b=20 3) c=10 4) d=5 5) print((a+b)\*c/d)** → **100.0 6) print((a+b)\*(c/d))** → **100.0 7) print(a+(b\*c)/d)** → **70.0 8) 9) 3/2\*4+3+(10/5)\*\*3-2 10) 3/2\*4+3+2.0\*\*3-2 11) 3/2\*4+3+8.0-2 12) 1.5\*4+3+8.0-2 13) 6.0+3+8.0-2 14) 15.0**

**44 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Mathematical Functions (math Module)**

֍ **A Module is collection of functions, variables and classes etc.** ֍ **math is a module that contains several functions to perform mathematical operations.** ֍ **If we want to use any module in Python, first we have to import that module.**

**import math** ֍ **Once we import a module then we can call any function of that module.**

**1) import math 2) print(math.sqrt(16)) 3) print(math.pi)**

**Output 4.0 3.141592653589793**

֍ **We can create alias name by using as keyword. import math as m** ֍ **Once we create alias name, by using that we can access functions and variables of that**

**module.**

**1) import math as m 2) print(m.sqrt(16)) 3) print(m.pi)**

֍ **We can import a particular member of a module explicitly as follows**

**from math import sqrt from math import sqrt,pi**

֍ **If we import a member explicitly then it is not required to use module name while**

**accessing.**

**1) from math import sqrt,pi 2) print(sqrt(16)) 3) print(pi) 4) print NameError: name** ⎝**(math.pi) 'math' is not defined**

**45 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Important Functions of math Module:**

**1) ceil(x) 2) floor(x) 3) pow(x,y) 4) factorial(x) 5) trunc(x) 6) gcd(x,y) 7) sin(x) 8) cos(x) 9) tan(x) 10) ....**

**Important Variables of math Module:**

**pi3.14 e** → **2.71 inf** → **infinity nan** → **not a number**

**Q) Write a Python Program to find Area of Circle pi\*r\*\*2**

**1) from math import pi 2) r = 16 3) print("Area of Circle is :",pi\*r\*\*2)**

**Output: Area of Circle is: 804.247719318987**

**46 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

INPUT AND OUTPUT STATEMENTS

**Reading Dynamic Input from the Keyboard: In Python 2 the following 2 functions are available to read dynamic input from the keyboard.**

**1) raw\_input() 2) input()**

**1) raw\_input():**

**This function always reads the data from the keyboard in the form of String Format. We have to convert that string type to our required type by using the corresponding type casting methods.**

**Eg: x = raw\_input("Enter First Number:")**

**print(type(x))** → **It will always print str type only for any input type**

**2) input():**

**input() function can be used to read data directly in our required format.We are not required to perform type casting.**

**x = input("Enter Value) type(x)**

**10** → **int "durga"**→ **str 10.5** → **float True** → **bool**

**\*\*\*Note:**

• **But in Python 3 we have only input() method and raw\_input() method is not available.**

• **Python3 input() function behaviour exactly same as raw\_input() method of Python2. i.e every input value is treated as str type only.**

• **raw\_input() function of Python 2 is renamed as input() function in Python 3.**

**1) >>> type(input("Enter value:")) 2) Enter value:10 3) <class 'str'> 4) 5) Enter value:10.5 6) <class 'str'> 7)**

**47 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**8) Enter value:True 9) <class 'str'>**

**Q) Write a program to read 2 numbers from the keyboard and print sum**

**1) x=input("Enter First Number:") 2) y=input("Enter Second Number:") 3) i = int(x) 4) j = int(y) 5) print("The Sum:",i+j)**

**Enter First Number: 100 Enter Second Number: 200 The Sum: 300 -----------------------------------------------------**

**1) x=int(input("Enter First Number:")) 2) y=int(input("Enter Second Number:")) 3) print("The Sum:",x+y)**

**-----------------------------------------------------------**

**print("The Sum:",int(input("Enter First Number:"))+int(input("Enter Second Number:")))**

**Q) Write a Program to read Employee Data from the Keyboard and**

**print that Data**

**1) eno=int(input("Enter Employee No:")) 2) ename=input("Enter Employee Name:") 3) esal=float(input("Enter Employee Salary:")) 4) eaddr=input("Enter Employee Address:") 5) married=bool(input("Employee Married ?[True|False]:")) 6) print("Please Confirm Information") 7) print("Employee No :",eno) 8) print("Employee Name :",ename) 9) print("Employee Salary :",esal) 10) print("Employee Address :",eaddr) 11) print("Employee Married ? :",married)**

