

# Foundations of Computing

Tutorial/Workshop ◦ ◦ ◦ ◦

Week 3

# Today's Tutorial

○ ○ ○ ○

1

Data types

2

Operators and overloaded  
operators

# What is a data type? Can the data type of an object change?



A data type is a **classification of data** which tells python how to store and process it and defines what we can do with it.

Data types are permanent when you create the object, but you can **convert a value** into another type with **functions**.


**In groups, fill in the below table.**



Type	Example	What does it store?	What can we do with it (functions, operations...)?	How do we convert to it?
	"Hello"			
	123			
	3.1415			
	True			

What is the difference between the second and third type, both being numerical?

# Completed table

				
Type	Example	What does it store	What can we do with it (functions, operations...)	How do we convert to it?
<code>str</code>	<code>"Hello"</code>	A sequence of characters	<code>len()</code> , <code>input()</code> , <code>print()</code> , (in future: slicing, indexing, <code>.lower()</code> )	<code>str()</code>
<code>int</code>	123	A whole number (integer)	Arithmetic operations, counting & numbering (in future: indexing and slicing)	<code>int()</code>
<code>float</code>	3.1415	A number containing a fractional part	Arithmetic operations, mathematics & real world measurements	<code>float()</code>
<code>bool</code>	True	A truth value (T/F)	(in future: result of truth tests, used in conditional statements)	<code>bool()</code>

Integer is a whole number with no fraction, float has a fractional part (can be .0 though)

# Exercises 1&2



# Exercise 1 answer



1. Look at the following form and decide which data types (`str`, `int`, `float`, or `bool`) should be used to store each field.

Name: `str`

Customer ID: `int` ... or possibly `str` if ID not numeric

Address: `str`

Postcode: `int` or `str`: in some countries a postcode is alphanumeric; with a string we can test the `len()` of the input to ensure the correct amount of characters are inserted.

Do you own or rent? `str`? But you may get "rent", "r", "Ren" and other answer variations. If question were "Do you rent?" a `bool` could be appropriate as there are only two options: yes/no.

Length of bench top: `float`

Width of bench top: `float`

Are you interested in further offers? `bool`

# Exercise 2 answer



2. Evaluate the following by hand:

(a) `str(3 + 4) + "cakes"`

A: `'7cakes'` (Note lack of space before "cakes")

(b) `int(5 / 2)`

A: `2` (2.5 without int conversion; this floors the number)

`int(3.9)`

`>3`

`float("357.23")`

(c) `float("357" + "." + "23")`

A: `357.23`

(d) `bool("anything")`

A: `True` (Note: from Worksheet 3. Any non-empty string will convert to True)

`print(bool(""))`

`>False`

`print(bool(" "))`

`>True`



# What is an “operator”? Which operators have we learned so far and what do they do?



An operator is a **symbol used to calculate some result** based on one or more operands.

We've used `+`, `-`, `*`, `/` as well as `%` (modulo: remainder when first value divided by second), `//` (integer division: converts result into integer – rounds down) and `**` (exponential). Also `=` for assignment.

Note that **order of operations** applies in Python.

# What is operator overloading?



Overloading is where the **same operator works in slightly different ways** for different numbers or types of operands.

For example: +

when used with **numerical** types, + is **arithmetic addition** and when used with **strings/ sequences** it is **concatenation**, which means joining together.

# What is a “variable”? Why/how do we use them?



A variable is a place in the computer's internal memory where a **value can be stored**. A name – or identifier – is used to access a variable.

Created by declaration, their value is controlled with the **assignment** operator (=). Variables are useful because we can **refer back to the same value** more easily.

