PAL Consolidation 1 *University of Hertfordshire – Week 3 – October 2015 –*

Aim & Objectives

- Re-cap on previous session
- Explanation of future sessions
- Short sessions with complimentary exercises

Introduction

In the "*foundation*" sessions we will be working on making sure that students (each of you) have a solid base in programming. To do this, we will be re-iterating key concepts and expanding on them.

Exercise - Start Idle

Please start the idle software as you have done before, making sure you start in the right version (3+). It's good practice to manually make sure each time!

Exercise - Representing Data

Please answer these questions on the back of your paper:

- What is RAM?

  RAM is Random Access Memory, a place where programs can store data temporarily until either the memory is re-allocated or the machine is powered down. This is because the memory is volatile.
- How do computers use RAM?

  Computer Operating Systems (OSes) typically handle how the RAM is used, where programs are allocated memory when/before they request it. Variables and code are loaded into this space with checking in place to make sure your program doesn't accidentally (or on purpose if you're a hacker!) access memory of another program.
- What is a CPU?

  A CPU is a Central Processing Unit and will do most, if not all the calculations involved in your program. Before general purpose programming chips we only had chips made for specific tasks, limiting the range of tasks a chip was capable of achieving.
- How does a CPU work?

  A CPU is a clever and delicate piece of engineering that is very dumb in what it understands. This is on purpose, as simple things done well are often the most effective. An extreme case would be the ARM processor in most of your mobiles, which was made popular for it's reasonable processing power but specifically it's low power consumption. Less instructions also means less power consumed...

In this exercise, please show how you represent the following data in computer memory:

- Bit

  [0/1]
- Byte

  [[01],[0/1],[0/1],[0/1],[0/1],[0/1],[0/1],[0/1]] (8 bits)
- Integer (int)

  [Byte,Byte,Byte,Bytes] (For a 32 bit machine)
- String (str)

  [Char/Byte,Char/Byte,...,'\0']
- Array (list)

  [],[],[],..]

Exercise - Strings

Strings are typically an array of Bytes or Chars, where a Byte is 8 bits and a Char is 16 bits (although variations exist). If we for the moment assume we are using a Byte to represent our files, messages and strings - how do we represent things like newlines, tabs, non-printable characters?

The problem is, we don't have a non-printable key to represents what a newline typically does. Sure we have an enter key but the interesting part is what it does behind the key. The enter key is actually two Bytes! Something call a "*line feed*" and "*carriage return*". This is back from the days of type writers where putting the "*cursor*" back to the beginning of the line was a separate task to moving it down by a line. Typically this stuck around in computing, along with the "*Qwertyuiop*" keyboards (ever wounder where that came from?).

In decimal, the line feed is a '`10`' and the carriage return is a '`13`'. Line feed is typically '`\n`' and carriage return is typically '`\r`'. Because everybody was always having to do '`\r\n`' all the time and rarely the other type, they changed it to just '`\`' to save space and time. These are referred to as escape characters.

What do the escape characters do in this list? Write it down on the back and add any others you know.

- \s

  Inserts a space at the current position.
- \n

  Inserts a newline.
- \r

  Redundant newline sometimes required in networking or file based programming.
- \t

  Inserts a tab at the current position.
- \\

  Inserts a backslash in the current position, as this is a reserved character to tell us about the other escape characters.

  Others:

  * \b - Backspace character for removing the previous character

  * \" - Speech mark character

  * \' - Quote character

In a String, sometimes we want to use the character we are using to describe the String inside the String itself. For example:

```
print("Jack said \"Hello\", \"Who are
you?\"", "\"I am me!\" the voice replied.")
```

Okay, it's not very readable - but we can sort that. The best thing to do when you don't understand is to break it down! Try underlining all of the pairs escape characters and their pairs.

```
print("Jack said \"Hello\", \"Who are
you?\"", "\"I am me!\" the voice replied.")
```

Exercise - Assignment

Don't worry! It's not coursework. Assignment is when you "*assign*" a value to a unique name. Value is meant vaguely here, as it could be a function, instance, class - anything.

Assign the below values:

- '`"Programming wizardry"`' to the unique name '`cool_var`'

  '`cool_var = "Programming wizardry"`'
- '`-34985134.3432`' to a unique name '`bigNum`'

  '`bigNum = -34985134.3432`'
- '`342398534 * 34`' to a unique name '`mathIsCool`'

  '`mathIsCool = 342398534 * 34`'

Resources & Further Reading

'*http://homepages.herts.ac.uk/~db12aba/*' – All content from these sessions updated weekly.

'*http://code.org/*' – A good resource testing your programming skills.

'*http://stackoverflow.com/*' – Highly recommended online help for programmers (NOTE: Employers are interested to know whether you're an active member of this site!).