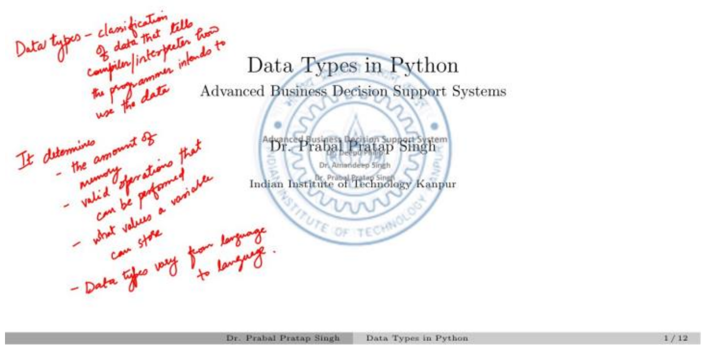
Advanced Business Decision Support Systems
Professor Deepu Philip
Department of Industrial Engineering and Management Engineering
Indian Institute of Technology, Kanpur
Professor Amandeep Singh
Imagineering Laboratory
Dr. Prabal Pratap Singh
Indian Institute of Technology, Kanpur

Lecture 28
Data Types of Python (Part 1 of 2)

Hello everyone, I welcome you all to the lecture series on Advanced Business Decision Support Systems and I am Prabal pratap Singh from IIT Kanpur. We are discussing about the implementation aspects of decision support systems using Python. So, as you all know, till now we have covered the elements and classifications of programming languages and discussed what is Python programming language. Next, we also created our development environment using the Python executable file downloaded from its official site and we created our development environment on Windows.



Data types - classification of data that tells compiler/interpreter how the programmer intends to use the data

It determines
- the amount of memory
- valid operations that can be performed
- what values a variable can store
- Data types vary from language to language.

Data Types in Python

Advanced Business Decision Support Systems

Dr. Prabal Pratap Singh

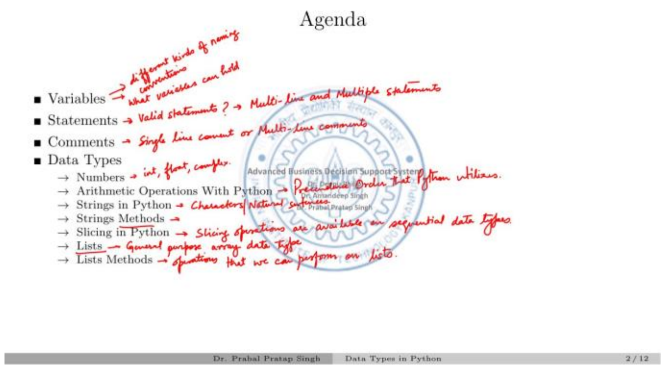
Indian Institute of Technology Kanpur

Dr. Prabal Pratap Singh | Data Types in Python | 1 / 12

So, today we will be discussing the data types in Python. So, we already discussed few things about Data Types. So, Data Types are classification of data that tells compiler or interpreter, how the programmer intends to use the data.

So, data type also it determines the amount of memory that will be used by declaring a particular Data Type, the valid operations that can be performed and what values a variable can store. So, as we have seen there are different kind of languages. So, data types vary from language to language.

So, a particular kind of data type that Python uses may not be available in other kinds of programming languages. So, let us see what are the different kinds of data types we have in Python programming language.



Agenda

- Variables → different kinds of naming conventions → what variables can hold
- Statements → Valid statements ? → Multi-line and Multiple statements
- Comments → Single line comment or Multi-line comments
- Data Types
  - → Numbers → int, float, complex.
  - → Arithmetic Operations With Python → Precedence Order that Python utilizes.
  - → Strings in Python → Characters/Natural sentences
  - → Strings Methods →
  - → Slicing in Python → Slicing operations are available on sequential data types.
  - → Lists → General purpose array data type
  - → Lists Methods → operations that we can perform on lists.

