

COMP1021  
Introduction to Computer Science

# Handling of Data Types

David Rossiter and Gibson Lam

# Outcomes

- After completing this presentation, you are expected to be able to:
  1. Explain the various data types in Python
  2. Write code to check the data types of variables
  3. Convert between some of the data types

# Data Types in Python

- Data types mean the ‘type’ of things that you store inside variables
- For example, if you run this line of code:

```
mynumber = 5
```

we say that the variable has an *integer* data type because it stores an integer value (5)

# Data Types You Have Used So Far

- You have used the following data types:
  - Numbers
    - Integers, a number with no decimal place e.g. 1 and 5
    - Floats (=floating point numbers), a number with a decimal place e.g. 1.2 and 3.14
  - Collections
    - Lists, e.g. [1, 0, 2, 1]
    - Tuples, e.g. (200, 100)
    - Strings, e.g. "I am a piece of text!"
  - Booleans, i.e. True or False
    - (Later we will probably see another type, a *dictionary*)

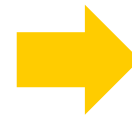
# A Float

- A float (=floating point number) is called that because it contains a decimal place which can ‘float’ (move around)
- For example, you could say these are all the same number, it is only the decimal place which has moved:
  - 10.458      • 1045.8      • 1.0458
- Even if the digit after the decimal place is a zero it is still a float e.g. this is a float: 2.0

# Knowing the Data Type

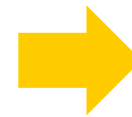
- You can use `type` to tell you the data type of the thing stored in a variable, some examples:

```
number_of_dogs = 1  
print(type(number_of_dogs))
```



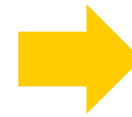
<class 'int'>  
*An integer*

```
age_of_my_dog = 1.5  
print(type(age_of_my_dog))
```



<class 'float'>  
*A float*

```
name_of_my_dog = "Toffee"  
print(type(name_of_my_dog))
```



<class 'str'>  
*A string*

# More Data Types

```
i_am_a_frog = False  
print(type(i_am_a_frog))
```



<class 'bool'>

*A boolean*

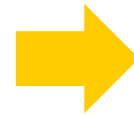
```
my_cats = ("Charlie", "Popcorn", "Kitty")  
print(type(my_cats))
```



<class 'tuple'>

*A tuple*

```
cat_ages = [2.5, 9, 5]  
print(type(cat_ages))
```



<class 'list'>

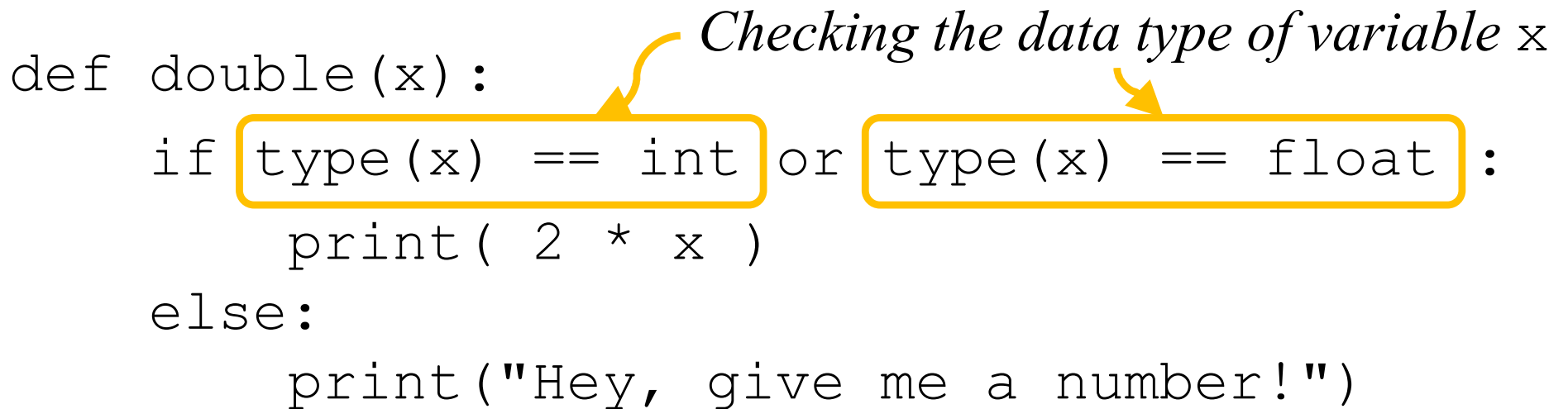
*A list*

# Checking Data Type

- Sometimes it is useful to make sure the data type is correct before you run some code
- Here is an example function `double()`

