Science & Technology
Facilities Council

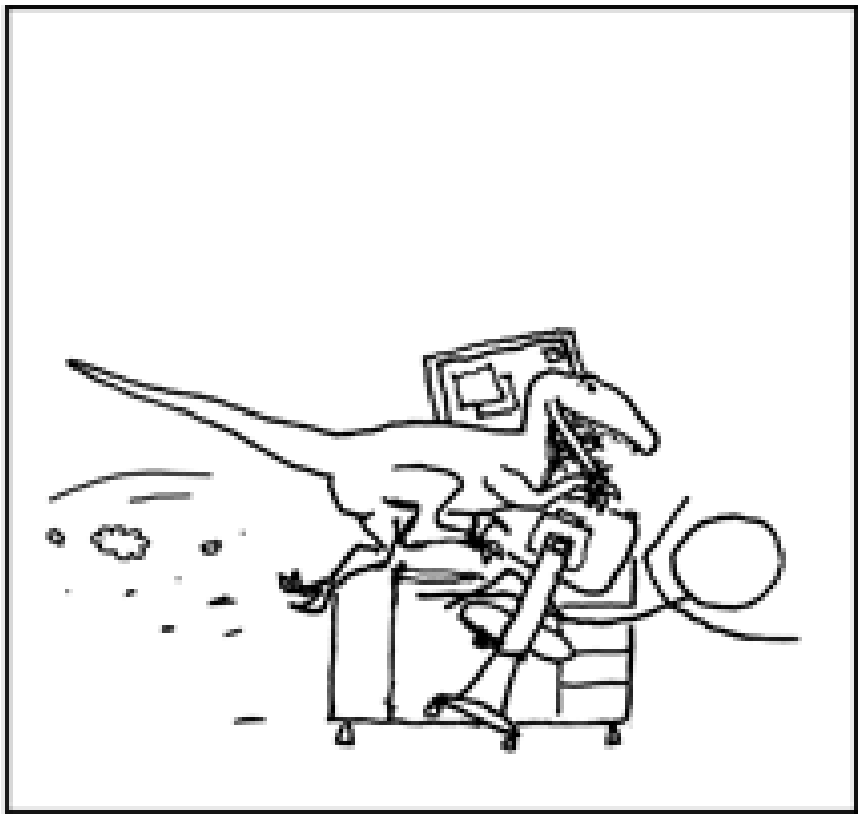
Introduction to Python: Day 2

Revised 2016 – George Ryall

Revised 2017 – Adrian Coveney

Refresher

- Variables
- Strings and integers
- Input
- Functions
- Comparisons
- Conditionals
- While loop
- Boolean operators



Refresher

- **Variables**
- Strings and integers
- Input
- Functions
- Comparisons
- Conditionals
- While loop
- Boolean operators

```
x = 5
y = 7
z = x + y
print(x)
```

Refresher

- Variables
- **Strings and integers**
- Input
- Functions
- Comparisons
- Conditionals
- While loop
- Boolean operators

```
x = int(input())
y = str(x)
```

Refresher

- Variables
- Strings and integers
- **Input**
- Functions
- Comparisons
- Conditionals
- While loop
- Boolean operators

```
print(“What  is your name?”)
name = input()
print(“Hello ” + name)
```

Refresher

- Variables
- Strings and integers
- Input
- **Functions**
- Comparisons
- Conditionals
- While loop
- Boolean operators

```
def greeting(name):
    print("Welcome " + name)

greeting("Dom")
greeting("Steven")
greeting("Lauren")
```

Science & Technology Facilities Council

Refresher

- Variables
- Strings and integers
- Input
- Functions
- **Comparisons**
- Conditionals
- While loop
- Boolean operators

==	equals
>	greater than
<	less than
!=	not equal

Refresher

- Variables
- Strings and int.
- Input
- Functions
- Comparisons
- **Conditionals**
- While loop
- Boolean operators

```
print("Please input x")
x = int(input())
if (x > 5):
  print("x is more than five")
elif (x < 5):
  print("x is less than five")
else:
  print("x equals five")
```

Refresher

- Variables
- Strings and int.
- Input
- Functions
- Comparisons
- Conditionals
- **while loop**
- Boolean operators

```
import random
numrolls = 0
while (numrolls < 4):
  roll = random.randint(1,6)
  print(roll)
  numrolls = numrolls + 1
```

Refresher

- Variables
- Strings and int.
- Input
- Functions
- Comparisons
- Conditionals
- while loop
- **Boolean operators**

```
and
or
not
```

More about functions

We can use functions to do calculations:

```
def atm(initialBalance):
  print("How much do you want to withdraw?")
  withdrawl = input()
  finalBalance = initialBalance - int(withdrawl)
  print("inside fn " + str(finalBalance))
  return finalBalance

balance = 100
balance = atm(balance)
print("outside fn " + str(balance))
```

Exercise 1

- Add to your guess.py game by writing a function which performs a mathematical operation on your guess.

