# COMP 10280
# Programming I (Conversion)

John Dunnion

School of Computer Science
University College Dublin

COMP 10280 Programming I (Conversion)/Lecture 5

# Outline

# Expressions

- The interpreter can act as a simple calculator
- When you type an expression, the interpreter evaluates the expression and prints out the value
- The operators +, -, * and / work just like in most programming languages, eg C and Java
- Parentheses (( and )) can be used to group sub-expressions
- Expressions in Python have a particular type
- Whole numbers (integers) are represented in Python using the type int
- Numbers with a fractional part (real numbers) are represented in Python using the type float

# Arithmetic operators in Python

| Python Operator | Operation |
|---|---|
| + | Addition |
| − | Subtraction |
| * | Multiplication |
| / | [Floating-point] Division |
| // | Integer Division |
| % | Remainder after integer division |
| ** | Power |

# Python Expressions (1)

```
>>> 2 + 2
4
>>> 50 * 4
200
>>> 4 * 3 + 2
14
>>> 4 * (3 + 2)
20
```

# Python Expressions (2)

- The integer numbers (eg 1, 2, 20, 20000000) have type int
- Numbers with a fractional part (eg 1.5, 2.444, 20.0) have type float
- Expressions with mixed type operands convert the integer operand to floating-point

# Python Expressions (3)

```
>>> 1 + 1
2
>>> 4 + 4
8
>>> 20.5 + 42.1234
62.6234
>>> 1234.5 + 765.5
2000.0
>>> 50 * 5
250
>>> 50 * 5.0
250.0
>>> 234.5 * 15
3517.5
```

# Division in Python

- Division (/) in Python 3.x always returns a `float`
- Division (/) in Python 2.x between two `int`s returns an `int`

# Division in Python 3.x

```
[john@localhost ~]$ python3
Python 3.5.0 (default, Sep 15 2015, 06:24:05)
[GCC 4.8.3 20140624 (Red Hat 4.8.3-1)] on linux
Type "help", "copyright", "credits" or "license"
for more information.
>>> 6 / 3
2.0
>>> 7 / 3
2.3333333333333335
>>> 6 / 3.0
2.0
```

# Division in Python 2.x

```
[john@localhost ~]$ python2
Python 2.7.10 (default, Sep 15 2015, 08:05:56)
[GCC 4.8.3 20140624 (Red Hat 4.8.3-1)] on linux2
Type "help", "copyright", "credits" or "license"
for more information.
>>> 6 / 3
2
>>> 7 / 3
2
>>> 6 / 3.0
2.0
```

# Division and “Integer Division” in Python 3.x (1)

- Division (/) in Python 3.x always returns a `float`
- To do integer division (“floor division”) and always get an `int` result, use the // operator
- To get the remainder after integer division, use the % operator

# Division and “Integer Division” in Python 3.x (2)

```
[john@localhost ~]$ python3
Python 3.5.0 (default, Sep 15 2015, 06:24:05)
[GCC 4.8.3 20140624 (Red Hat 4.8.3-1)] on linux
Type "help", "copyright", "credits" or "license"
for more information.
>>> 23 / 3
7.666666666666667
>>> 23 // 3
7
>>> 23 % 3
2
>>> 7 * 3 + 2     # result * divisor + remainder
23
```

# Powers

- The "**" operator can be used to calculate powers

```
>>> 3 ** 2      # 3 squared
9
>>> 2 ** 8      # 2 to the power of 8
256
```

# Variables and assignment

- A value can be assigned to a variable using the = operation
- After an assignment in the interpreter, no result is displayed before the next prompt

```
>>> length = 20
>>> breadth = 12
>>> area = length * breadth
>>> area
240
```

# Variables must be defined

- If a variable is ‘not defined” (not assigned a value), trying to use it will generate an error

```
>>> x
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'x' is not defined
```

# The variable _

- In interactive mode, the last printed expression is assigned to the variable _
- This makes it easier to continue calculations and use the Python interpreter as a “desk calculator”
- The _ variable should be treated as read only: don’t explicitly assign a value to it

```
>>> tax_rate = 13.5 / 100     # Tax rate of 13.5%
>>> nett_price = 2000     # Nett price
>>> nett_price * tax_rate     # Tax due on item
270.0
>>> nett_price + _     # Nett price + tax
2270.0
```