Advanced Business Decision Support Systems
Dr. Amandeep Singh
Dr. Prabal Pratap Singh

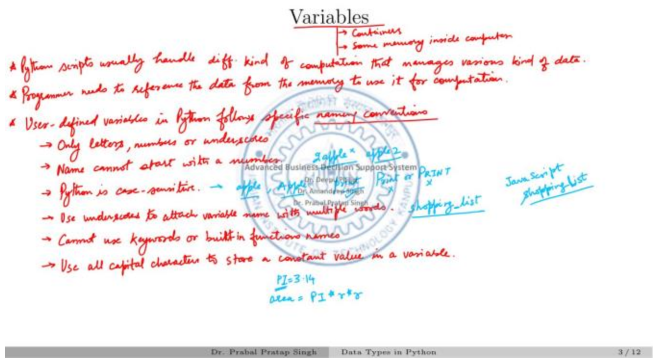
Dr. Prabal Pratap Singh | Data Types in Python | 2 / 12

So, the agenda for today is before getting to know the different data types, we will first see, what are the different kinds of Variables. Next, we will see what are the Valid Statements in Python and what are multi line and multiple statements. Next, we will see, how can we use the Comments inside our code base and what are the different kinds of comments like, Single line comment or Multi line comments.

After that, we will switch to the different kinds of data types like numbers, this includes integer type number, float type number or complex numbers. Then, we will see, what are the different kinds of arithmetic operations that are available on these numbers and the precedence order that Python utilizes. After that, we will move to another kind of data type which is the most common that is the String data types and these are those data types that contains character values or the natural language that we use, Characters/Natural Sentences. Then, we will see, what are the different kinds of operations that we can perform on these strings inside these string methods. After that, we will see Slicing in Python.

Slicing, there are different kinds of data types, but some of them are sequential data types. So, Slicing operations are available on sequential data types. The first and foremost is string which is also a sequential data type. So, we will start looking at slicing using the strings. After that, we will move to a general purpose array data type known as Lists.

These kind of data types can hold any kind of values inside it and it can be multiple values not just a single value. And, after that, again we will see, what are the different kinds of operations that we can perform on list.



Variables

→ Containers
→ Some memory inside computer

* Python scripts usually handle diff. kind of computation that manages various kind of data.
* Programmer needs to reference the data from the memory to use it for computation.
* User-defined variables in Python follows specific naming conventions
  - → Only letters, numbers or underscores
  - → Name cannot start with a number. 2apple × apple2
  - → Python is case-sensitive. → apple, Apple print Print or PRINT ×
  - → Use underscores to attach variable name with multiple words. shopping_list JavaScript shoppingList
  - → Cannot use keywords or built-in functions names
  - → Use all capital characters to store a constant value in a variable.

PI=3.14
area = PI*r*r

Dr. Prabal Pratap Singh | Data Types in Python | 3 / 12

So, let us start with variable. Variables are nothing but kind of containers and they assign some memory inside computer. So, Python scripts usually handle different kind of computations that manages various kind of data.

Now, we have seen earlier that, we can use Python interpreter, but what interpreter does is, it can always use the in process memory and it will fade away once you close the interpreter. But a variable can store a value during the process until it completes and until we explicitly do not delete that variable. So, that is why, programmer often needs to reference the data from the memory to use it for computation. So, user defined variables in Python follow specific naming conventions like. So, these naming conventions are different and they differ by each programming language.

So, the first naming convention for any kind of variable in Python is only letters or numbers or _s. Only these things can be used to define a Variable. The next is name cannot start with a number. So, let us say, if somebody is creating a variable and they write at the interpreter like to apple. So, this will not be a correct naming convention.

So, the interpreter will throw an error saying that, this is not the correct syntax right, but if somebody says, apple to without spaces in between. So, this will be a correct variable name. Next naming convention is Python is case sensitive. So, what does that mean is, if you have a variable apple and another variable as Apple. So, these two variables refer to two different things in memory for Python.

So, they are two different kinds of variables they are not same. Similarly, there are some predefined keywords like print statement, which we seen in the last lecture. So, there is this print keyword which is a predefined function. So, what it does it prints anything inside it to the screen? Now, if you instead of writing it as a in small cases, if you write it in like Print or PRINT. So, these things will not work.

So, these are not the correct usages, but this is the right one. So, this is how case sensitivity works. Next convention is, only use _s to attach variable names with multiple words. So, what happens is, let us say, you are doing some large, you are creating some large code base and you are defining many variables. So, if there is a variable called shopping list.

