LEEDS
BECKETT
UNIVERSITY

Introduction to Programming

Week 7

Tony Jenkins
a.m.jenkins@leedsbeckett.ac.uk

OBJECTIVE

We have now covered all the main ideas in programming.

So we need to practice, and practice some more.

Week 7

Pre-requisites

You should have viewed or read the lecture:

"Scripts and Modules"

on MyBeckett.

We have actually been doing this all along, so there was not much new there.

You should now be happy with writing small programs, running them, and checking whether the results look promising.

Scripts and Modules

Python Programs

A Python program is sometimes called a *script*, and Python itself is sometimes called a *scripting language.*

Compare with C (a *compiled language*), or Java (which is somewhere between the two).

A compiled language is transformed into an executable version before it can be run.

A scripting language is *interpreted* line-by-line, and each line executed as it is reached.

We have been using the Python 3 Interpreter.

Scripts and Modules

Python Programs

A Python program is sometimes called a *script*, and Python itself is sometimes called a *scripting language.*

Compare with C (a *compiled language*), or Java (which is somewhere between the two).

A *module* is a related set of functions, held in a file, and intended to be imported when needed.

Compiled languages *link* all the code together into the single executable.

Interpreted languages insert the code from the module where indicated in the main script.

Scripts and Modules
Definitions

A ***script*** is a single Python program.

A ***module*** is a related set of functions, held in a file, and intended to be imported when needed. A module may also contain constants.

A ***library*** is a collection of modules. You will meet two:

- The standard library.
- PyPi (the Python Package Index).

Using modules from PyPi is beyond the scope of this module as it involves dependency and package management.

Scripts and Modules
Python Scripts

The top line of a Python script can be used on Unix-like operating systems to tell the OS what interpreter to use to run the program.

Opposite, the operating system would use the Python 3 interpreter.

```
#!/usr/bin/env python3

if __name__ == '__main__':
    print('Hello, World')
```

Scripts and Modules

Python Scripts

The top line of a Python script can be used on Unix-like operating systems to tell the OS what interpreter to use to run the program.

Opposite, the operating system would use the Python 3 interpreter.

But in this second program it would use the Ruby interpreter.

This allows scripts to run outside any IDE.

Note: This was illustrated in Lecture #3.

```
#!/usr/bin/env python3

if __name__ == '__main__':
    print('Hello, World')
```

```
#!/usr/bin/env ruby

puts "Hello, world!"
```

Modules
Useful Chunks

If we `import` a module we are writing, the code is not executed. But it would be good to test it.

So it is common (and good practice) to include some simple tests at the bottom of a module file.

These do not get executed when the module is imported, but are handy for testing.

They *are* executed when the module is passed to the interpreter as if it were a script.

```
def circ_area (radius):
    pass

def square_area (side_1, side_2):
    pass

if __name__ == '__main__':

    print(circ_area(2.0))
```

Module
Shapes

Write a ***module*** that contains functions that will find:

- The area of a circle, given the radius.
- The area of a triangle, given the base and height.
- The area of a rectangle, given the two sides.
- The area of a square, given one side.

Include a short `__main__` program in the module that tests these functions (work out some test data by hand).

Use the functions in separate program to check that the `import` works as expected.

Programs
Weather Station

A weather station contains two thermometers. One is set to measure in Celsius and the other, for some reason, uses Fahrenheit.

A user enters the day's readings into a program. After all are entered, the program displays the maximum, minimum, and average temperatures.

Input is the reading, followed by "C" or "F" to indicate the scale. The end of the readings is indicated by blank input.

Programs
High-Low

The program "thinks" of a number between 0 and 100.

The player guesses, and the program answers "higher" or "lower" until the guess is correct.

The player wins if they guess correctly in ten guesses or fewer, otherwise they lose.

Note: The program can increase its chances of winning by considering the player's likely strategy, and by not picking certain numbers!

Programs
RoShamBo

Or "Paper - Scissors - Stone".

Write a program that simulates this game.

A player enters their choice, and the computer randomly picks (it does not cheat!).

The program then displays the result.

A match is best out of 10 games. At the end of the match the program should display the final result.

Programs
High-Low

The program "thinks" of a number between 0 and 100.

The player guesses, and the program answers "higher" or "lower" until the guess is correct.

The player wins if they guess correctly in ten guesses or fewer, otherwise they lose.

Note: The program can increase its chances of winning by considering the player's likely strategy, and by not picking certain numbers!

NEXT

It is now time to practice, and practice.

And then practice some more.

And next week we will have the Practice Test.

Thank you

LEEDS
BECKETT
UNIVERSITY