**D:\Python\_classes>py test.py Enter Employee No:100 Enter Employee Name:Sunny Enter Employee Salary:1000 Enter Employee Address:Mumbai**

**48 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Employee Married ?[True|False]:True Please Confirm Information**

**Employee No : 100 Employee Name : Sunny Employee Salary : 1000.0 Employee Address : Mumbai Employee Married ? : True**

**How to read multiple values from the keyboard in a single line:**

**1) a,b= [int(x) for x in input("Enter 2 numbers :").split()] 2) print("Product is :", a\*b)**

**D:\Python\_classes>py test.py Enter 2 numbers :10 20 Product is : 200**

**Note: split() function can take space as seperator by default .But we can pass anything as seperator.**

**Q) Write a program to read 3 float numbers from the keyboard**

**with, seperator and print their sum**

**1) a,b,c= [float(x) for x in input("Enter 3 float numbers :").split(',')] 2) print("The Sum is :", a+b+c)**

**D:\Python\_classes>py test.py Enter 3 float numbers :10.5,20.6,20.1 The Sum is : 51.2**

**eval(): eval Function take a String and evaluate the Result.**

**Eg: x = eval(“10+20+30”)**

**print(x) Output: 60**

**Eg: x = eval(input(“Enter Expression”))**

**Enter Expression: 10+2\*3/4 Output: 11.5**

**49 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**eval() can evaluate the Input to list, tuple, set, etc based the provided Input.**

**Eg: Write a Program to accept list from the keynboard on the display**

**1) l = eval(input(“Enter List”)) 2) print (type(l)) 3) print(l)** COMMAND LINE ARGUMENTS

• **argv is not Array it is a List. It is available sys Module.**

• **The Argument which are passing at the time of execution are called Command Line Arguments.**

**Eg: D:\Python\_classes py test.py 10 20 30**

**Command Line Arguments**

**Within the Python Program this Command Line Arguments are available in argv. Which is present in SYS Module.**

**Note: argv[0] represents Name of Program. But not first Command Line Argument. argv[1] represent First Command Line Argument.**

**Program: To check type of argv from sys**

**import argv print(type(argv))**

**D:\Python\_classes\py test.py**

**Write a Program to display Command Line Arguments**

**1) from sys import argv 2) print(“The Number of Command Line Arguments:”, len(argv)) 3) print(“The List of Command Line Arguments:”, argv) 4) print(“Command Line Arguments one by one:”) 5) for x in argv: 6) print(x)**

**test.py 10 20 30**

**50 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**D:\Python\_classes>py test.py 10 20 30 The Number of Command Line Arguments: 4 The List of Command Line Arguments: [‘test.py’, ‘10’,’20’,’30’] Command Line Arguments one by one: test.py 10 20 30 ---------------------------**

**1) from sys import argv 2) sum=0 3) args=argv[1:] 4) for x in args : 5) n=int(x) 6) sum=sum+n 7) print("The Sum:",sum)**

**D:\Python\_classes>py test.py 10 20 30 40 The Sum: 100**

**Note 1: Usually space is seperator between command line arguments. If our command line argument itself contains space then we should enclose within double quotes(but not single quotes)**

**1) from sys import argv 2) print(argv[1])**

**D:\Python\_classes>py test.py Sunny Leone Sunny**

**D:\Python\_classes>py test.py 'Sunny Leone' 'Sunny**

**D:\Python\_classes>py test.py "Sunny Leone" Sunny Leone**

**Note 2: Within the Python program command line arguments are available in the String form. Based on our requirement, we can convert into corresponding type by using type casting methods.**

**1) from sys import argv 2) print(argv[1]+argv[2]) 3) print(int(argv[1])+int(argv[2]))**

**51 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**D:\Python\_classes>py test.py 10 20 1020 30 Note 3: If we are trying to access command line arguments with out of range index then we will get Error.**

**1) from sys import argv 2) print(argv[100])**

**D:\Python\_classes>py test.py 10 20 IndexError: list index out of range**

**Note: In Python there is argparse module to parse command line arguments and display some help messages whenever end user enters wrong input.**

**input() raw\_input()**

**Command Line Arguments**

**Output Statements:**

**We can use print() function to display output.**

**Form-1: print() without any argument Just it prints new line character**

**Form-2:**

**1) print(String): 2) print("Hello World") 3) We can use escape characters also 4) print("Hello \n World") 5) print("Hello\tWorld") 6) We can use repetetion operator (\*) in the string 7) print(10\*"Hello") 8) print("Hello"\*10) 9) We can use + operator also 10) print("Hello"+"World")**