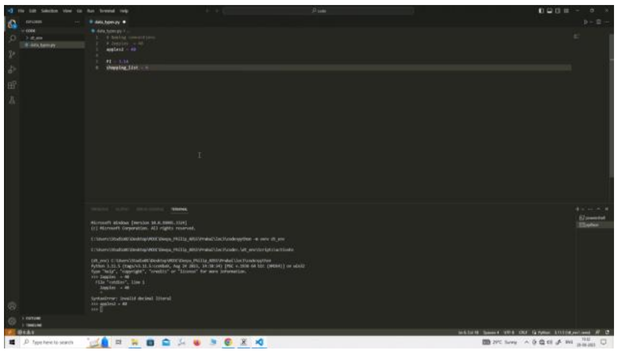
So, these are two different words. Now, how if you just write it with a blank space in between them, it will not be a correct usage of a variable name. So, to combine this and use it as a variable name, you need to use some kind of character in between them instead of using just space. So, the convention in python programming language is to use _s. In other programming languages like JavaScript, they use camel cases.

So, what happens is, in JavaScript, they use shopping list. So, this is the capital L. So, any variable which combines more than two words, will use the first character of the next word as a capital letter. So, this is camel case. Some of the old packages in python also uses, but now, most of all around every packages used today the _ to create these kinds of variable names.

The last convention is that, we cannot use keywords or built in functions names. So, this is the same thing which I told you earlier regarding print open. So, one additional convention is which usually programmers uses that, if you are using some variable to store a constant value then, use all capital characters to store a constant value in a variable. So, your program will still work if you do not follow this last line, but it is just community convention that people uses. So, for example, if you are creating a program to calculate the area of a circle.

So, we know that the value of pi is 3.14. So, this is constant this remains the same. So, we can use it in a variable pi. So, I am using the all caps like, P is capital L is capital.

Now, area is pi into radius into radius. So, here there are three different variables area pi, pi constant, which is constant then next is r radius. So, this is pi r square. So, this is how we try to follow this convention here.



So, let us see, all these rules inside Visual Studio code and if you are still struggling with your development environment then, you can ask your questions in the forums TAs will help you.

So, we will create a new directory here, name code and we will open it in our Visual Studio code. Now, this is a new directory. So, it will ask again the same question. Click yes and here create a new file delta times . py. Now we can open our terminal and it is in power shell.

So, we can use command prompt. After that, you can see that, we have not yet initialized our environment. So, we can write:$python - m\ venv\ dt\_env$. So, I am again creating a new development environment for our lecture today. So, usually, Visual Studio code ask you that, they noticed a new environment has been created, do you want to select it for the workspace folder? Now, the line just below it is called the Status bar.

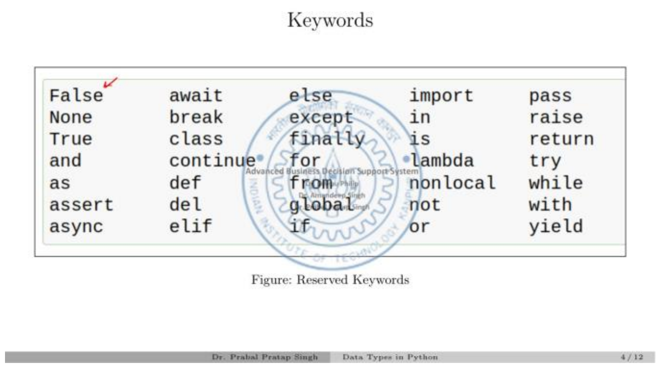
So, it is just showing Python. So, the best way to activate an environment for the whole visual studio code is to open the command palette using control + shift + P and then right select interpreter. So, it will ask to select an interpreter, here you can enter the interpreter path or you can select it by browsing it. So, here dtn scripts and you can select this python.exe. So, now you can see here, where I am hovering my mouse that the current python version is dtn for choosing the vn module, but still terminal needs to use that variable using the command. So, we write again, $.\backslash dt\_env\backslash Scipts\backslash activate$. Now, you can see that, we have activated our environment dtn here.

One more thing, we will be going to do today is to create one more shortcut. Open your command palette, open keyboard shortcuts and here, search for run python file in terminal. So, this comes, now you can see that, there are no key bindings for this shortcut. So, we can create here like alt + shift + N. So, whenever we will use this key binding, it will run our python file in the terminal.

So, you can see that, we have just updated our keyboard shortcut here, now you can close it. Now, we are ready for the today's data type lecture. So, I am writing the first command as naming conventions. So, the first naming convention was that we cannot use a number for a variable declaration.

