A children's guide to *Python* programming

By Simon Haughton

(Tested on *Pythonista 1.4* for *iOS*, using a *Python* 2.7.5 interpreter.)

1. Printing text and creating variables

a) Open the *Pythonista* app >>> .

Press ✎ and start an Empty Script.

Type a name for your program at the top. ✎

Type these commands into the 'script' window:

```
print 'Hello world.'
print '\n'
print 'I am learning Python.'
```

Press ▷ and watch the 'interpreter' window. (Swipe right to go back.)

Program - A sequence of commands that are followed in order to carry out a task.
Run - Carrying out the commands in a program. Also known as execute.

What does the `print` *command do?*

What does printing **\n** *do?*

What happens if you make a mistake in your commands?

b) Press ☰ and create an Empty Script .

Type a name for your program at the top. ✎

Type these commands in and then run them ▷ :

```
forename = raw_input('What is your forename? ')
print 'Hello', forename
```

What does the `raw_input` *command do?*

Does it matter if you type in text other than your

Variable – A value that can be stored and used in a program.

Edit and improve:

- Add a variable to store a **surname** when the user is asked. Then add a print command so it prints their full name.

```
surname = raw_input('What is your surname? ')
print 'Hello', forename, surname, '!'
```

2. Calculations and random numbers

a) Open the *Pythonista* app .

Press ☰ and create an Empty Script .

Type a name for your program at the top. ✎

Type these commands in and then run them ▷ :

```
print 100 + 10
```

Is the calculation still solved if you use a negative number or a decimal number?

Edit and improve:

- Change the commands to do a different calculation, such as a: take away **-**, multiplication ***** or division **/**.

Testing - Trying out a program to check if it works as expected.
Debugging - Finding and correcting mistakes in a program's source code.

b) Press ☰ and create an Empty Script .

Type a name for your program at the top. ✎

Type these commands in and then run them a few times ▷ :

```
import random
number = random.randrange(10,20,1)
print number
```

What does the `.randrange` *command do?*

Edit and improve:

- Change the number **10** to a smaller number and the number **20** to a bigger number to see what effect this has on the program.
- Add some commands to do calculations with the random number. e.g.

```
print number + 10
```

3. Number variables and adding comments

a) Open the *Pythonista* app >>> and create an Empty Script .

Type these commands in and then run them ▷ :

```
number = int(raw_input(‘Type a whole number: ’))
answer = number * 8
print number, ‘multiplied by 8 is’ , answer
```

What happens if you type in a decimal number instead of an **integer** *(whole number)?*

Edit and improve:

- Find out what changing `int` to `float` lets you do. (Remember to change it back to `int` afterwards!)
- Add commands so the answer to an addition is printed as well. You will need to use another variable called **answer2**:

```
answer2 = number + 6
print number, ‘add 6 is’ , answer2
```

- Change the program so you have to type in a second number at the start to use in each calculation. You will need to use another variable called **number2**. Here is a start:

```
number2 = int(raw_input(‘Type a whole number: ’))
```

b) Add these commands to your program:

```
# This is a comment.
```

Does text on a line starting with a hash then a space (#) do anything when the program is run?

Comments - Notes in a program's code which explain what commands do to remind you. They are not run.

Edit and improve:

- Type some comments at the end of some lines in your program to explain what they do.

4. Conditional (if) statements

a) Open the *Pythonista* app and create an Empty Script .

Type these commands in and then run them ▷ :

```
answer = raw_input(‘Do cats bark? ’)
if answer == ‘no’:
  print ‘Correct’
else:
  print ‘Wrong’
```

What does this program do?

Why do you think two equals signs are used and not just one?

IF statement - Decides which commands to run depending on whether certain things (conditions) are true or false.

Edit and improve:

- Change the question being asked (and the answer too, if needed).

b) Create an Empty Script .

Type these commands in and then run them ▷ :

```
mark = int(raw_input(‘Score: ’))
if mark > 80:
  print ‘Outstanding’
elif mark > 40:
  print ‘Great’
else:
  print ‘Good’
```

What does this program do?

What does the `elif` command let you do?

Edit and improve:

- Add another `elif` command between 80 and 40 so that a score of more than 60 is rated as `‘Super’`.

Programming challenge:

Create a program that asks a maths calculation and prints if the user answers it right or wrong. *Can you change one of the numbers in it to a random number?*

5. Lists

a) Open the *Pythonista* app and create an Empty Script .

Type these commands in and then run them ▷ :

```
import random

colours = [‘red’, ’green’]
animals = [‘lions’, ‘bears’]

print ‘My rainbow zoo has:’

colour = random.choice(colours)
animal = random.choice(animals)
print colour, animal

colour = random.choice(colours)
animal = random.choice(animals)
print colour, animal
```

Use copy and paste to help you quickly copy this!

List - A set of values.

What does this program do?

What is the purpose of a list?

