Computer Language Class

What is a computer language ?

- A Computer Language is a set of rules and instructions that can be understood by a computer.
- The art of using those rules and instructions to make a computer perform some task is called Programming or Coding.
- A computer has a Language it understands just like human beings do.
- Just like people have different languages depending on the country they come from, they are different types of computer languages (programming languages).
- A set of rules and instructions written in a well defined manner to instruct a computer to perform tasks is called a computer program.

Why Learning a programming Language ?

- Meet Our friend Mike. He opened a company in china and he has offered lots of jobs to Chinese people. The problem is, he does not speak Chinese. He is considering taking some Chinese classes to learn to communicate better with his employees and give them instructions.
- Just like our friend Mike will have to learn Chinese to communicate with his staff, we must learn a programming language to communicate with a computer and give it instructions.
- If a computer can understand your instructions then it can do whatever you want it to do.

Programming Language examples

- They are many different types of programming languages.

- All programming languages have the same purpose which is to instruct a computer to perform a task.
- Our main focus in this class is on the Python programming language.

What is a Programmer ?

- A programmer is a person who uses different types of programming languages to instruct a computer to perform a task.
- A programmer understands a certain programming language well and knows how to implement the language to make a program.
- A programmer can also be called a “software developer” or just a “developer”.
- There are different types of developers like game developers, website developers, mobile application developers, etc...

A “Program” vs “Software”

- A program is a set of rules and instructions packed together to instruct a computer to perform a task.
- Software is a set of rules, instructions and documentation on how to use the program and how the program was created.

program + documentation = software

The best way to learn a programming language

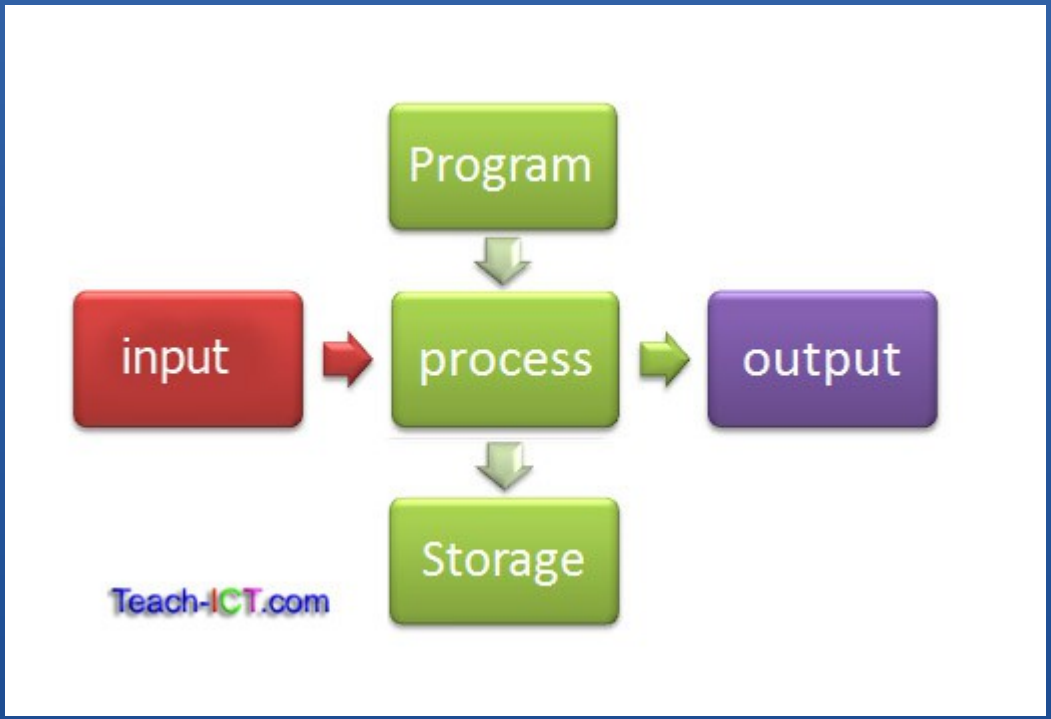
- Learn the way the language is written and the basics of the language.
- Practice (coding) the language over and over….and over!
- Ask questions if you don't understand.
- Solve many problems using the skills you have learned.
- Don't cram...but understand.
- Try to teach yourself a new concept every time you use the language.