**52 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Note:** ֍ **If both arguments are String type then + operator acts as concatenation operator.** ֍ **If one argument is string type and second is any other type like int then we will get**

**Error.** ֍ **If both arguments are number type then + operator acts as arithmetic addition**

**operator.**

**Note:**

**1) print("Hello"+"World") 2) print("Hello","World")**

**HelloWorld Hello World**

**Form-3: print() with variable number of arguments**

**1) a,b,c=10,20,30 2) print("The Values are :",a,b,c)**

**Output: The Values are : 10 20 30**

**By default output values are seperated by space.If we want we can specify seperator by using "sep" attribute**

**1) a,b,c=10,20,30 2) print(a,b,c,sep=',') 3) print(a,b,c,sep=':')**

**D:\Python\_classes>py test.py 10,20,30 10:20:30**

**Form-4:print() with end attribute**

**1) print("Hello") 2) print("Durga") 3) print("Soft")**

**Output: Hello Durga Soft**

**53 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**If we want output in the same line with space**

**1) print("Hello",end=' ') 2) print("Durga",end=' ') 3) print("Soft")**

**Output: Hello Durga Soft**

**Note: The default value for end attribute is \n, which is nothing but new line character.**

**Form-5: print(object) statement**

**We can pass any object (like list, tuple, set etc) as argument to the print() statement.**

**1) l=[10,20,30,40] 2) t=(10,20,30,40) 3) print(l) 4) print(t)**

**Form-6: print(String, variable list)**

**We can use print() statement with String and any number of arguments.**

**1) s = "Durga" 2) a = 48 3) s1 ="Java" 4) s2 ="Python" 5) print("Hello",s,"Your Age is",a) 6) print("You are teaching",s1,"and",s2)**

**Output: Hello Durga Your Age is 48 You are teaching java and Python**

**Form-7: print (formatted string)**

**1) %i** → **int 2) %d** → **int 3) %f** → **float 4) %s** → **String type**

**Syntax: print("formatted string" %(variable list))**

**54 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Eg 1: 1) a=10 2) b=20 3) c=30 4) print("a value is %i" %a) 5) print("b value is %d and c value is %d" %(b,c))**

**Output a value is 10 b value is 20 and c value is 30**

**Eg 2: 1) s="Durga"**

**2) list=[10,20,30,40] 3) print("Hello %s ...The List of Items are %s" %(s,list))**

**Output: Hello Durga ...The List of Items are [10, 20, 30, 40]**

**Form-8: print() with replacement operator {}**

**Eg:**

**1) name = "Durga" 2) salary = 10000 3) gf = "Sunny" 4) print("Hello {0} your salary is {1} and Your Friend {2} is waiting".**

**format(name,salary,gf)) 5) print("Hello {x} your salary is {y} and Your Friend {z} is waiting".**

**format(x=name,y=salary,z=gf))**

**Output Hello Durga your salary is 10000 and Your Friend Sunny is waiting Hello Durga your salary is 10000 and Your Friend Sunny is waiting**

**55 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**FLOW CONTROL**

**56 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Flow control describes the order in which statements will be executed at runtime.**

**I. Conditional Statements**

**1) if if condition : statement**

**OR**

**if condition :**

**statement-1 statement-2 statement-3**

**If condition is true then statements will be executed.**

**Eg:**

**1) name=input("Enter Name:") 2) if name=="durga" : 3) print("Hello Durga Good Morning") 4) print("How are you!!!")**

**D:\Python\_classes>py test.py Enter Name:durga Hello Durga Good Morning How are you!!!**

**Control Flow**

**Conditional Statements**

**Transfer Statements**

**Iterative Statements**

**1) if 2) if-elif 3) if-elif-else**

**1) break 2) continue 3) pass**

**1) for 2) while**

**57 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**D:\Python\_classes>py test.py Enter Name: Ravi How are you!!!**

**2) if-else:**

**if condition:**

**Action-1 else:**

**Action-2**

**if condition is true then Action-1 will be executed otherwise Action-2 will be executed.**

**1) name=input("Enter Name:") 2) if name=="durga" : 3) print("Hello Durga Good Morning") 4) else: 5) print("Hello Guest Good Moring") 6) print("How are you!!!")**

