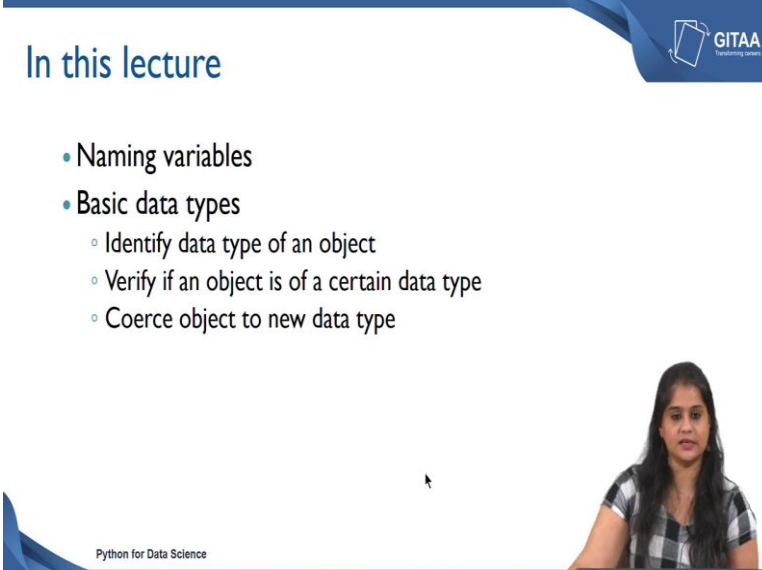


**Python for Data Science**  
**Prof. Ragnathan Rengasamy**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Madras**

**Lecture – 05**  
**Variables and Data Types**

(Refer Slide Time: 00:18)



The slide features a blue header with the text "In this lecture" and the GITAA logo. Below the header is a bulleted list of topics. In the bottom right corner, there is a video inset showing a woman with long dark hair wearing a black and white checkered shirt. The text "Python for Data Science" is visible in the bottom left corner of the slide area.

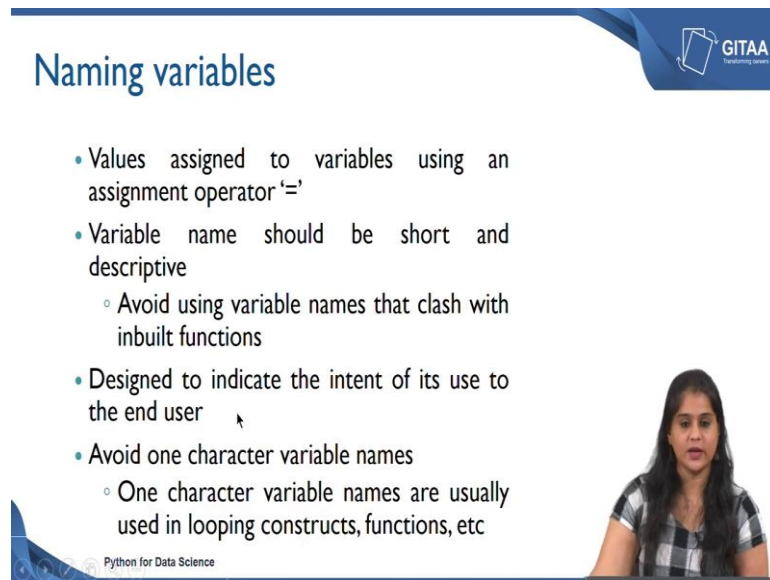
**In this lecture**

- Naming variables
- Basic data types
  - Identify data type of an object
  - Verify if an object is of a certain data type
  - Coerce object to new data type

Python for Data Science

Welcome to the lecture on Variables and Data Types. In this lecture, we are going to look at how name variables, some of the common rules and conventions in naming a variable. We are also going to look at some of the basic data types that we are going to use throughout this course and in Python. As a part of this task, we are going to look at how to identify a data type of an object, how to verify if an object is of a certain data type and how to coerce objects to a new data type.

(Refer Slide Time: 00:43)



The slide is titled "Naming variables" in a blue font. It features a list of four bullet points with sub-points. In the top right corner, there is a logo for "GITAA Traditional Growth". In the bottom left corner, there is a logo for "Python for Data Science". On the right side of the slide, there is a small video inset showing a woman with long dark hair wearing a black and white checkered shirt.