So, let us say, 2f = 40. So, you can see here that the visual studio code itself is highlighting it that there is something wrong with it. Statement must be separated by new lines or semicolons, but if you just copy this in the interpreter. So, it is showing that invalid decimal literal. So, this is the syntax error here, but if you write the opposite a pulse 2 = 40, then this works. So, you can comment it by using the key stroke control + forward slash.

So, the correct thing is a pulse 2 = 40. The next convention was to use a constant here like pi = 3.14. So, you can see here. So, this is how you define a constant variable and to join two words together. So, you can use an _ like this. So, let us move forward by saving this file.



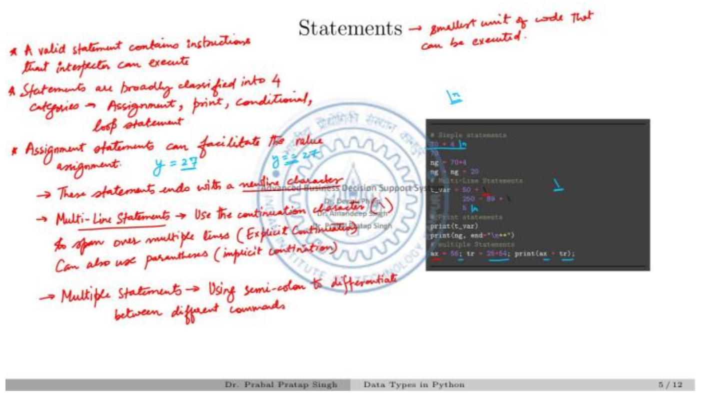
Keywords

False	await	else	import	pass
None	break	except	in	raise
True	class	finally	is	return
and	continue	for	lambda	try
as	def	from	nonlocal	while
assert	del	global	not	with
async	elif	if	or	yield

Figure: Reserved Keywords

Dr. Prabal Pratap Singh Data Types in Python 4 / 12

These are few of the keywords that python predefines and we should not use these words while using, while creating some new variable. So, let us say you should not use False. So, this is a predefined keyword. These all are like that. So, just keep it in mind.



Statements → smallest unit of code that can be executed.

* A valid statement contains instructions that interpreter can execute
* Statements are broadly classified into 4 categories → Assignment, print, conditional, loop statement
* Assignment statements can facilitate the value assignment. $y = 27$ $y == 27$
  → These statements ends with a newline character
  → Multi-Line Statements → Use the continuation character (\) to span over multiple lines (Explicit Continuation) Can also use parantheses (implicit continuation)
  → Multiple Statements → Using semi-colon to differentiate between different commands

\n

```
# Simple statements
70 + 4 \n
74
ng = 70+4
ng = ng + 20
# Multi-Line Statements
t_var = 50 + \
        250 - 89 + \
        5 \n
# Print statements
print(t_var)
print(ng, end="\n**")
# multiple Statements
ax = 56; tr = 28+54; print(ax + tr);
```

Now, let us move forward to what are the Statements. So, statements are the smallest unit of code that can be executed. So, we always try to write valid statements. So, what is a valid statement? These statements are broadly classified into four categories which are assignment statements, current statements, conditionals and loop statements.

So, while writing code, we can only use the combination of these kind of statements to create our code base. So, what assignment statements can do is they can facilitate the value assignment. So, we usually try to store some kind of value into a variable. So, let us say we are writing a valid statement as y = 27. So, what this statement is telling to the interpreter is that y is a variable, a container that can be referred to some particular location in the memory and it should store this value 27 in it.

So, in mathematics, the equal to is for different usage. n the python programming, the equal to stands for something different. So, here if you want to see whether y = 27 or not which usually we do in mathematical notations is. So, to do that, we need to write y == 27. So, this is the different kind of operator that we will check whether y == 27 or not.

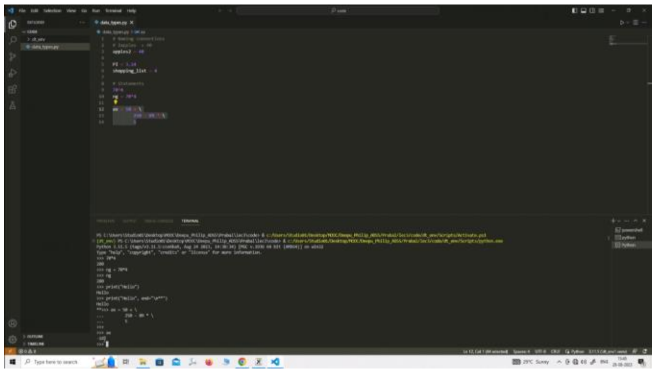
But here, while assigning anything, we only use one = different languages, use different kinds of operators to assign things python uses = operator. So, these statements ends with a new line character. So, why do we do that? Because interpreter needs to know where a particular statement gets completed. So, it is defined that whenever a new line character appears, the interpreter thinks that, the one single statement is over. So, you can see here that I have written this 17 to 4.