**D:\Python\_classes>py test.py Enter Name:durga Hello Durga Good Morning How are you!!!**

**D:\Python\_classes>py test.py Enter Name:Ravi Hello Guest Good Moring How are you!!!**

**3) if-elif-else:**

**if condition1: Action-1 elif condition2:**

**Action-2 elif condition3:**

**Action-3 elif condition4:**

**Action-4 ... else:**

**Default Action**

**Based condition the corresponding action will be executed.**

**58 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**1) brand=input("Enter Your Favourite Brand:") 2) if brand=="RC" : 3) print("It is childrens brand") 4) elif brand=="KF": 5) print("It is not that much kick") 6) elif brand=="FO": 7) print("Buy one get Free One") 8) else : 9) print("Other Brands are not recommended")**

**D:\Python\_classes>py test.py Enter Your Favourite Brand:RC It is childrens brand**

**D:\Python\_classes>py test.py Enter Your Favourite Brand:KF It is not that much kick**

**D:\Python\_classes>py test.py Enter Your Favourite Brand: KALYANI Other Brands are not recommended**

**Note: 1) else part is always optional. Hence the following are various possible syntaxes.**

**1) If 2) if – else 3) if-elif-else 4) if-elif 2) There is no switch statement in Python**

**Q) Write a Program to find Biggest of given 2 Numbers from the**

**Commad Prompt?**

**1) n1=int(input("Enter First Number:")) 2) n2=int(input("Enter Second Number:")) 3) if n1>n2: 4) print("Biggest Number is:",n1) 5) else : 6) print("Biggest Number is:",n2)**

**D:\Python\_classes>py test.py Enter First Number:10 Enter Second Number:20 Biggest Number is: 20**

**59 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Q) Write a Program to find Biggest of given 3 Numbers from the**

**Commad Prompt?**

**1) n1=int(input("Enter First Number:")) 2) n2=int(input("Enter Second Number:")) 3) n3=int(input("Enter Third Number:")) 4) if n1>n2 and n1>n3: 5) print("Biggest Number is:",n1) 6) elif n2>n3: 7) print("Biggest Number is:",n2) 8) else : 9) print("Biggest Number is:",n3)**

**D:\Python\_classes>py test.py Enter First Number:10 Enter Second Number:20 Enter Third Number:30 Biggest Number is: 30**

**D:\Python\_classes>py test.py Enter First Number:10 Enter Second Number:30 Enter Third Number:20 Biggest Number is: 30**

**Q) Write a program to find smallest of given 2 numbers? Q) Write a program to find smallest of given 3 numbers? Q) Write a program to check whether the given number is even or odd?**

**Q) Write a Program to Check whether the given Number is in**

**between 1 and 100?**

**1) n=int(input("Enter Number:")) 2) if n>=1 and n<=10 : 3) print("The number",n,"is in between 1 to 10") 4) else: 5) print("The number",n,"is not in between 1 to 10")**

**60 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Q) Write a Program to take a Single Digit Number from the Key**

**Board and Print is Value in English Word?**

**1) 0** → **ZERO 2) 1** → **ONE 3) 4) n=int(input("Enter a digit from o to 9:")) 5) if n==0 : 6) print("ZERO") 7) elif n==1: 8) print("ONE") 9) elif n==2: 10) print("TWO") 11) elif n==3: 12) print("THREE") 13) elif n==4: 14) print("FOUR") 15) elif n==5: 16) print("FIVE") 17) elif n==6: 18) print("SIX") 19) elif n==7: 20) print("SEVEN") 21) elif n==8: 22) print("EIGHT") 23) elif n==9: 24) print("NINE") 25) else: 26) print("PLEASE ENTER A DIGIT FROM 0 TO 9")**

**61 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**II. Iterative Statements**

֍ **If we want to execute a group of statements multiple times then we should go for**

**Iterative statements.** ֍ **Python supports 2 types of iterative statements. 1) for loop 2) while loop**

**1) for loop:**

**If we want to execute some action for every element present in some sequence (it may be string or collection) then we should go for for loop.**

**Syntax: for x in sequence:**

**Body**

**Where sequence can be string or any collection. Body will be executed for every element present in the sequence.**

**Eg 1: To print characters present in the given string**

**1) s="Sunny Leone" 2) for x in s : 3) print(x)**

**Output S u n n y L e o n e Eg 2: To print characters present in string index wise:**

**1) s=input("Enter some String: ") 2) i=0 3) for x in s : 4) print("The character present at ",i,"index is :",x) 5) i=i+1**

**62 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**D:\Python\_classes>py test.py Enter some String: Sunny Leone The character present at 0 index is : S The character present at 1 index is : u The character present at 2 index is : n The character present at 3 index is : n The character present at 4 index is : y The character present at 5 index is : The character present at 6 index is : L The character present at 7 index is : e The character present at 8 index is : o The character present at 9 index is : n The character present at 10 index is : e**