- Values assigned to variables using an assignment operator '='
- Variable name should be short and descriptive
  - Avoid using variable names that clash with inbuilt functions
- Designed to indicate the intent of its use to the end user
- Avoid one character variable names
  - One character variable names are usually used in looping constructs, functions, etc

So, let us begin with naming variables. So, values are assigned to variables using the assignment operator equal to(=). So, the variable name should be short and descriptive, and that is because there is an intent for creating the variable and it is supposed to convey an information hence it is better that it is short and descriptive. So, avoid using variable names that clash with inbuilt functions.

Like I earlier said, the variable names are designed to indicate the intent and purpose of its use to the end user. So, hence it is better to avoid one character variable names. So, one character variable names are usually used in iterations, and functions and looping constructs.

(Refer Slide Time: 01:23)


## Naming variables

- Variables can be named alphanumerically

Age=55   age=55   age2=55   Age2=55

- However the first letter must start with an alphabet (lowercase or uppercase)

```
In [3]: 2age=55
File "<ipython-input-3-00efac86f1ae>", line 1
      2age=55
         ^
SyntaxError: invalid syntax
```



Python for Data Science

So, variables can be named alpha numerically. So, if I have a variable called age whose value is 55, then I can either have the entire age in lower case or I can also begin with an upper case. You can also add a number 2 weight. So, let us say if I am creating another variable called age 2, then I can add the number after the alphabets. So, one thing that you would have noticed here is that the first letter should always begin with an alphabet, but however, if you were to begin with the number the compiler throws an error saying invalid syntax.


(Refer Slide Time: 02:01)

## Naming variables

- Other special character
  - Underscore ( \_ ) → Employee\_id=501
  - Use of any other special character will throw an error →

```
In [6]: Employee@id=501
File "<ipython-input-6-e2ec8cb31ebe>", line 1
      Employee@id=501
                ^
SyntaxError: can't assign to operator
```
  - Variable names should not begin or end with underscore even though both are allowed →

```
_age=55
age_=55
```



Python for Data Science

So, the only other special character that is allowed while naming a variable is underscore. Now, let us say if I want to create a variable that conveys the employee id, then I can separate the employee and id with an underscore. Now, underscore is the only other special character that is allowed.

Now, if you use any of the other special characters you will get an error that says cannot assign to the operator. So, though underscore is allowed, it is better to not begin or end with an underscore and that is a common unaccepted naming convention, though it is accepted by the compiler. So, if you begin or end with an underscore you are not likely to get an error. So, it is usually not a practice that is followed.

(Refer Slide Time: 02:46)

**Naming conventions**

- Commonly accepted case types
  - Camel (lower and upper)  
`ageEmp=45`    `AgeEmp=45`
  - Snake  
`age_emp=45`    `Age_emp=45`
  - Pascal  
`AgeEmp=45`

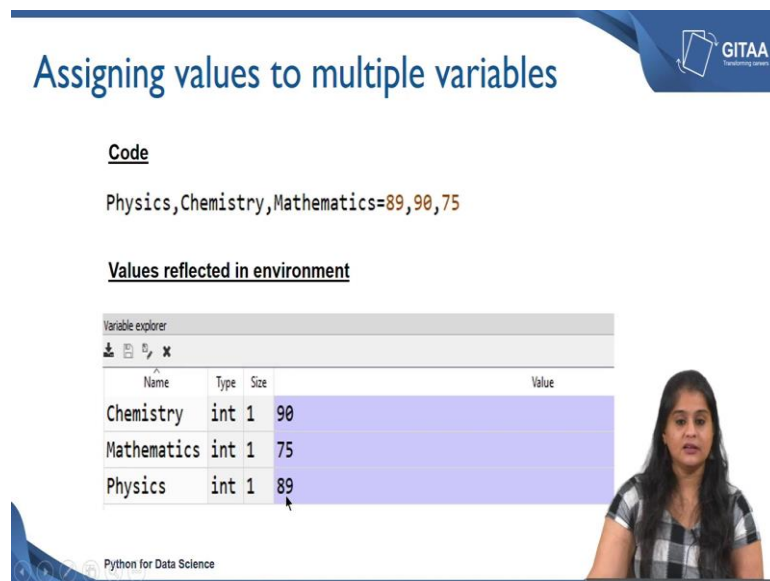
Python for Data Science

So, there are a few case types that is commonly accepted while naming variables, the first is the camel case. It can be lower or upper. First example, that you have here where age of the employees given with E in capital. Now, this is a lower camel case, so the example on the right is an example for the upper camel case. Now, in this example the letter 'a' in age of the employee begins with the capital letter.

So, the next case type is the snake case where I am separating age and emp by an underscore. So, an underscore can be used between two set of letters or between two letters and that is a snake case. The letter after the underscore should always be in lower case, but the first letter of the variable name can be in lower or upper case.