How can we use a programming language?

- Making games
- Making websites
- Making mobile applications (APP)
- Making robots
- Etc...

What is a computer system ?

- A computer system is one that is able to take a set of inputs, process them and create a set of outputs. This is done by a combination of hardware and software.



The main parts of a computer system

- Hardware
- Software
- Control Unit
- Input
- Output
- Storage

The computer system

- An input is any information or data that is sent to a computer system for processing.
- An output is any information that has been processed by and sent out from a computer system.

Input devices

output devices

The computer system

- The computer's hardware is a collection of physical elements that constitutes a computer system.
- System software is a type of a computer program that is designed to run a computer's hardware and application programs.
- The control unit (CU) is a component of the computer system that tells the input and output devices how to respond to a program's instructions.
- The set of all these components makes up a computer system.

System Software

- There are two types of System software :
  - Application software
  - Utility software
- Application software allows the parts of a computer system to work together by performing tasks like rendering output onto a display device. e.g: Windows, Mac OS, etc...
- Utility Software is any other software that performs a specific task on top of application software.
- The Python programming language lies under the utility software category

5 Basics of any Programming Language

- Variables
- Control structures
- Data types
- Syntax
- Tools

Variables

- A variable is a storage location and associated symbolic name which contains some known or unknown quantity or information.
- Every variable has a 'word' (symbolic name) that represents it. This name is used to track the information stored in the variable.
- Every variable has a type of information or quantity stored in it. In a variable we can store a number, a string, an ID, etc...
- Variables are extremely powerful in programming and are used all the time.

Control Structures

- Control structures determine how the program should respond when given some condition(s) or parameter(s).
- This is usually presented in a programming language using: if...else.

A simple example of how a control structure looks like in Python

```
1 # a simple example in python to show the use of control structures
2
3 if (4 + 5) == 9:
4     print "Programming is great!"
5 else:
6     print "Programming might not be great!"
7
```

Data Types

- Data types determine the type of data or information we are going to be using in our program.
- Commonly used types are integers, floats, and strings.
- Examples:

  - integers : 1,2,45,78,897,...

  - floats : 3.4, 56.7, 8.9, ...

  - strings: “Mike”, “ 你好” , ...

Syntax

- The syntax of a programming language is simply the way the language is written, different rules for how to write the language according to conventions (agreements).
- Just like human languages have grammatical rules to tell us how they should be written and structured, programming languages have rules for their structure (grammar, punctuation).
- Different aspects determine the syntax of a programming language like the spaces ( ), commas(,) , semi-colons (;), colons(:), brackets (()), braces ({}), etc..