**Eg 3: To print Hello 10 times**

**1) for x in range(10) : 2) print("Hello")**

**Eg 4: To display numbers from 0 to 10**

**1) for x in range(11) : 2) print(x)**

**Eg 5: To display odd numbers from 0 to 20**

**1) for x in range(21) : 2) if (x%2!=0): 3) print(x)**

**Eg 6: To display numbers from 10 to 1 in descending order**

**1) for x in range(10,0,-1) : 2) print(x)**

**Eg 7: To print sum of numbers presenst inside list**

**1) list = eval(input("Enter List:")) 2) sum=0; 3) for x in list: 4) sum=sum+x; 5) print("The Sum=",sum)**

**63 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**D:\Python\_classes>py test.py Enter List:[10,20,30,40] The Sum= 100**

**D:\Python\_classes>py test.py Enter List:[45,67] The Sum= 112**

**2) while loop:**

**If we want to execute a group of statements iteratively until some condition false,then we should go for while loop.**

**Syntax: while condition :**

**body**

**Eg: To print numbers from 1 to 10 by using while loop**

**1) x = 1 2) while x <= 10: 3) print(x) 4) x = x+1**

**Eg: To display the sum of first n numbers**

**1) n=int(input("Enter number:")) 2) sum=0 3) i=1 4) while i<=n: 5) sum=sum+i 6) i=i+1 7) print("The sum of first",n,"numbers is :",sum)**

**Eg: Write a program to prompt user to enter some name until entering Durga**

**1) name="" 2) while name!="durga": 3) name=input("Enter Name:") 4) print("Thanks for confirmation")**

**64 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Infinite Loops:**

**1) i=0; 2) while True : 3) i=i+1; 4) print("Hello",i)**

**Nested Loops: Sometimes we can take a loop inside another loop,which are also known as nested loops.**

**1) for i in range(4): 2) for j in range(4): 3) print("i=",i," j=",j)**

**Output D:\Python\_classes>py test.py i= 0 j= 0 i= 0 j= 1 i= 0 j= 2 i= 0 j= 3 i= 1 j= 0 i= 1 j= 1 i= 1 j= 2 i= 1 j= 3 i= 2 j= 0 i= 2 j= 1 i= 2 j= 2 i= 2 j= 3 i= 3 j= 0 i= 3 j= 1 i= 3 j= 2 i= 3 j= 3**

**Q) Write a Program to dispaly \*'s in Right Angled Triangled Form**

**\* 1) n = int(input("Enter number of rows:")) \* \***

**2) for i in range(1,n+1): \* \* \***

**3) for j in range(1,i+1): \* \* \* \* \* \* \* \* \***

**4) print("\*",end=" ") 5) print()**

**Alternative Way \* \* \* \* \* \* \* \* \* \* \* \* \* 1) n = int(input("Enter number of rows:"))**

**2) for i in range(1,n+1): 3) print("\* " \* i)**

**65 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Q) Write a Program to display \*'s in Pyramid Style**

**(Also known as Equivalent Triangle)**

**III. Transfer Statements**

**1) break:**

**We can use break statement inside loops to break loop execution based on some condition.**

**1) for i in range(10): 2) if i==7: 3) print("processing is enough..plz break") 4) break 5) print(i)**

**D:\Python\_classes>py test.py 0 1 2 3 4 5 6 processing is enough..plz break**

**Eg:**

**1) cart=[10,20,600,60,70] 2) for item in cart: 3) if item>500: 4) print("To place this order insurence must be required") 5) break 6) print(item)**

**\* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \***

**1) n = int(input("Enter number of rows:")) 2) for i in range(1,n+1): 3) print(" " \* (n-i),end="") 4) print("\* "\*i)**

**66 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**D:\Python\_classes>py test.py 10 20 To place this order insurence must be required**

**2) continue:**

**We can use continue statement to skip current iteration and continue next iteration.**

**Eg 1: To print odd numbers in the range 0 to 9**

**1) for i in range(10): 2) if i%2==0: 3) continue 4) print(i)**

**D:\Python\_classes>py test.py 1 3 5 7 9 Eg 2: 1) cart=[10,20,500,700,50,60]**

**2) for item in cart: 3) if item>=500: 4) print("We cannot process this item :",item) 5) continue 6) print(item)**

**D:\Python\_classes>py test.py 10 20 We cannot process this item : 500 We cannot process this item : 700 50 60**

**67 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Eg 3: 1) numbers=[10,20,0,5,0,30]**

**2) for n in numbers: 3) if n==0: 4) print("Hey how we can divide with zero..just skipping") 5) continue 6) print("100/{} = {}".format(n,100/n))**

