UNIVERSITY OF HULL

Python

Let’s Start Programming

UNIVERSITY OF HULL

Reminder: If you missed any earlier presentations, be sure to catch up by watching recordings!

Why Python

- Syntactically Simple
- Capable of powerful expression
- Fast for computation
- Popular, Mature
- Lots of support, Libraries etc (Ecosystem)

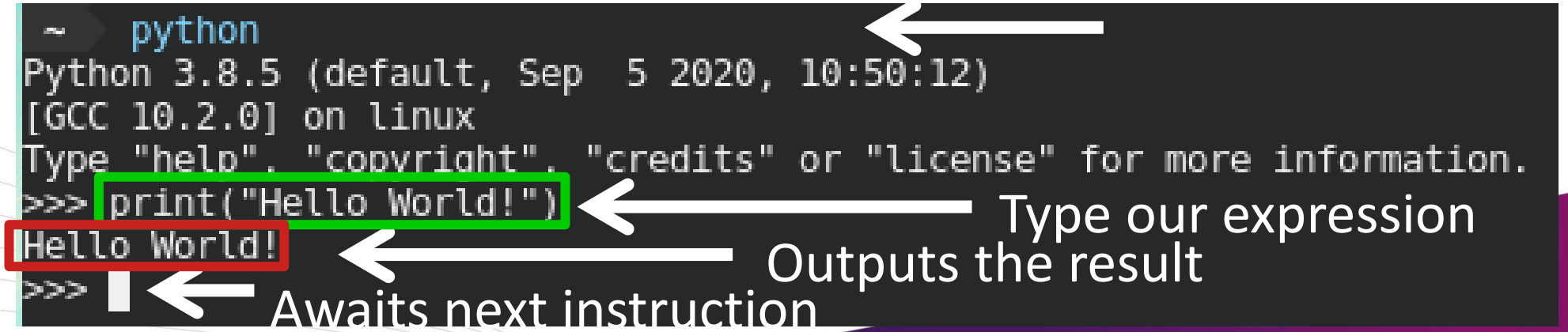
NB:
Programming is introduced in UK primary Schools
From about age 8
Many use Python

UNIVERSITY
OF HULL

Python Interpreter

- A Python Script is a file named *.py
- Executed in sequence by the Python Interpreter
- Allows the creation of more complex algorithms
- Python can also be directly entered into the interpreter:

"Hello World"
The simplest first program



UNIVERSITY OF HULL

What does it look like?

```
print( "------------------------" )
print( "Testing all compilers..." )
print( "------------------------\n" )
for compiler in compilers:

                # Try to extract student name from compiler path if we can
                searchStr = "08348 Compilers Assessment"
                nameIndex = compiler.find(searchStr)
                prelen = len(searchStr)
                studentName = ""
                if nameIndex > 0:
                                # Get the path and print the name
                                studentName = compiler[nameIndex + prelen + 1:compiler.find(S, nameIndex + prelen + 2)]
                                reportEntry(report, studentName)

                reportEntry(report, str_join( "Testing compiler: ", compiler) )
                if not os.access(compiler, os.X_OK):
                                reportEntry(report, str_join( "Cannot find compiler: ", compiler) )
                                continue # Skip it
                # Check for plain localname, make relative path for shell
                if os.path.basename(compiler) == compiler:
                                compiler = str_join(".", S, compiler)

                # Test function
                passed = 0

                # Run all tests
                numExecutedTests = 0
                for f in testSPLfiles:
                                result = runTest(compiler, f, report)
                                if result > 0:
                                                passed += 1
                                                if result == 2:
                                                                numExecutedTests += 1

                # Output summary
                if studentName != "":
                                reportEntry(report, studentName + "'s compiler ")
                percent = passed * 100 / len(testSPLfiles)
                reportEntry(report, str_join("Passed ", passed, " tests out of ", len(testSPLfiles), " (", percent, "%)") )
                #reportEntry(report, "Executed " + numExecutedTests + " test programs successfully")
                compilersTested += 1
```

UNIVERSITY
OF HULL

Python Types

- Python Object have types
  - Called classes
  - Type => the nature of the thing to be stored
  - The type determines what can be done with an object
- There are basic “built-in” classes
- Can build own object classes from these

UNIVERSITY OF HULL

Basic Classes

- int – whole numbers: -2, -1, 0, 1, 2
- float – “real” numbers: 1.412, 0.32 (also complex)
- bool – True or False
- str – “Brian”, “Hello World”
- NoneType – None : a special type for nothing!

UNIVERSITY OF HULL

What class am I?