*Checking the data type of variable x*

```
def double(x):  
    if type(x) == int or type(x) == float:  
        print( 2 * x )  
    else:  
        print("Hey, give me a number!")
```

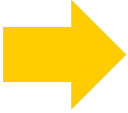






- The function doubles the given number but prints an error if the input `x` is not a number



# Running the Example

- You can test the function in the previous slide by using different input values

<code>double (5)</code>		10
<code>double (7.2)</code>		14.4
<code>double ("flower")</code>		Hey, give me a number!
<code>double ([200])</code>		Hey, give me a number!
		<i>A list</i>


# Data Type Conversion

- Some code may generate errors when the correct data type is not being used
- Some Python code may have a different behaviour when it is used with different data types
- E.g. using ‘+’ with two numbers means addition; using ‘+’ with two strings/lists/tuples means ‘gluing’ the two things together
- So sometimes it’s wise to check that data types are appropriate before the data is used
- You may need *data type conversion*

# Converting Between Numeric Data Types

- You have used two types of numeric data: integers and floats (=floating point numbers)
- To convert from an integer to a floating point number you use `float()`
- To convert from a floating point number to an integer you use `int()`

*Python thinks a number is an integer if it doesn't have a decimal point; otherwise it's a float*



```
print(float(5))
```



5.0

```
print(int(5.0))
```



5

# Storing as an Integer or a Float

- For a numeric value 5, Python displays it as '5' when it is stored as an integer
- For the same value 5, Python displays it as '5.0' when it is stored as a float

*The number is  
stored as an integer*

```
mynumber = int(5)  
print(mynumber)
```

➡ 5

*The number is  
stored as a float*

```
mynumber = float(5)  
print(mynumber)
```

➡ 5.0

# Converting from Numbers to Strings

- When you need to display a number you typically need to convert the number to a string before you can put the number together with other text
- You use the `str()` function to convert a number to a string, for example:

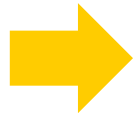
```
age = 21  
print("I am " + str(age) + " years old!")
```



I am 21 years old!

# Converting from Numbers to Strings

```
print("Just like 1+1 is", 1+1, "my heart for you is", True)
```



```
Just like 1+1 is 2 my heart for you is True
```

- `print()` is clever, it can print almost anything
- However, `turtle.write()` is not so clever
- For example, this doesn't work:

```
turtle.write("Just like 1+1 is", 1+1, "my heart for you is", True)
```

- You can fix it like this:

*This means the code continues  
on the next line*

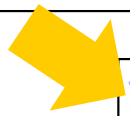


```
turtle.write("Just like 1+1 is "+str(1+1) \
            +" my heart for you is "+str(True))
```

# Converting From Strings to Numbers

- You can use the `int()` function to convert a string to an integer
- You can use the `float()` function to convert a string to a floating point number
- For example, you need to do that after you ask a user for number input using the `input()` function:

```
age = input("How old are you? ")
age = int(age)
print("You look like a "+ str(age*2) + "-year-old to me!")
```




```
How old are you? 21
You look like a 42-year-old to me!
```

# Possible Problem When You Convert a Number to an Integer

- You need to be careful when you convert a string to an integer
- In Python you will get an error if the string contains a decimal point, like this:

```
age = "21.5"  
age = int(age)
```



```
Traceback (most recent call last):  
  File "C:/Users/dross/OneDrive/Documents/00_teaching/1021/  
datatype1.py", line 56, in <module>  
    age = int(age)  
ValueError: invalid literal for int() with base 10: '21.5'
```



# A Safer Approach

- A safer approach to convert a string to an integer is:
  - First, convert the string to a floating number
  - Then, convert the floating number to an integer
- Here is an example:

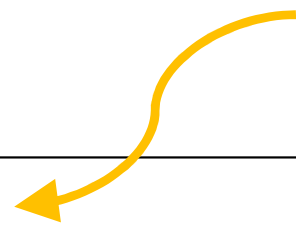
```
age = "21.5"  
age = int(float(age))  
print(age)
```

 21


# When a Float is Converted to a String

- Sometimes when converting a number to a string the result may not be what you expect
- For example, if the number is stored as a floating point number you will have a decimal place in the resulting string:

*Because there is a '.0' at the end it means this is a floating point number*



```
age = 21.0
print("I am " + str(age) + " years old!")
```



I am 21.0 years old!