Lists

- **Lists** are collections of values, so we can use **lists** to store many variables
- **Lists** are indexed from zero

```
colours = [“red”, “green”, “blue”]
print(colours[0])
print(colours[1])
print(colours[2])
```

Lists

- We can use **lists** to store many variables
- **Lists** are indexed from zero

```
colours[0] = “purple”
colours[1] = “orange”
colours[2] = “yellow”
print(colours[1])
```

Lists

- You can add lists together: concatenation

```
coloursA = [“purple”, “red”]
coloursB = [“orange”, “blue”]
colours = coloursA + coloursB
print(colours)
```

- You can add strings to list using *append*

```
coloursB.append(“green”)
print(coloursB)
```

Exercise 2 - other tools

Experiment with these methods to see what they do:

- **`list.reverse()`** – for lists

- **`str.upper()`** – for strings
- **`str.lower()`** – for strings
- **`str.split()`** – for strings

for loops

- *for loops* are used when you want to repeat a block of code a certain number of times
- We can loop over integers

```
for i in range(0,5):
  print(i)
```

for loops

- *for loops* are used when you want to repeat a block of code a certain number of times
- We can loop over lists

```
for i in [“red”, “green”, “blue”]:
  print(i)
```

for loops

- *for loops* are used when you want to repeat a block of code a certain number of times
- We can loop over the characters in a string

```
for i in “Hello”:
    print(i)
```

Exercise 3

- Write a **while** loop that fulfils the same function as a **for** loop

Tuples

- **Tuples** are like lists, but we use () not []
- **Tuples** cannot be modified

```
Exampletuple = (42, ‘star wars’, ‘hitchhiker’)
Print(Exampletuple[0])
```

Dictionaries

- **Dictionaries** are collections of **values**
- **Dictionaries** are indexed with **keys**
- **Keys** must be an immutable data type (e.g. a string, number, or tuple) and are unique within the dictionary
- **Values** can be any data type

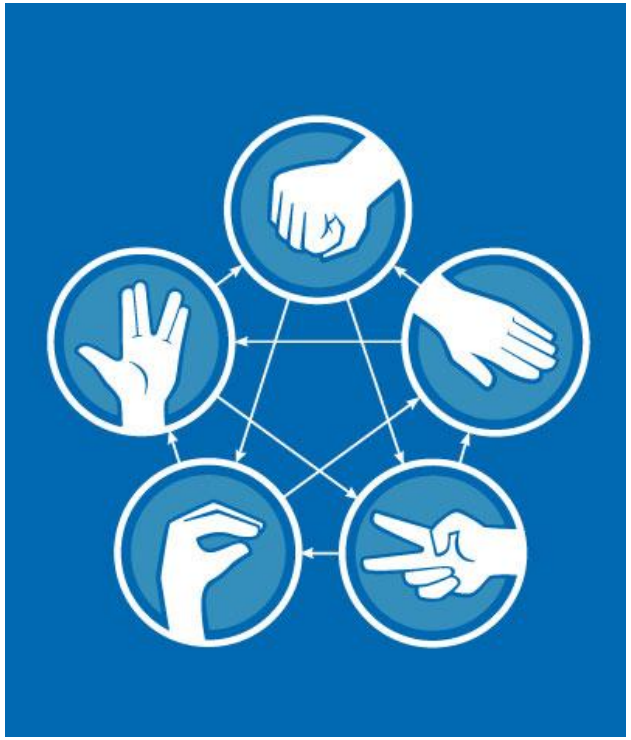
```
newdict={“key0”:“smile”, 1: “Dr Who”, “2”:43}
print(newdict[“key0”])
print(newdict[1])
print(newdict[“2”])
```

Science & Technology Facilities Council

Dragon Realm

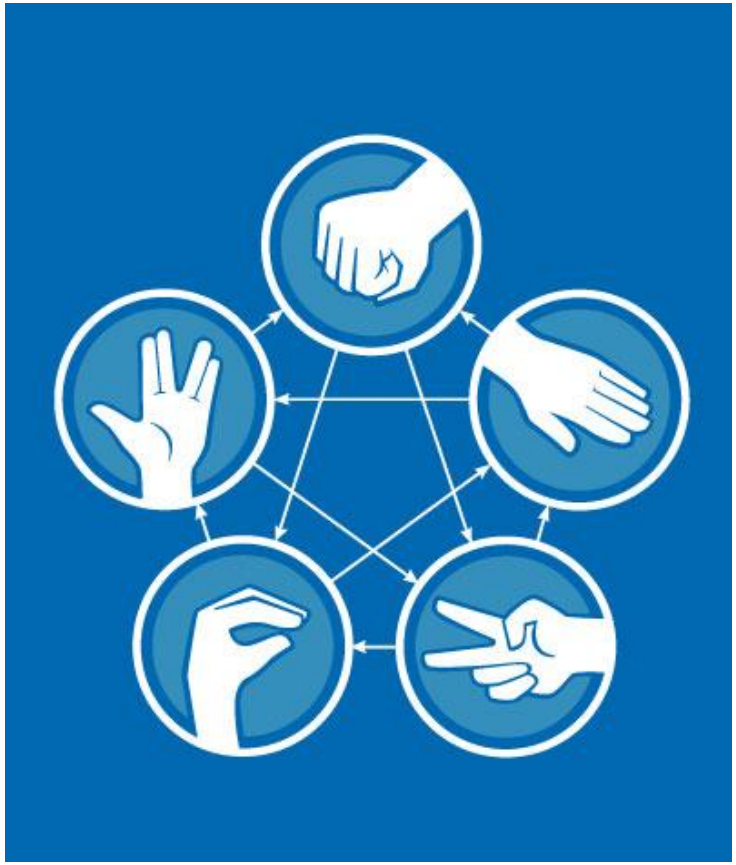
Flowcharts

- Flowcharts are a useful way of mapping out your code



Rock Paper Scissors Lizard Spock

- Scissors cuts Paper
- Paper covers Rock
- Rock crushes Lizard
- Lizard poisons Spock
- Spock smashes Scissors
- Scissors decapitates Lizard
- Lizard eats Paper
- Paper disproves Spock
- Spock vaporizes Rock
- Rock crushes Scissors



Science & Technology Facilities Council

Exercise 4

Write a text adventure game. Your player can choose which cave to go into. In each cave there's a dragon – the player must beat the dragon at a game or be eaten!

- Guess the number
- Anagrams
- Write your name backwards
- Rock paper scissors lizard Spock
- Hangman
- Noughts and crosses