**Output 100/10 = 10.0 100/20 = 5.0 Hey how we can divide with zero..just skipping 100/5 = 20.0 Hey how we can divide with zero..just skipping 100/30 = 3.3333333333333335**

**Loops with else Block:**

• **Inside loop execution, if break statement not executed, then only else part will be executed.**

• **else means loop without break.**

**1) cart=[10,20,30,40,50] 2) for item in cart: 3) if item>=500: 4) print("We cannot process this order") 5) break 6) print(item) 7) else: 8) print("Congrats ...all items processed successfully")**

**Output 10 20 30 40 50 Congrats ...all items processed successfully**

**68 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Eg: 1) cart=[10,20,600,30,40,50]**

**2) for item in cart: 3) if item>=500: 4) print("We cannot process this order") 5) break 6) print(item) 7) else: 8) print("Congrats ...all items processed successfully")**

**Output D:\Python\_classes>py test.py 10 20 We cannot process this order**

**Q) What is the difference between for loop and while loop**

**in Python?** ֍ **We can use loops to repeat code execution** ֍ **Repeat code for every item in sequence** → **for loop** ֍ **Repeat code as long as condition is true** → **while loop**

**Q) How to exit from the loop? By using break statement**

**Q)How to skip some iterations inside loop? By using continue statement.**

**Q)When else part will be executed wrt loops? If loop executed without break**

**3) pass statement:**

• **pass is a keyword in Python.**

• **In our programming syntactically if block is required which won't do anything then we can define that empty block with pass keyword.**

**pass**

**|- It is an empty statement |- It is null statement |- It won't do anything**

**Eg: if True:**

**SyntaxError: unexpected EOF while parsing if True: pass** → **valid**

**69 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**def m1(): SyntaxError: unexpected EOF while parsing**

**def m1(): pass**

**Use Case of pass:**

**Sometimes in the parent class we have to declare a function with empty body and child class responsible to provide proper implementation. Such type of empty body we can define by using pass keyword. (It is something like abstract method in Java)**

**Eg: def m1(): pass**

**1) for i in range(100): 2) if i%9==0: 3) print(i) 4) else:pass**

**D:\Python\_classes>py test.py 0 9 18 27 36 45 54 63 72 81 90 99 del Statement:**

• **del is a keyword in Python.**

• **After using a variable, it is highly recommended to delete that variable if it is no longer required,so that the corresponding object is eligible for Garbage Collection.**

• **We can delete variable by using del keyword.**

**1) x = 10 2) print(x) 3) del x**

**70 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**After deleting a variable we cannot access that variable otherwise we will get NameError.**

**1) x = 10 2) del x 3) print(x)**

**NameError: name 'x' is not defined.**

**Note: We can delete variables which are pointing to immutable objects.But we cannot delete the elements present inside immutable object.**

**1) s = "durga" 2) print(s) 3) del s** → **valid 4) del s[0]** → **TypeError: 'str' object doesn't support item deletion**

**Difference between del and None: In the case del, the variable will be removed and we cannot access that variable(unbind operation)**

**1) s = "durga" 2) del s 3) print(s)** → **NameError: name 's' is not defined.**

**But in the case of None assignment the variable won't be removed but the corresponding object is eligible for Garbage Collection (re bind operation). Hence after assigning with None value, we can access that variable.**

**1) s = "durga" 2) s = None 3) print(s)** → **None**

**71 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**STRING** DATA TYPE

**72 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**The most commonly used object in any project and in any programming language is String only. Hence we should aware complete information about String data type.**

**What is String? Any sequence of characters within either single quotes or double quotes is considered as a String.**

**Syntax: s = 'durga' s = "durga"**

**Note: In most of other languges like C, C++, Java, a single character with in single quotes is treated as char data type value. But in Python we are not having char data type.Hence it is treated as String only.**

**Eg: >>> ch = 'a' >>> type(ch) <class 'str'>**

**How to define multi-line String Literals? We can define multi-line String literals by using triple single or double quotes.**

**Eg: >>> s = '''durga software solutions'''**

**We can also use triple quotes to use single quotes or double quotes as symbol inside String literal.**

**1) s = 'This is ' single quote symbol'** → **Invalid 2) s = 'This is \' single quote symbol'** → **Valid 3) s = "This is ' single quote symbol"** → **Valid 4) s = 'This is " double quotes symbol'** → **Valid 5) s = 'The "Python Notes" by 'durga' is very helpful'** → **Invalid 6) s = "The "Python Notes" by 'durga' is very helpful"** → **Invalid 7) s = 'The \"Python Notes\" by \'durga\' is very helpful'** → **Valid 8) s = '''The "Python Notes" by 'durga' is very helpful'''** → **Valid**