So, this is a valid statement and here I used an enter. So, that I can go to the next line. So, that new line character is invisible here, but the interpreter will see that the new line character is / n and it will find this / n here and then it will think that this is a complete statement which the programmer has coded. So, it will interpret it and see whether it is a valid statement or not and then, it will execute otherwise, it will throw syntax error or some kind of different errors right.

So, the next thing is Multi-Line statements. What is the use of Multi-Line statements is sometimes, what happens is, when we are writing our code in python, there are lines get overly long. So, horizontally it will move to beyond 80 characters or 100 or 150 characters. So, it will be cumbersome for a code reader to read it and it is not a readable code anymore. So, to get out of that, the python defines that if you use this back / character, then this back / character. Then what happens is, the interpreter thinks that the statement has not yet completed and it will look for the next line to see, what it contains and if it again contains a back / character, then it will go to the next line as well until unless, it will find that invisible / n character.

So, here T var (variable) contains the computation of 50 + 250 minus 89 into 5 and the value of it will be stored in T var not just 50 +. So, this is where Multi-Line statement come. So, here we use the continuation character back / to span over multiple lines. So, this kind of continuation is explicit continuation. We can also use parenthesis or small brackets which is known as implicit continuation.

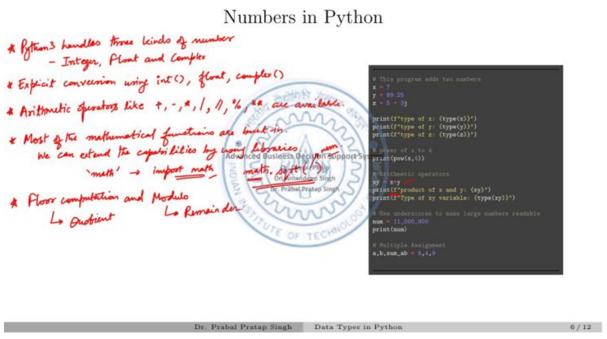
The last thing with statement is Multiple statements. So, the earlier one is Multi-Line statements, now we are looking at Multiple statements. So, whenever interpreter finds a / n character, it will see that the line completed, but we can use different kinds of statements on a single line by using semicolon to differentiate between different commands. So, here in the last line, we can see that I have initialized a variable Ax goes to 56. Then, I used a semi-colon, then I again initialized a different variable Tr and performed inline computation with 25 + 54 and then I printed Ax + Tr. So, we can also see this on our environment using visual studio code.



So, this is the statement section. So, let us say, we write 70 * 4 and we just use our shortcut key which we defined as L + shift + N. It will run the complete file in the interpreter to run only this single line, we can use shift + enter. So, it started a different interpreter and then ran this 70 * 4 line. The next line which we can see is we can initialize a different variable ng = 70 * 4 and you can save this file then run this file in the interpreter.

So, ng = 70 to 4 and now we can see whether ng stores 280 or not. So, it stores 280 here. Now, one more thing that I wanted to tell you about print statements is, if we write $print$("Hello") and just execute this command, it will write the Hello on the screen and then on the next line, interpreter comes. But why the next line because the print command has a default of /env at the end. So, we can change that default by using an end command, end argument to the print function and we can write here our own string like $print("Hello",\ end$ "\n**")

So, you can see that now it is printing Hello and on the next line, it is printing **>>> then interpreter line starts after the two stars. So, this is how this print function can be modified as per your liking. So, to see the multi-line statements we can write here. So, if we run this complete. Then, see there are no errors here and we can see that ax variable contains our computed value of minus 145.