Edit and improve:

- Put more items in the list to make the rainbow zoo more fun!

b) Create an Empty Script , type these commands in and then run them ▷ :

```
vehicles = [‘bus’, ‘car’, ‘train’]

print vehicles[0]
print vehicles[1]
print vehicles[2])

vehicles.append(‘plane’)
print vehicles

vehicles.pop(2)
vehicles.insert(2, ‘boat’)
print vehicles

vehicles.remove(‘car’)
print(vehicles)
```

Can you see what the: `.append`, `.pop`, `.insert` and `.remove` commands do?

Programming challenge:

Create a list to store some names. Add commands to: `.append`, `.pop`, `.insert` and `.remove` names. Find out what the `.sort()` command does.

6. Functions

a) Open the *Pythonista* app and create an Empty Script .

Type these commands in and then run them ▷ :

```
import random

def cointoss():
  options = [‘heads’, ‘tails’]
  result = random.choice(options)
  print result

cointoss()
cointoss()
cointoss()
cointoss()
cointoss()
```

Function - A sub-program which is placed at the start of a bigger program and can be called (run) later using its name.

What does this program do?

Why is better to call the function five times than to copy all of its commands five times?

Edit and improve:

- Change the program so it shows the results of rolling a six-sided dice instead. You don't need to put ‘’ around the options because they are numbers.

Programming challenge:

Create a program that tells a user's fortune by calling (running) a function two times which randomly picks a prediction from a list:

e.g. You will be given money.
You will become famous.
You will see an alien.
You will find a lost item.
You will score well in a test.

Can you ask the user to `raw_input` *their name so that it is included in the predictions (e.g. Tom will be given money)?*

7. Iteration (looping)

a) Open the *Pythonista* app >>> and create an Empty Script .

Type these commands in and then run them ▷ :

```
for i in range(4):
  print 'Hello world.’
```

What happens if you change **4** *to a different number?*

Iteration - A way of repeating or looping commands multiple times.

b) Create an Empty Script, type these commands in and then run them ▷ :

```
for i in range(1,11):
  print (i * 10)
```

What happens if you change **1** *and* **11** *to different numbers?*

c) Create an Empty Script, type these commands in and then run them ▷ :

```
password = ‘fish’
guess = ‘’

while (password != guess):
  guess = raw_input(‘Enter password: ’)
  if password == guess:
    print ‘Correct’
  else:
    print ‘Try again’
```

If **==** *means 'equal to',*
what does **!=** *mean?*

What does a **`while`** *loop do?*

Programming challenge:

Create a program in which the computer sets the password as a random `int`eger from 1 to 100 and user has to correctly guess it.

Can you use: `if`, `elif` *and* `else` *commands to give the user clues (e.g.* `‘Too high’` *or* `‘Too low’`*)? Can you add a variable which counts the number of guesses (*`count = count + 1`*)?*

8. Parameters and validation

a) Open the *Pythonista* app and create an Empty Script .

Type these commands in and then run them ▷ :

```
def spell(word):
  for i in range(0, len(word)):
    print(word[i])

spell(‘said’)
spell(‘because’)
```

Parameter - A way of passing a value from the main program to a function when it is called (run).

Edit and improve:

- Add `print len(word)` above the `for` command. *What does it do?*
- Change the program so you can type any word in to pass to the function.

```
word = raw_input(‘Type a word: ’)
spell(word)
```

- Insert the `ord(word[i])` command to the iteration so the special Unicode number of each letter is printed as the word is spelled out.

Programming challenge:

Create a function that uses the `chr(integer)` command to convert a Unicode `int`eger into a letter. Call it several times to decipher a secret word made from Unicode numbers 97-122. For example: 115, 99, 104, 111, 111, 108

b) Create an Empty Script, type these commands in and then run them ▷ :

```
def validation():
  number = 0
  while True:
    try:
      number = int(raw_input(‘Type a whole number: ’))
    except ValueError:
      print ‘Not a whole number!’
    else:
      return(number)

x = validation()
```

What is the purpose of this function?

How could it be useful?

Validation - Automatic checking by a computer to ensure that an entered value is sensible.

Programming challenge:

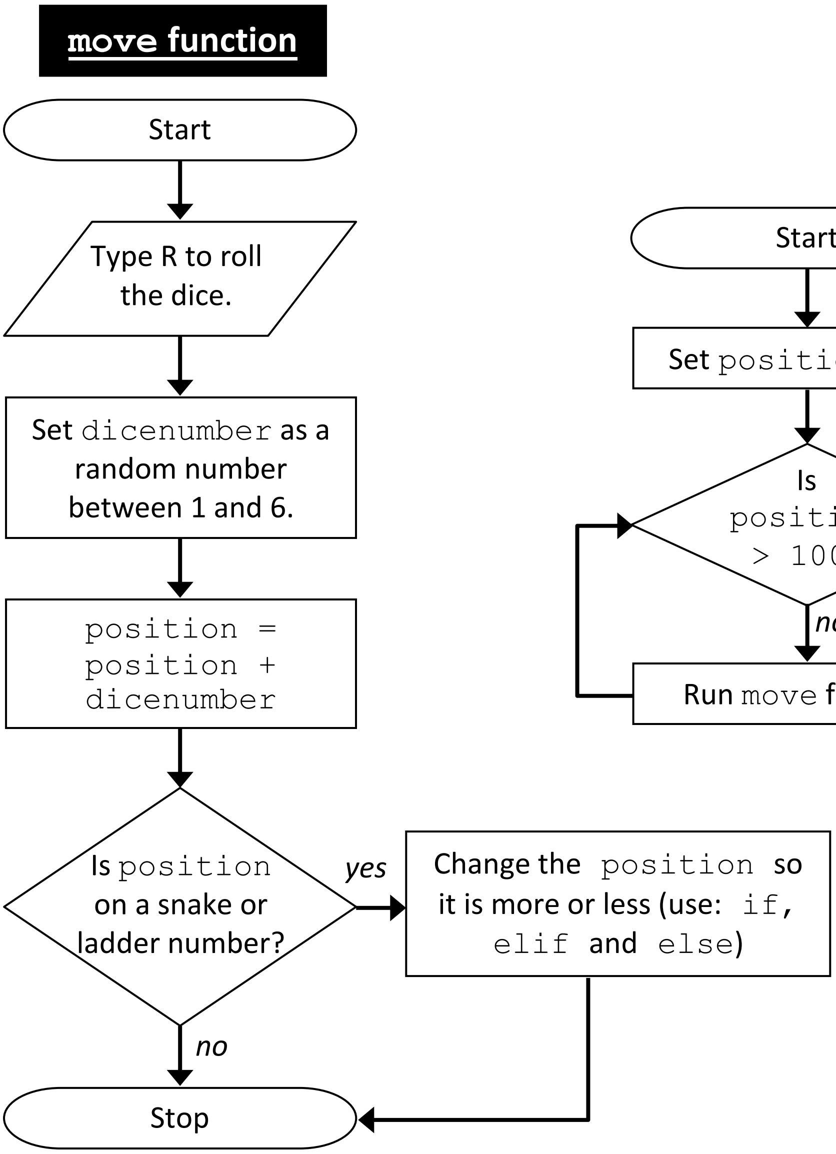
Create a function that prints the biggest of two values (using `if`/`else`), passed to it in parameters. The user will input the two `int`egers they want to compare using the validation function.

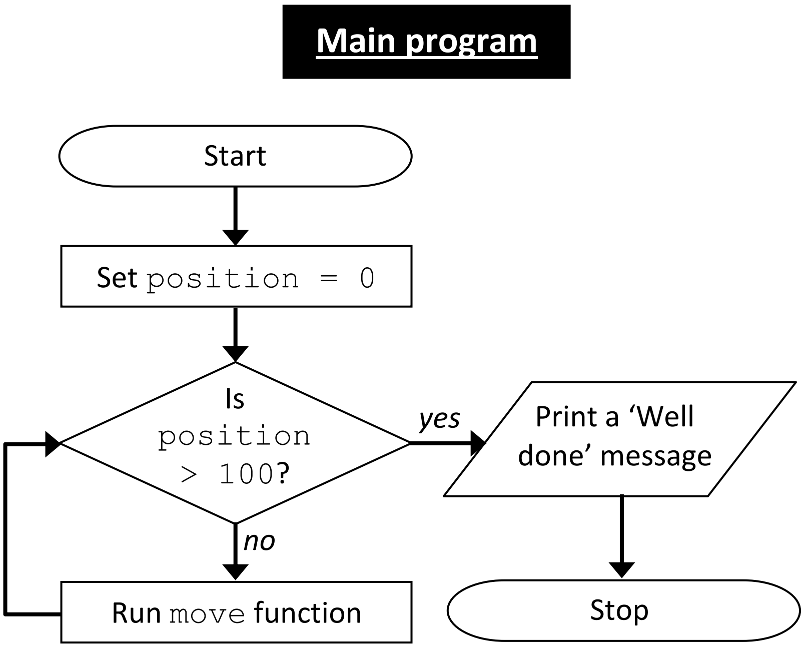
9. Algorithms

Algorithm - An explanation of the processes or instructions a program carries out, usually described in a flowchart.

Programming challenge:

Create a **simple** version of a Snakes and Ladders game:


move function
Start
Type R to roll the dice.
Set dicenumber as a random number between 1 and 6.
position = position + dicenumber
Is position on a snake or ladder number?
yes
Change the position so it is more or less (use: if, elif and else)
no
Stop



- *Can you add more `print` commands to display what is happening on screen?*
- *Can you make the game print the player's name at the end?*
- *Can you add another player to the game whose position is stored in a variable called `position2`?* You will need to make the game let each player move in turns. You could create a variable called `finished` which is set to 0 at the start and changes to 1 when a player wins, forcing the game to stop.

Many thanks to Paul Meakin and Phil Bagge for the inspiration to learn *Python* and write this guide!
www.code-it.co.uk/philbagge.html