

## John Romero Programming Proverbs

- 1. “No prototypes. Just make the game. Polish as you go. Don’t depend on polish happening later. Always maintain constantly shippable code. (Large teams require more planning though.)”
- John Romero, “The Early Days of Id Software - John Romero @ WeAreDevelopers Conference 2017”

## The first lecture

- this module consists of two pieces of coursework
  - in the first term, [Missile Command](#)  
〈CS2S566\_CW1P1M\_Cover\_PRCW\_PRACTCW1.pdf〉  
implemented in Python3 and Pygame
  - in the second term, a map editing tool for a tablet implemented in Python3
- both pieces of coursework are worth 50%

## Access to the software in this module

- in this module Python3 will be taught on the GNU/Linux operating system
- there are two supported approaches to run Python3
  - firstly using vmware
  - secondly using the Raspberry Pi-4
- both give the same user level experience
- please see the other two components of the lecture this week for more details on either approach

# Python

- Python is a scripting language

# Python Gotha's

- blocks are defined by indentation!
- turn off tabs in your favourite editor
- in your own programs examples never create a name clash with a Python library module
- Python2 vs Python3
  - we will be using Python3


## Python verses similar tools

- Python is a scripting language
  - it can be compiled if necessary to increase speed
- is more powerful than many other scripting languages, Tcl
  - applicable to larger systems development (games, net admin)
- has a much cleaner syntax than Perl
  - easier to maintain
- does not compete head on with Java
  - Java is a systems language like C++

# Python and games

- examples of games which use Python `<http://wiki.python.org/moin/PythonGames>`

# Python can be simple



```
#!/usr/bin/python3  
print("hello world")
```



# Python Modules allow for problem decomposition

- similar to Modula-2

- myfile.py

```
#!/usr/bin/python3  
  
title = "hello world"
```

- foo.py

```
#!/usr/bin/python3  
  
import myfile  
print(myfile.title)
```

- when run prints hello world

## Alternative import



**bar.py**

```
#!/usr/bin/python3  
  
from myfile import title  
print(title)
```

- note that all python modules need to be saved as *name.py*
  - so in our example the module `myfile` was saved into a file called `myfile.py`

# Python builtin types

- python contains many builtin types
  - use them..
- builtin objects make simple programs easy to understand
  - lists, dictionaries, exist, don't reinvent the wheel
- built in objects are more efficient than custom data types

## Builtin objects



numbers	3.14159, 1234
strings	'spam', "fred's"
lists	[1, [2, 'three'], 4]
dictionaries	{'food':'spam', 'taste':'yum'}
tuples	(1, 'spam', 4, 'U')
files	text=open('/etc/passwd', 'r').read()

# Expression operators

■ or, and, not

logical operators (short circuit)

<, <=, >, >=, ==, <>, !=

comparison operators

x | y

bitwise or

z & y

bitwise and

x << y

shift left by y bits

x >> y

shift right by y bits

x[i]

indexing

x[i:y]

slicing

x.y

qualifying (imports)

x(y)

function calls

# Strings

- concatenation via +
  - repeated via \*
- - to write out the 3 times table