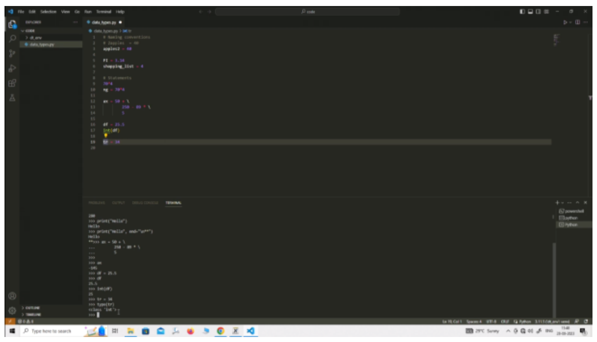


Numbers in Python

* Python3 handles three kinds of number
  - Integer, Float and Complex
* Explicit conversion using int(), float, complex()
* Arithmetic operators like +, -, *, /, //, %, ** are available.
* Most of the mathematical functions are built-in.
  We can extend the capabilities by using libraries.
  'math' → import math   math.sqrt(num)
* Floor computation and Modulo
  ↳ Quotient   ↳ Remainder

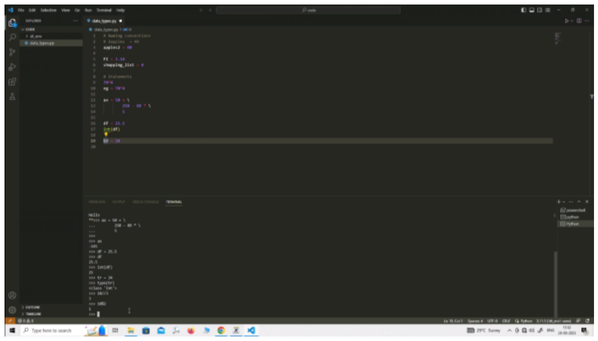
```
# This program adds two numbers
x = 7
y = 89.25
z = 5 + 3j

print(f"type of x: {type(x)}")
print(f"type of y: {type(y)}")
print(f"type of z: {type(z)}")

# power of x to 4
print(pow(x,4))

# Arithmetic operators
xy = x*y
print(f"product of x and y: {xy}")
print(f"Type of xy variable: {type(xy)}")

# Use underscores to make large numbers readable
num = 11_000_000
print(num)

# Multiple Assignment
a,b,sum_ab = 5,4,9
```

Dr. Prabal Pratap Singh   Data Types in Python   6 / 12



```
# Naming conventions
# 2apples = 40
apples2 = 40

PI = 3.14
shopping_list = 4

# Statements
70*4
ng = 70*4

ax = 50 + \
        250 - 89 * \
        5

df = 25.5
int(df)

tr = 34
```

```
280
>>> print("Hello")
Hello
>>> print("Hello", end="\n**")
Hello
**>>> ax = 50 + \
...         250 - 89 * \
...         5
>>>
>>> ax
-145
>>> df = 25.5
>>> df
25.5
>>> int(df)
25
>>> tr = 34
>>> type(tr)
<class 'int'>
>>>
```



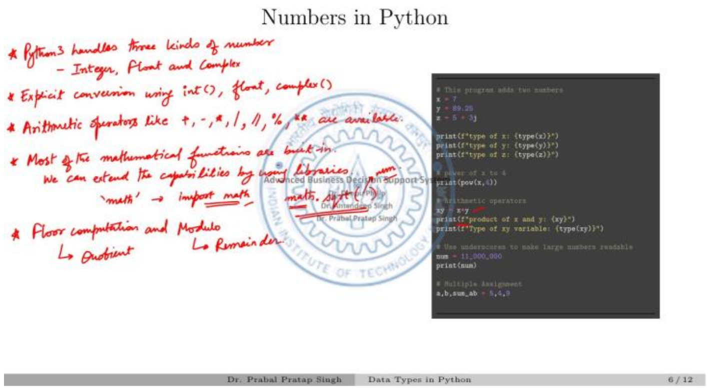
```
Hello
**>>> ax = 50 + \
...         250 - 89 * \
...         5
>>>
>>> ax
-145
>>> df = 25.5
>>> df
25.5
>>> int(df)
25
>>> tr = 34
>>> type(tr)
<class 'int'>
>>> 10//3
3
>>> 10%3
1
>>>
```

So, let us move to the next data type which is Numbers. So, python 3 handles three kinds of numbers which are integer, float and complex numbers. So, we can use explicit conversions from one kind of number data type to another using predefined functions like int() or float or complex().

You can see here in our code like, let us say, we have a variable df = 25.5. So, we ran it in our interpreter and we can see that df now contains 25.5.