The next case type is Pascal; so, where I am again going to take AgeEmp. Now, here the first letter of age is in uppercase and the first letter of emp is also in uppercase. Now, these are the commonly used case types in Python. Now, the compilers is not going to throw an error if you name the variables wrongly, but these are convention that are in books and are accepted. However, there are other case types which use hyphen, but those case types will not be allowed in Python.

(Refer Slide Time: 04:06)



The slide features a blue header with the title "Assigning values to multiple variables" and the GITAA logo. Below the title, the code `Physics,Chemistry,Mathematics=89,90,75` is displayed. Underneath, the text "Values reflected in environment" is followed by a screenshot of a "Variable explorer" window. This window contains a table with three rows: Chemistry (value 90), Mathematics (value 75), and Physics (value 89). A woman is visible in the bottom right corner of the slide.

### Assigning values to multiple variables

**Code**

```
Physics,Chemistry,Mathematics=89,90,75
```

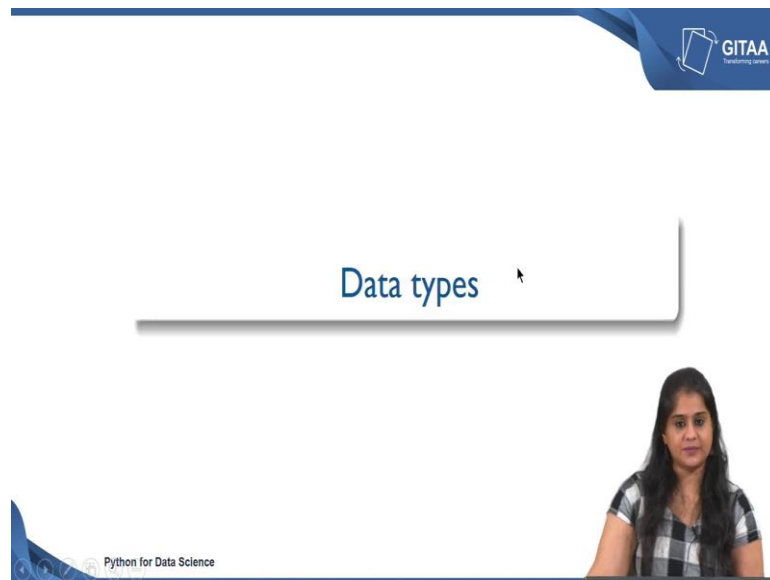
**Values reflected in environment**

Name	Type	Size	Value
Chemistry	int	1	90
Mathematics	int	1	75
Physics	int	1	89

Python for Data Science

So, if you want to assign multiple values to a variable you can create all the variable at once and sequentially assign the value. Now, if you run this command you will see that the values of variables chemistry mathematics and physics have been assigned accordingly.

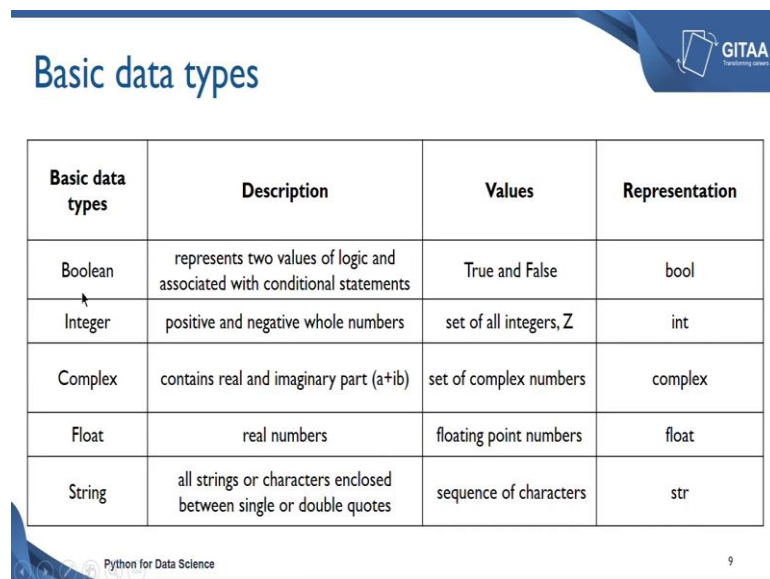
(Refer Slide Time: 04:24)



The slide features a blue header with the GITAA logo (GITAA Traditional Growth) and a white box containing the text "Data types". A video feed of a presenter is visible in the bottom right corner. The footer includes the text "Python for Data Science".