**73 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**How to Access Characters of a String? We can access characters of a string by using the following ways.**

**1) By using index 2) By using slice operator**

**1) Accessing Characters By using Index:**

• **Python supports both +ve and -ve Index.**

• **+ve Index means Left to Right (Forward Direction)**

• **-ve Index means Right to Left (Backward Direction) Eg: s = 'durga'**

**1) >>> s='durga' 2) >>> s[0] 3) 'd' 4) >>> s[4] 5) 'a' 6) >>> s[-1] 7) 'a' 8) >>> s[10] 9) IndexError: string index out of range**

**Note: If we are trying to access characters of a string with out of range index then we will get error saying: IndexError**

**Q) Write a Program to Accept some String from the Keyboard and display its**

**Characters by Index wise (both Positive and Negative Index)**

**test.py:**

**1) s=input("Enter Some String:") 2) i=0 3) for x in s: 4) print("The character present at positive index {} and at nEgative index {} is {}".fo**

**rmat(i,i-len(s),x)) 5) i=i+1**

**Output: D:\python\_classes>py test.py**

**Enter Some String:durga The character present at positive index 0 and at nEgative index -5 is d The character present at positive index 1 and at nEgative index -4 is u The character present at positive index 2 and at nEgative index -3 is r The character present at positive index 3 and at nEgative index -2 is g The character present at positive index 4 and at nEgative index -1 is a**

**74 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**2) Accessing Characters by using Slice Operator:**

• **Syntax: s[bEginindex:endindex:step]**

• **Begin Index: From where we have to consider slice (substring)**

• **End Index: We have to terminate the slice (substring) at endindex-1**

• **Step: Incremented Value.**

**Note:**

• **If we are not specifying bEgin index then it will consider from bEginning of the string.**

• **If we are not specifying end index then it will consider up to end of the string.**

• **The default value for step is 1.**

**1) >>> s="Learning Python is very very easy!!!" 2) >>> s[1:7:1] 3) 'earnin' 4) >>> s[1:7] 5) 'earnin' 6) >>> s[1:7:2] 7) 'eri' 8) >>> s[:7] 9) 'Learnin' 10) >>> s[7:] 11) 'g Python is very very easy!!!' 12) >>> s[::] 13) 'Learning Python is very very easy!!!' 14) >>> s[:] 15) 'Learning Python is very very easy!!!' 16) >>> s[::-1] 17) '!!!ysae yrev yrev si nohtyP gninraeL'**

**Behaviour of Slice Operator: 1) s[bEgin:end:step] 2) Step value can be either +ve or –ve 3) If +ve then it should be forward direction(left to right) and we have to consider bEgin**

**to end-1 4) If -ve then it should be backward direction (right to left) and we have to consider bEgin**

**to end+1.**

**\*\*\*Note:**

• **In the backward direction if end value is -1 then result is always empty.**

• **In the forward direction if end value is 0 then result is always empty.**

**75 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**In Forward Direction: default value for bEgin: 0 default value for end: length of string default value for step: +1**

**In Backward Direction: default value for bEgin: -1 default value for end: -(length of string+1)**

**Note: Either forward or backward direction, we can take both +ve and -ve values for bEgin and end index.**

**Slice Operator Case Study: 1) S = 'abcdefghij' 2) s[1:6:2]** → **'bdf' 3) s[::1]** → **'abcdefghij' 4) s[::-1]** → **'jihgfedcba' 5) s[3:7:-1]** → **'' 6) s[7:4:-1]** → **'hgf' 7) s[0:10000:1]** → **'abcdefghij' 8) s[-4:1:-1]** → **'gfedc' 9) s[-4:1:-2]** → **'gec' 10) s[5:0:1]** → **'' 11) s[9:0:0]** → **ValueError: slice step cannot be zero 12) s[0:-10:-1]** → **'' 13) s[0:-11:-1]** → **'a' 14) s[0:0:1]** → **'' 15) s[0:-9:-2]** → **'' 16) s[-5:-9:-2]** → **'fd' 17) s[10:-1:-1]** → **'' 18) s[10000:2:-1]** → **'jihgfed'**