Now, this is a float value, you all know that it has a decimal, but what if we want to change it to a integer type? So, we can use the predefined int function and write df inside it as an argument and run it. So, now it is showing that it has converted from 25.5 to 25.

Also, let us define another variable tr = 34. So, this is an integer type, but you can also check it with python whether it is an integer or not. So, to do that, we can use the inbuilt type function and write the variable name tr and it shows that this tr variable is of class int. So, I have already told you that everything is an object in python and objects are based on the blueprint that is class. So, here it is using the class blueprint of int that was defined by the developers of python.

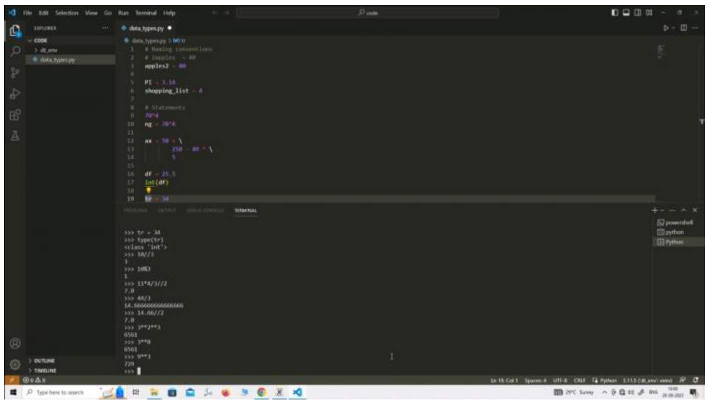


Numbers in Python

* Python3 handles three kinds of number
  - Integer, Float and Complex
* Explicit conversion using int(), float, complex()
* Arithmetic operators like +, -, *, /, //, %, ** are available.
* Most of the mathematical functions are built-in.
  We can extend the capabilities by using libraries.
  'math' → import math   math.sqrt(num)
* Floor computation and Modulo
  ↳ Quotient   ↳ Remainder

```python
# This program adds two numbers
x = 7
y = 89.25
z = 5 + 3j

print(f"type of x: {type(x)}")
print(f"type of y: {type(y)}")
print(f"type of z: {type(z)}")

# power of x to 4
print(pow(x,4))

# Arithmetic operators
xy = x*y
print(f"product of x and y: {xy}")
print(f"Type of xy variable: {type(xy)}")

# Use underscores to make large numbers readable
num = 11_000_000
print(num)

# Multiple Assignment
a,b,sum_ab = 5,4,9
```

Dr. Prabal Pratap Singh   Data Types in Python   6 / 12



```python
# Naming conventions
# 1apples  = 40
apples2 = 40

PI = 3.14
shopping_list = 4

# Statements
70*4
ng = 70*6

ax = 50 + \
        250 - 89 * \
        5

df = 25.5
int(df)

tr = 34
```

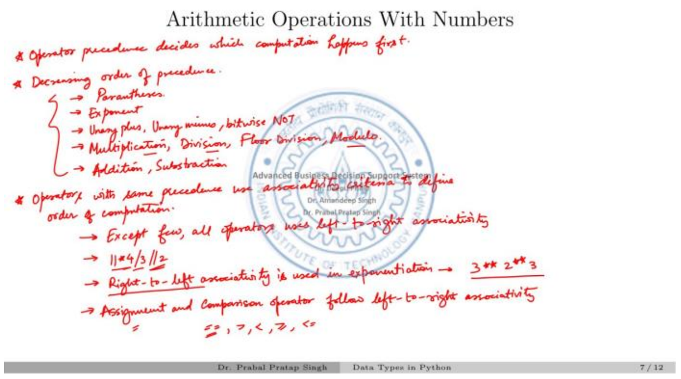
```
>>> tr = 34
>>> type(tr)
<class 'int'>
>>> 10//3
3
>>> 10%3
1
>>> 11*4/3//2
7.0
>>> 44/3
14.666666666666666
>>> 14.66//2
7.0
>>> 3**2**3
6561
>>> 3**8
6561
>>> 9**3
729
>>>
```

So, let us move forward and see that numbers can use the arithmetic operators like +, -, *, /, //, %, ** are available These are available in Python for any kind of computation. So, most of the operators and mathematical functions are built in, but we can extend the capabilities by using libraries.