- This is useful for to determine if a calculation didn't provide output
- To interrogate what type an object has, we can use type() on it.

```
>>> type(3.141)
<class 'float'>
```

```
>>> type(5)
<class 'int'>
```

What type(5) is evaluated as.

```
>>> type(True)
<class 'bool'>
```

```
>>> type(False)
<class 'bool'>
```

```
>>> type("Need input!")
<class 'str'>
```

```
>>> type(None)
<class 'NoneType'>
>>>
```

UNIVERSITY OF HULL

Type Conversion

- We can convert from one type of object to another by casting
  - float(5) converts an integer, 5, to the float 5.0.
  - int(3.141) converts the float, 3.141, to the integer 3.
- We can also do stranger casting to.
  - Caution: Some casts don't make sense! (Semantics)
  - E.g. int("bob") → The interpreter will error and say ValueError!
  - E.g. int("5") → Converts the string "5" to the integer 5.

UNIVERSITY OF HULL

Operators

- With these types we can do things.
  - +, -, *, / operators for plus, minus, multiply, and divide.
  - **, % operators for exponent , and modulo (The remainder of division)
- Examples:
  - A + B → Summation
  - A – B → Minus
  - A * B → Product
    - Integer yields integer, float yields float.

A / B → Division
Outputs float

A % B → Remainder
A ** B → A^B

UNIVERSITY
OF HULL

Operator precedence

- From School: BODMAS or PEMDAS
- Calculate 3 * 4 ** 2 : is it 144 or 48?

- Calculate 6 - 4 / 2 : is it 1 or 4?

- Use parentheses for clarity 6 – (4/ 2)

UNIVERSITY
OF HULL

Variable Assignment

- We can store results in named variables (memory of the computer)

pi = 3.14159265358979

Variable name Value

tau = pi * 2

UNIVERSITY OF HULL

a	b	not b	a and b	a or b
True	True	False	True	True
True	False	True	False	True
False	True	False	False	True
False	False	True	False	False

Some more operations

- Comparison
  - <, >, ==, !=, >=, <= (deliver True or False)
- For completeness
  - Bitwise ~ (negation), & (and), | (or), ^ (xor), <<, >> (bit shift)
- For Boolean logic
  - and, or, not
- Details in the documentation

UNIVERSITY OF HULL

Choosing Variable Names

- Choose a name that describes the result
- Helps the human understand the program
- Python has a list of protected keywords which we can’t name variables

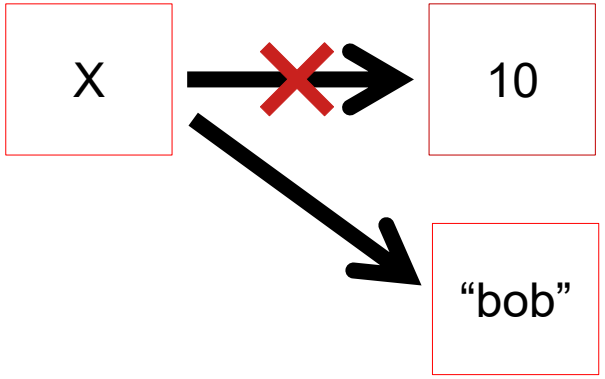
['False', 'None', 'True', 'and', 'as', 'assert', 'async', 'await', 'break', 'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass’, 'raise', 'return', 'try', 'while', 'with', 'yield']

UNIVERSITY OF HULL

Warning to Mathematicians

- The = symbol is not
  - Denoting equivalence
  - Denoting equality
  - Denoting Identity
  - It is not an equation to be solved
- It is just copying a value into memory
- That value can be overwritten later

UNIVERSITY OF HULL



Variable Assignment

- We can re-use the variable name for another assignment operation.
  - Rebind the variable to a new value.
  - Old one still exists in computer memory
  - May get cleaned up later (Garbage Collected)
- Rebinding of variable doesn't have to be of the same type.
  - Usually advised, otherwise you could run into semantics issues ("But I thought x was an int!")

X = 10 → X = "bob"

Known as Dynamic Typing

UNIVERSITY OF HULL

Two commands

- The python function: print
- Use it to output the result

```
>>> print(True and True)
True
>>> print(True or False)
True
>>> print(1|2)
3
```

- Use input to get values from keyboard

In practical applications we rarely use input
- We read from data files
- Used mainly in small examples

UNIVERSITY OF HULL

So now go program

- See the practical worksheet on canvas
- Upload assessment results

UNIVERSITY OF HULL

Thank you

For more information
visit www.hull.ac.uk