# Using numbers in Python programs (1)

```
# Calculating area of a rectangle
# p10.py

length = 2.7
breadth = 5.5

print('Length is: ', length)
print('Breadth is: ', breadth)

area = length * breadth
print('Area of rectangle is: ', length * breadth)
print('Area of rectangle is: ', area)
```

# Using numbers in Python programs (2)

```
Length is:  2.7
Breadth is:  5.5
Area of rectangle is:  14.850000000000001
Area of rectangle is:  14.850000000000001
```

# Using numbers in Python programs (3)

```
# Calculating area of a rectangle
# Note use of "+" in print statements
# p11.py

length = 2.7
breadth = 5.5

print('Length is: ' + length)
print('Breadth is: ' + breadth)

area = length * breadth
print('Area of rectangle is: ', length * breadth)
print('Area of rectangle is: ', area)
```

# Using numbers in Python programs (4)

```
>>>

Traceback (most recent call last):
  File "/home/john/Documents/dept/comp10280/
        2016/progs/p11.py", line 8, in <module>
    print('Length is: ' + length)
TypeError: Can't convert 'float' object to str
        implicitly
>>>
```

# Using numbers in Python programs (5)

```
# Calculating tax due on item
# p12.py
tax_rate = 13.5     # 13.5% VAT rate
nett_price = 199.99     # Net price in Euro
print('Nett Price is: ', nett_price)
print('Tax rate is: ', tax_rate)

tax_due = nett_price * tax_rate / 100
print('Tax due: ', tax_due)

total_price = nett_price + tax_due
print('Total price: ', total_price)

print('Total price is: ',
          nett_price + nett_price * tax_rate / 100)
```

# Using numbers in Python programs (6)

```
>>>
Nett Price is:  199.99
Tax rate is:  13.5
Tax due:  26.99865
Total price:  226.98865
Total price is:  226.98865
>>>
```

# Using numbers in Python programs (7)

```
# Calculating area of a square and a circle
# Length of side of square = Diameter of circle
# p13.py

length = 2.7      # Length of side of square
radius = length / 2      # Radius of circle

pi = 3.1415927      # Defining pi

print('Length of side is: ', length)
print('Area of square is: ', length ** 2)

print('Radius of circle is: ', radius)
print('Area of circle is: ', pi * radius ** 2)
```

# Using numbers in Python programs (8)

```
>>>
Length of side is:  2.7
Area of square is:  7.2900000000000001
Radius of circle is:  1.35
Area of circle is:  5.72555269575
>>>
```

# Importing the `math` module (1)

```
# Calculating area of a square and a circle
# Length of side of square = Diameter of circle
# Using math.pi
# p14.py

import math

length = 2.7     # Length of side of square
radius = length / 2     # Radius of circle
print('Length of side is: ', length)
print('Area of square is: ', length ** 2)

print('Radius of circle is: ', radius)
print('Area of circle is: ', math.pi * radius ** 2)
print('Value of math.pi: ', math.pi)
```

# Importing the `math` module (2)

```
>>>
Length of side is:  2.7
Area of square is:  7.290000000000001
Radius of circle is:  1.35
Area of circle is:  5.7255526111167399
Value of math.pi:  3.1415926535897931
>>>
```

# Importing the `math` module

- The `math` module provides some constants and a number of maths functions
- Functions include square root, factorial, trigonometric functions, ...
- Constants include $\pi$ (`math.pi`) and $e$ (`math.e`)
- As you write bigger programs, you will find yourself importing several modules

# Importing modules

- Documentation on the modules available is available at:
  - `https://docs.python.org/3/library`
    (currently **Python 3.7.0** documentation [27 June 2018])
  - `https://docs.python.org/2/library`
    (currently **Python 2.7.15** documentation [1 May 2018])
- For example, documentation on the `math` module is available at:
  - Python 3.x:
    `https://docs.python.org/3/library/math.html`
  - Python 2.x:
    `https://docs.python.org/2/library/math.html`