* Python is a free and open source language. That means, you can read/write/modify and distribute the source code of the python script.
* Python was developed by Guido Van Rossum and was released in 1991
* There are two versions of python available; Python 2x(old dated now) and Python 3x
* Python 3 is more object oriented than Python 2.
* Python is an interpreted language, so there is no need to compile them
* Python directly source code er maddhom e run kore; Source code to intermediate bytecode e conver hoy and erpor python interpreter er maddhom e computer er native language poriBortito hoy
* A programming language is the language used to create programs.
* print() ➖ it is a built-in function in python to display things on the screen
* A built-in function is a function which is predefined and can be used directly.
* Comments are the code which is ignored by the python interpreter.
* # By this hash Python declares a single line comment.
* Indentation : indentation in python refers to the spaces and tabs that are used at the beginning of the statement and the statement with the same indentation belongs to the same group called a suit.

**Variables**

* Variables act as containers to store variables.(String, int, float, boolean)
* Variable names are case-sensitive (age, Age and AGE are three different variables)
* Python automatically identifies the type based on values and we do not need to specify the type manually.
* Python allows to assign values to multiple variables in one line, like:

x, y, z = "Orange", "Banana", "Cherry" here, x = Orange; y = Banana; z = Cherry

And it can assign the same value to multiple variables in one line,

x = y = z = "Orange" here every variable has the same value.

* If you have a collection of values in a list, tuple etc. Python allows you to extract the values into variables. This is called unpacking,

fruits = ["apple", "banana", "cherry"]

x, y, z = fruits

Here, x = apple y = banana and z = cherry ; means, values have been stored in variables.

**Global and Local Variables**

* Variables that are created outside of a function are known as global variables.Global variables can be used by everyone, both inside of functions and outside.
* Normally, when you create a variable inside a function, that variable is local, and can only be used inside that function.
* To create a global variable inside a function, you can use the global keyword.

x = "awesome"

def myfunc():

global x

x = "fantastic"

myfunc()

print("Python is " + x)

Here,by writing global x we are calling the x = “awesome” inside the function and then changing the value to fantastic, now, globally the value has changed. So output is Python is fantastic

# Python Data Types

* Python data types are

Text Type: str examples x = str("Hello World")

Numeric Types: int, float, complex examples x = int(20, x = float(20.5, x = complex(1j)

Sequence Types: list, tuple, range examples

List:if specifies then

x = list(("apple", "banana", "cherry"))

Or normally

x = ["apple", "banana", "cherry"]

tuple:if specifies then

x = tuple(("apple", "banana", "cherry"))

Or normally

x = ("apple", "banana", "cherry")

x = range(6) range is always same to look

Mapping Type: dict examples

dict:if specifies then

x = dict(name="John", age=36)

Or normally

x = {"name" : "John", "age" : 36}

Set Types: set, frozenset examples

if specifies then

x = set(("apple", "banana", "cherry"))

Or normally

x = {"apple", "banana", "cherry"}

x = frozenset(("apple", "banana", "cherry"))

Frozenset is always same to look

Boolean Type: bool examples x = bool(5)’if specifies nor this is like’; x = True

Binary Types: bytes, bytearray, memoryview examples

bytes:if specifies then x = bytes(5) Or normally x = b"Hello"

bytearray:is always same to look

x = bytearray(5)

Memoryview: memoryview is always same to look

x = memoryview(bytes(5))

None Type: NoneType examples

type() using this function we can find the type.

* Int, or integer, is a whole number, positive or negative, without decimals, of unlimited length.
* Float, or "floating point number" is a number, positive or negative, containing one or more decimals.
* Complex numbers are written with a "j" as the imaginary part: & we cannot convert complex numbers into another number type.
* Python does not have a random() function to make a random number, but Python has a built-in module called random that can be used to make random numbers.Import the random module, and display a random number between 1 and 9

examples

import random

print(random.randrange(1, 10))

* We can assign a multiline string to a variable by using three quotes.
* Get the character at First position

a = "Hello, World!"

print(a[0]) #this is how strings First character is prints

# Python Functions

String Functions

* Python has four types of functions. Which are : Built-in, recursion, lambda, and user-defined functions are Python’s four types of functions.
* **int() -** constructs an integer number from an integer literal, a float literal (by removing all decimals), or a string literal (providing the string represents a whole number)

x = int(1) # x will be 1

y = int(2.8) # y will be 2

z = int("3") # z will be 3

* **float() -** constructs a float number from an integer literal, a float literal or a string literal (providing the string represents a float or an integer)

x = float(1) # x will be 1.0

y = float(2.8) # y will be 2.8

z = float("3") # z will be 3.0

w = float("4.2") # w will be 4.2

* **str()** - constructs a string from a wide variety of data types, including strings, integer literals and float literals.

x = str("s1") # x will be 's1'

y = str(2) # y will be '2'

z = str(3.0) # z will be '3.0'

* **input()** ➖ The input() function allows user input. input(“Enter your username”) & it’s data type is String, so if needed to take numeric value from user then type: varName = int(input(“Enter your age”)) or float(input(“Enter your age”)) , here the python interpreter will take only integer input type.
* Both upper() and lower() functions are built-in functions for string handling in Python. For a given string, upper() converts the string to all uppercase, while lower() converts the string to all lowercase.
* **capitalize()** Converts the first character to uppercase
* **casefold() -** Converts string into lower case
* **len()** - The len() function returns the length of a string  **print(len(a))**
* The **strip()** method removes any whitespace from the beginning or the end.

a = " Hello, World! "

**print(a.strip())** # returns "Hello, World!"

* To check if a certain phrase or character is present in a string, we can use the keyword **in** examples:

txt = "The best things in life are free!"

**if "free" in txt:**

print("Yes, 'free' is present.")

* To check if a certain phrase or character is NOT present in a string, we can use the keyword **not in**. examples

txt = "The best things in life are free!"

if "expensive" not in txt:

print("No, 'expensive' is NOT present.")

* The **replace()** method replaces a string with another string.

a = "Hello, World!"

print(a.replace("H", "J"))

* The **split()** method returns a list where the text between the specified separator becomes the list items. method splits the string into substrings if it finds instances of the separator.

a = "Hello, World!"

print(a.split(",")) # returns ['Hello', ' World!']

* **find() & index() are same:** returns the specified argument or the value is where or the position index number or can say: **Searches the string for a specified value and returns the position of where it was found**

practiceFind = cow.find("cow") and then, print(practiceFind)

practiceIndex = cow.index("cow") and then print the variable.

* **isalnum()-** Returns True if all characters in the string are alphanumeric || works for only alphabet not for hole string

userNameTxt = "noman.swe130"

alfaNumericTest = userNameTxt.isalnum()

print("alfaNumericTest", alfaNumericTest)

* **isnumeric()** - Returns True if all characters in the string are numeric; digitsTxt = "232323"

isNumericTest = digitsTxt.isnumeric()

print(isNumericTest)

* **isdecimal() -** Returns True if all characters in the string are decimals

digitsTxt = "232323"

isDecimalTest = digitsTxt.isdecimal()

print(isDecimalTest)

* The **isprintable()** method returns True if all the characters are printable, otherwise False.
* Returns true if the string ends with the specified value.

txt = "Hello, welcome to my world." xEndsWith = txt.endswith("world.")

* The **isidentifier()** method returns **True** if the string is a valid identifier, otherwise False.

A string is considered a valid identifier if it only contains alphanumeric letters (a-z) and (0-9), or underscores (\_). A valid identifier cannot start with a number, or contain any spaces.