So, now let us look at the commonly used data types in Python.

(Refer Slide Time: 04:27)



The slide features a blue header with the GITAA logo and the title "Basic data types". Below the title is a table with four columns: "Basic data types", "Description", "Values", and "Representation". The table lists five data types: Boolean, Integer, Complex, Float, and String. A video feed of a presenter is visible in the bottom right corner. The footer includes the text "Python for Data Science" and the number "9".

Basic data types	Description	Values	Representation
Boolean	represents two values of logic and associated with conditional statements	True and False	bool
Integer	positive and negative whole numbers	set of all integers, $Z$	int
Complex	contains real and imaginary part ( $a+ib$ )	set of complex numbers	complex
Float	real numbers	floating point numbers	float
String	all strings or characters enclosed between single or double quotes	sequence of characters	str

So, the basic data types are Boolean which represents two values of logic and are associated with the conditional statement. So, the output values that you would get, when you use a Boolean data type is true or false, and it is represented as bool. The next data type is integer. It consists of set of all integers which are positive or negative whole numbers. It is represented by the letters int. The next data type is complex which

contains real and imaginary part. So, any expression of the form  $a+ib$  is of a complex data type.

It consists of all complex numbers and it is represented by `complex`. Now, float data type consists of all real numbers which are of floating point numbers, it is represented by `float`. String data type consists of all strings and characters, so anything that is enclosed between single or double quotes is treated as a string data type. The value that is enclosed between the quotes can be a number, a special character, alphabets anything. So, anything that is contained within quotes is treated as a string data type. It is represented by the letters `str`.

(Refer Slide Time: 05:38)

The slide features a blue header with the title "Statistically vs dynamically typed language" and the GITAA logo. Below the title, there are two main bullet points, each with three sub-bullets. A small video inset in the bottom right corner shows a woman with long dark hair wearing a black and white checkered shirt. The bottom left corner of the slide has the text "Python for Data Science".

- Statistically typed language
  - Type of variable is known at compile time
  - Type of variables declared upfront
  - Eg- Java, C, C++
- Dynamically typed language
  - Type of variable known at run time
  - Variable type need not be declared
  - Eg- Python, PHP

Now, before we go ahead with data type operations, it is important to know the difference between statistically and a dynamically typed language. So, a statistically typed language is the one, where the type of the variable is known at the compile time and you also have to declare the data type of the variables upfront. So, examples of such programming languages are C, C++ and Java.

So, contrary to this a dynamically typed language is the one where the data type need not be declared upfront, the type of the variable is known only at the run time. So, whenever you declare a variable and you assign a value of it; the moment you run that specific line, the data type of the variable is known. Examples of such languages are Python, PHP. So, Python that we are using here is a dynamically typed language.

(Refer Slide Time: 06:29)

## Identifying object data type

- Find data type of object using
- Syntax: `type(object)`

```
Employee_name="Ram"  
Age=55  
Height=150.6
```

Checking the data type of an object

```
In [10]: type(Employee_name)  
Out[10]: str  
  
In [11]: type(Age)  
Out[11]: int  
  
In [12]: type(Height)  
Out[12]: float
```

Python for Data Science

GITAA  
Traditional Growth

11

So, let's go further and see how to identify the data type of an object. Now, to find the data type of an object you will be using the function `type` and give the object as an input. Now, an object can be a variable, can be any of the data structures, it can be array or anything. In this case in particular, we are just going to look at how to find the data type of variables. Now, let us take a small example here. I have 3 variables here, the first is *Employee\_name* which is the name of the employee and the value is "Ram".

Next is age of the employee presented by the variable name *Age*, the value for which is 55. The third variable is height, which is the height of the employee whose value is 150.6. Now, if you want to check the data type you can give type of *Employee\_name*, now *Employee\_name* is a string because it is enclosed between double quotes. If you give type of *Age*, *Age* is an integer and hence the output is `int` and if you give type of *Height* which is a floating point number the data type is `float`.



(Refer Slide Time: 07:40)

## Verifying object data type

- Verify if an object is of a certain data type
- Syntax: `type(object) is datatype`

```
Employee_name="Ram"
Age=55
Height=150.6
```

Verifying the data type of an object

```
In [13]: type(Height) is int
Out[13]: False

In [14]: type(Age) is float
Out[14]: False

In [15]: type(Employee_name) is str
Out[15]: True
```

Python for Data Science 12

So, now, let us see how to verify the data type of an object. Now, if you want to verify if an object belongs to a certain data type, then you basically give type of the object in this case it is a variable name, followed by the keyword `is` and the data type name. So, this data type will basically be the representation of the data type; `int` is for integer, `str` is for string so on and so forth. Now, I am going to use the same example that we have used earlier. Now, here I would like to verify if height is of integer data type. So, I am giving type of height followed by `is` and the keyword `int` which is the abbreviation for integer data type.

