Variables and Data Types

Moro Abdul Wahab

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

Naming variables

- 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.

Naming variables

- Variables can be named alphanumerically

Age = 55   age = 5 age2=55 Age2=55

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

  2age=55

  SyntaxError: invalid decimal literal

Naming variables

- Other special character
  - Underscore(_)
  - Use of any other special character will throw an error
  - Variable names should not begin or end with underscore even both are allowed

```
Student_id=501
```

```
>>> Student@id=203
SyntaxError: cannot assign to expression here. Maybe you meant '==' instead of '='?
```

```
_age=55

age_=44
```

Naming conventions

- Common accepted case types
  - Camel (lower and upper)

    ageEmp=45 AgeEmp=45

- Snake

  age_emp=45 Age_emp=45

- Pascal

  AgeEmp=45

Assigning values to multiple variables

Code

```
English,Mathematics,Science=78, 82, 90
```

Data types

Basic data types

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

Identifying object data type

Find data type of object using

Syntax: type(object)

```
Student_name=”Sam”

Age=9

Height=80.9
```

Checking the data type of an object

```
>>> type(Student_name)
<class 'str'>
>>> type(Age)
<class 'int'>
>>> type(Height)
<class 'float'>
```

Verifying object data type

Verify if an object is of a certain data type

Syntax: type(object) is datatype

Student_name = "Moro"

Age = 18

Height = 150.6

type(Height) is int

type(Age) is float

type(Student_name)

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

```
Student_name = “Ram”
                type(score)

class = 4
                new_score = int(score)

score = 78.5
                                type(new_score)
```

Coercing the data type of an object

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

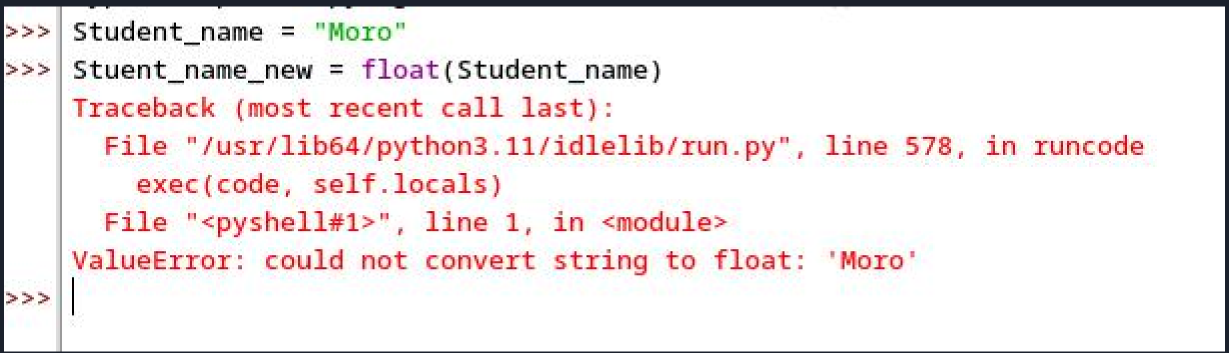
Salary_tier = int(Salary_tier)

type(Salary_tier)

type(Salary_tier)

Coercing object to new data type

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



```
>>> Student_name = "Moro"
>>> Stuent_name_new = float(Student_name)
    Traceback (most recent call last):
      File "/usr/lib64/python3.11/idlelib/run.py", line 578, in runcode
        exec(code, self.locals)
      File "<pyshell#1>", line 1, in <module>
    ValueError: could not convert string to float: 'Moro'
>>>
```

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