**Note: Slice operator never raises IndexError**

**Mathematical Operators for String:**

**We can apply the following mathematical operators for Strings.**

**1) + operator for concatenation 2) \* operator for repetition**

• **print("durga"+"soft")** → **durgasoft**

• **print("durga"\*2)** → **durgadurga**

**76 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Note: 1) To use + operator for Strings, compulsory both arguments should be str type. 2) To use \* operator for Strings, compulsory one argument should be str and other**

**argument should be int.**

**len() in-built Function: We can use len() function to find the number of characters present in the string. Eg: s = 'durga' print(len(s))** → **5**

**Q) Write a Program to access each Character of String in**

**Forward and Backward Direction by using while Loop?**

**1) s = "Learning Python is very easy !!!" 2) n = len(s) 3) i = 0 4) print("Forward direction") 5) while i<n: 6) print(s[i],end=' ') 7) i +=1 8) print("Backward direction") 9) i = -1 10) while i >= -n: 11) print(s[i],end=' ') 12) i = i-1**

**Alternative ways:**

**1) s = "Learning Python is very easy !!!" 2) print("Forward direction") 3) for i in s: 4) print(i,end=' ') 5) print("Forward direction") 6) for i in s[::]: 7) print(i,end=' ') 8) 9) print("Backward direction") 10) for i in s[::-1]: 11) print(i,end=' ')**

**77 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Checking Membership: We can check whether the character or string is the member of another string or not by using in and not in operators**

**s = 'durga' print('d' in s)** → **True print('z' in s)** → **False**

**1) s = input("Enter main string:") 2) subs = input("Enter sub string:") 3) if subs in s: 4) print(subs,"is found in main string") 5) else: 6) print(subs,"is not found in main string")**

**Output: D:\python\_classes>py test.py Enter main string:durgasoftwaresolutions Enter sub string:durga durga is found in main string**

**D:\python\_classes>py test.py Enter main string:durgasoftwaresolutions Enter sub string:python python is not found in main string**

**Comparison of Strings:**

• **We can use comparison operators (<, <=, >, >=) and equality operators (==, !=) for strings.**

• **Comparison will be performed based on alphabetical order.**

**1) s1=input("Enter first string:") 2) s2=input("Enter Second string:") 3) if s1==s2: 4) print("Both strings are equal") 5) elif s1<s2: 6) print("First String is less than Second String") 7) else: 8) print("First String is greater than Second String")**

**Output: D:\python\_classes>py test.py Enter first string:durga**

**78 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**Enter Second string:durga Both strings are equal**

**D:\python\_classes>py test.py Enter first string:durga Enter Second string:ravi First String is less than Second String**

**D:\python\_classes>py test.py Enter first string:durga Enter Second string:anil First String is greater than Second String**

**Removing Spaces from the String:**

**We can use the following 3 methods**

**1) rstrip()** → **To remove spaces at right hand side 2) lstrip()** →**To remove spaces at left hand side 3) strip()** → **To remove spaces both sides**

**1) city=input("Enter your city Name:") 2) scity=city.strip() 3) if scity=='Hyderabad': 4) print("Hello Hyderbadi..Adab") 5) elif scity=='Chennai': 6) print("Hello Madrasi...Vanakkam") 7) elif scity=="Bangalore": 8) print("Hello Kannadiga...Shubhodaya") 9) else: 10) print("your entered city is invalid")**

**Finding Substrings:**

**We can use the following 4 methods**

**For forward direction:**

**1) find() 2) index()**

**For backward direction:**

**1) rfind() 2) rindex()**

**79 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**

**find():**

**s.find(substring) Returns index of first occurrence of the given substring. If it is not available then we will get -1.**

**1) s="Learning Python is very easy" 2) print(s.find("Python")) #9 3) print(s.find("Java")) # -1 4) print(s.find("r"))#3 5) print(s.rfind("r"))#21**

**Note: By default find() method can search total string. We can also specify the boundaries to search.**

**s.find(substring,bEgin,end) It will always search from bEgin index to end-1 index.**

**1) s="durgaravipavanshiva" 2) print(s.find('a'))#4 3) print(s.find('a',7,15))#10 4) print(s.find('z',7,15))#-1**

**index(): index() method is exactly same as find() method except that if the specified substring is not available then we will get ValueError.**

**1) s=input("Enter main string:") 2) subs=input("Enter sub string:") 3) try: 4) n=s.index(subs) 5) except ValueError: 6) print("substring not found") 7) else: 8) print("substring found")**

**Output: D:\python\_classes>py test.py Enter main string:learning python is very easy Enter sub string:python substring found**

**80 DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,**

☎ **040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com**