So, standard installation of python comes with a pre-installed math library. So, we can use 'import math' statement to use this. So, if you try to find the square root of a particular number, then you cannot find the answer by just typing square root because there are no predefined functions for that, but if you write this statement and then use 'math.sqrt()' operator to use the functions or methods defined inside this library.

So, now you can use this valid statement to find the square root of a particular number any number. So, I have shown you here that arithmetic operators are we can write these as x * y and we can use these f strings to provide in line computation and then give the output inside a print statement. And, the last thing is regarding floor computation and modulo. So, what floor computations give you is the quotient and modulo will give you the remainder.

So, let us see the example of these two things. So, if I write 10 // 3. So, it is giving me 3. So, 3 times 3 is 9 and the remainder is 1. So, // give you the quotient. So, it is showing 3, but if we want the remainder of this computation, then we can write 10 % 3 and it will give you 1. So, this is how these two things work in Python.



Arithmetic Operations With Numbers

* Operator precedence decides which computation happens first.
* Decreasing order of precedence.
  - → Parantheses
  - → Exponent
  - → Unary plus, Unary minus, bitwise NOT
  - → Multiplication, Division, Floor Division, Modulo.
  - → Addition, Substraction
* Operators with same precedence use associativity criteria to define order of computation.
  - → Except few, all operators uses left-to-right associativity
  - → 11*4/3//2
  - → Right-to-left associativity is used in exponentiation → 3** 2** 3
  - → Assignment and Comparison operator follow left-to-right associativity
    =   ==, >, <, >=, <=

Dr. Prabal Pratap Singh   Data Types in Python   7 / 12

So, let us see how different kinds of arithmetic operations are performed because there are different multiple kinds of operators available. So, which one will be used first when a programmer is defining multiple operators in a single statement. So, this kind of operations we need to use the precedence order. So, operator precedence decides which computation happens first. So, we will write the decreasing order of precedence, where the highest precedence is given to Parentheses.

The next is Exponent, third precedence is given to unary operator plus, unary minus and bitwise not fourth precedence is given to Multiplication Division floor Division and modulo. The last precedence is Addition, Subtraction. There are other operators as well, but we will stick to these for this. Now, the problem arises like, if a particular statement contains Multiplication Division or modulo in a single statement. So, which one we will do first? So, to do that, we use operators with same precedence, use associativity criteria to define order of computation.

So, the thumb rule is except all operators uses left to right associativity. So, for this, we can use an example like 11 into 4 divided by 3 modulo floor Division 2. So, how will this be interpreted? So, we can see here that we have to use left to right and these precedence orders. So, there are no Parentheses, no Exponent, no unary operators, but we have Multiplication. So, the first thing is to do Multiplication, after that, we have Division. So, since we are moving from left to right we will first do Multiplication, then we will do the outcome of this Multiplication divided by 3.

After that, we will do the Floor Division. So, we can see this in our interpreter, how it is doing it. So, we can write 11 * 4 / 3 // 2. So, it should give 7. Why? Let us see, 11 * 4 is 44. 44 divided by 3 is 14.66 and this 14.66, Floor Division 2 is 7. So, this is how operator precedence works and if you use Parentheses somewhere here, then the first thing to compute is the Parentheses, but an exception here is right to left associativity is used in exponentiation. So, we can see it with an example again like, 3**2**3.

So, let us see, how this will work in our interpreter, 3 ** 2 ** 3. Now, it is giving the answer 6561 why because as we have seen that we need to first compute the right Exponent. So, what happens here is, we will first do 2 to the power 3 which is 8. So, if we write 3 to the power 8, it should give 6561. So, here we can confirm this that, we are using the right associativity in Python. Otherwise, if we if we have done this from left to right, then the first computation should be 3 ** 2, which is 9 and 9 ** 3, which is 729.

So, the answer will be very different, if we are using left to right or right to left. So, be sure that the during exponentiation, we will move from right to left and otherwise, we will move from left to right. Also, Assignment and Comparison operators. So, Comparison operators are like, ==, which will check whether the left hand value and right hand value are same or not or >, <, ≥, ≤ these are the comparison operators and assignment operators, we have already seen it. So, they also follow left to right associativity. So, these are some of the major data types that python contains and we will discuss further in the upcoming lectures.

So, thank you very much, I hope you are practicing all these things in your development environment as well, if you find any difficulties, please reach out us to in forums and we will help you out. Thank you.