They are easy to update and use in calculations

# Exercises 3&4



# Exercise 3 answer

3. Evaluate the following by hand, given the assignments  $a = 1$ ,  $b = 2$ ,  $c = 2.0$ :

**A:** *Note: these variable names are bad and you shouldn't use single-letter names in your code.*

(a)  $a / a$

**A:**  $1.0$  (Note type conversion because of division)

(b)  $b + b$

**A:**  $4$

(c)  $b + c$

**A:**  $4.0$  (Note type conversion because of float operand)

(d)  $a / b$

**A:**  $0.5$  (Type conversion: division)

(e)  $a // b$

**A:**  $0$  (Floor division converts to integer; rounds down)

(f)  $a \% b$

**A:**  $1$

(g)  $a + b / c$

**A:**  $2.0$  (Order of operations: division evaluated first)

(h)  $(a + b) / c$

**A:**  $1.5$  (Order of operations: brackets indicate priority)

# Exercise 4 answer

4. What is the output of the following? Why?

**A:** *This question is a demonstration of operator overloading*

(a) `123 + 123`

**A:** `246` (Arithmetic addition)

(b) `"123" + "123"`

**A:** `'123123'` (String concatenation)

(c) `"123" + 123`

**A:** `TypeError: must be str, not int`  
(Can't mix non-numeric types)

(d) `3 * 4`

**A:** `12` (Arithmetic multiplication)

(e) `"3" * 4`

**A:** `'3333'` (String multiplication/repetition)

(f) `"3" * "4"`

**A:** `TypeError: can't multiply sequence by non-int of type 'str'`  
(Can't multiply two strings)

# How does the `input()` function work?



Takes a **string** prompt as **argument**. When run, **displays that prompt** to the user and lets them enter some text.

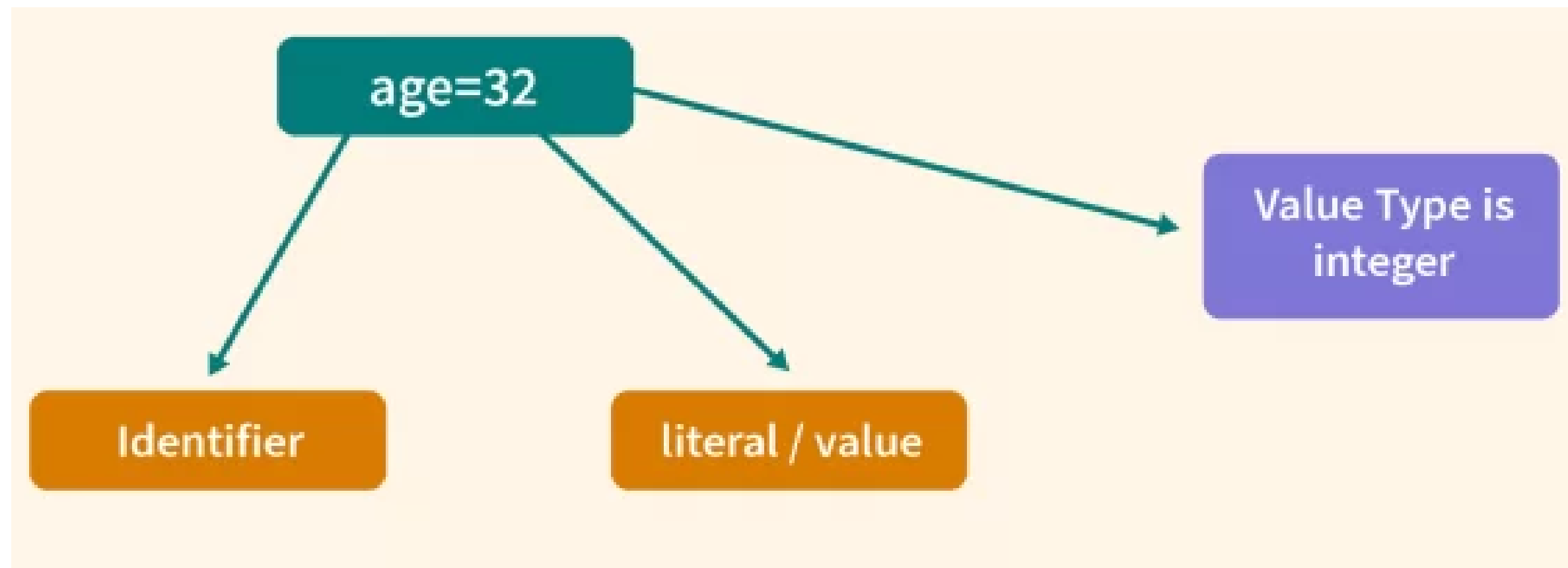
That text becomes the return value of the `input()` function, as a string. This can be saved as a variable.

```
name = input("What is your name? ")  
print(name)
```

```
>>>What is your name? John  
>>>John
```

# What is a literal?

A literal is a value typed directly into a program, rather than one referenced by a variable. Can be int, float, string, etc.





# WORKSHOP

○ ○ ○ ○

Grok, problems from sheet, ask me questions :)