So, one important thing to keep in mind while verifying the data type is that in this case the output is just going to be Boolean. So, the output is going to be true or false. We are actually checking given a variable does it belong to a desired data type or not. We are not trying to assign it or change it, but what we are just trying to do is just cross verification. So, hence the output for this is either going to be true or false. So, I want to know if Age belongs to a float data type, which is false, because Age is actually an integer. Now, in this case I want to check if `Employee_name` is a string. Yes, the output is true because it is a string.

(Refer Slide Time: 09:02)

## Coercing object to new data type

- Convert the data type of an object to another
- Syntax: `datatype(object)`
- Changes can be stored in same variable or in different variable

```
Employee_name="Ram"
Age=55
Height=150.6
```

Coercing the data type of an object

```
In [16]: type(Height)
Out[16]: float

In [17]: ht=int(Height)

In [18]: type(ht)
Out[18]: int

In [19]: Height=int(Height)

In [20]: type(Height)
Out[20]: int
```

So, till now we have seen how to find the data type of an object. We have also seen how to verify if an object has a desired data type. Now, we are going to look at how to coerce object to new a data types. Now, if I want to convert the data type of an object to another, I will be using the data type, you need to replace data type with the abbreviation used for each of the data types and you give the object as the input here. Now, all the changes that have been made to the variables can be stored in the same variable or different variable.

Now, in this case height is of a float data type, let us say I want to convert it to an integer data type. So, I say `int(Height)` which converts it into an integer data type, but if I want these changes to be reflected in my environment, I have to store it on to a variable. Now, the variable can have the same name or a different name. So, in this case I am converting height to an integer data type and I am storing it onto a new variable called `ht`. Now, if I get the type of `ht` the output is `int`, which means height has been converted to an integer data type.

Now, if I want to reflect the changes on the same variable name, I will just say `Height=int(Height)`. So, earlier height was float, now once the operation is done if you check the type of height again the output that you will be getting is `int`. So, this is how you would coerce existing objects to new a data types.

(Refer Slide Time: 10:37)

## Coercing object to new data type

- Only few coercions are accepted
- Consider the variable '*Salary\_tier*' which is of string data type
- '*Salary\_tier*' contains an integer enclosed between single quotes

```
Salary_tier='1'
```

Coercing the data type of an object

```
In [20]: type(Salary_tier)
Out[20]: str

In [21]: Salary_tier=int(Salary_tier)

In [24]: type(Salary_tier)
Out[24]: int
```

Python for Data Science 14

So, one important point here is that only few types of coercions are accepted. Now, let us say I have a variable call `Salary_tier`, which is a string data type. Now, `Salary_tier` contains an integer which is enclosed between single quotes. So, in this case the `Salary_tier` is basically a number which is 1, but that is enclosed within single quotes so it is still treated as a string data type.


Now, if I want to change it to an integer, I will just say `Salary_tier=int(Salary_tier)`. Now, this will convert the data type for `Salary_tier`. So now, the data type is converted from string to an integer.

(Refer Slide Time: 11:15)

## Coercing object to new data type

- However if the value enclosed within the quotes is a string then conversions will not be possible

```
In [19]: Employee_name="Ram"
In [20]: Employee_name=float(Employee_name)
Traceback (most recent call last):
  File "<ipython-input-20-b0286eb77de0>", line 1, in <module>
    Employee_name=float(Employee_name)
ValueError: could not convert string to float: 'Ram'
```




Python for Data Science

However, not all types of coercion is possible. For instance, if I would like to convert the *Employee\_name*, Ram, to an integer or a float I will be getting an error. So, the value which is enclosed between codes, if it is a float or a integer type then such types of coercion will be possible. However, if it is a string or a set of characters than those types of coercion would not be possible.

(Refer Slide Time: 11:42)

## Summary

- Conventions to name a variable
- Basic data types
  - Get data type of a variable
  - Verify if a variable is of a certain data type
  - Coerce variable to new data type



Python for Data Science

So, to summarize in this lecturer we looked at some of the common naming conventions to name a variable in Python. We saw some of the basic data type related operations, one

of that was to get the data type of a variable. Then, verify if a variable is of a certain data type and how to coerce the data type of an existing variable to a newer one.

Thank you.