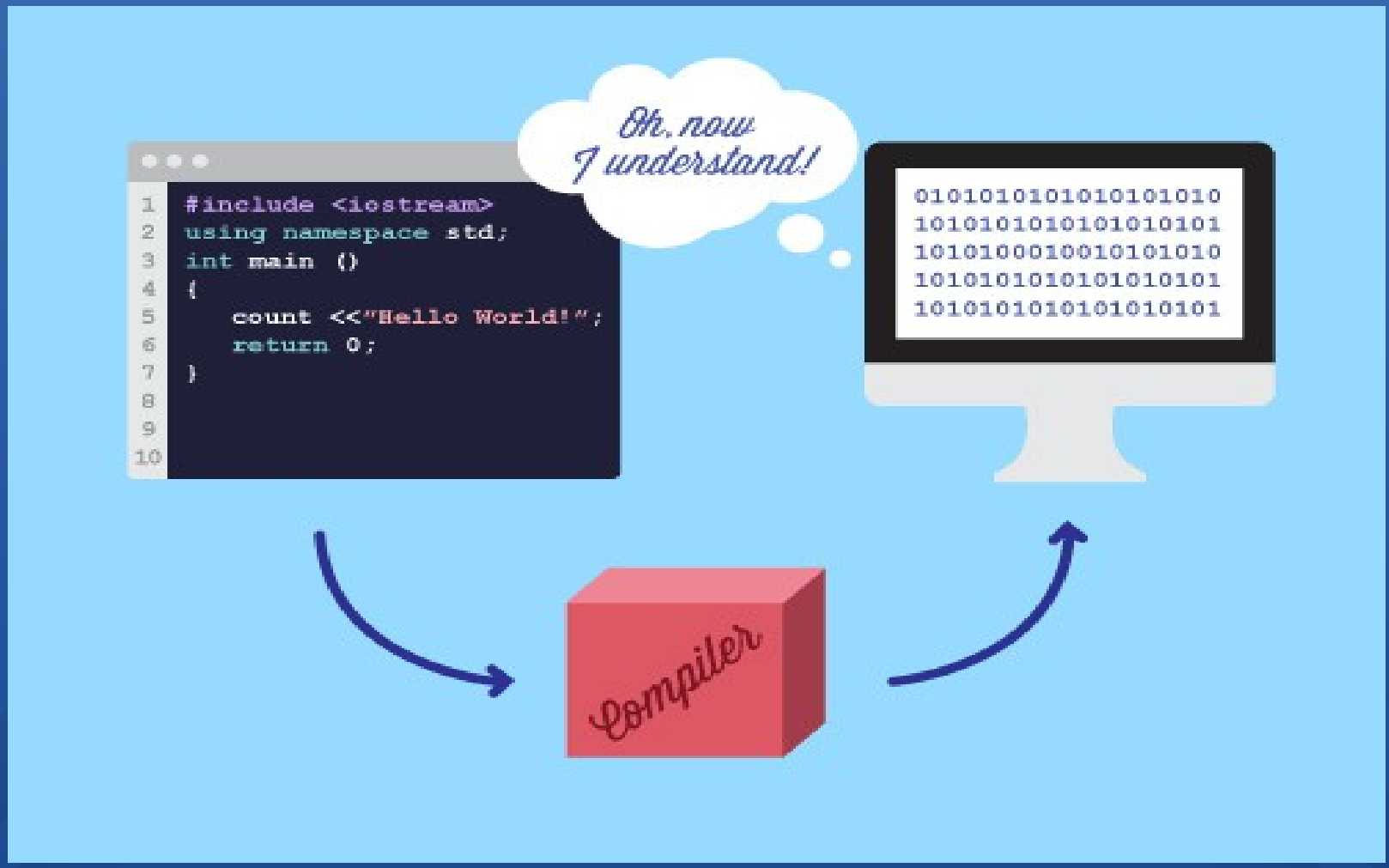
Tools

- There are different tools that are needed to be able to write and run any programming language.
- Examples of these tools are: a debugger, a compiler, an interpreter, a text editor, and a terminal or “shell”
- In our class we might use some of these tools but not all. We will be explaining what each tool does and why it's needed in the coming chapters.

The Compiler

- A compiler is a program that translates a programming language's *statements* (lines of code) into machine code (a special "on" and "off" code to control the computer).
- The machine code is the language that the computer system understands. The machine code interacts directly with the computer's hardware.
- A compiler takes care of changing your source code into zeros and ones that the computer system understands.
- The source code for a program is all the statements (lines of code) you have written. This is what you give to the compiler to make machine code.

How a compiler works



The Interpreter

- An interpreter does almost the same job as a compiler but it doesn't directly change the source code into machine code, it changes the source code into bytecode.
- Bytecode is an in-between “language” that isn’t quite machine code but it isn’t the source code.
- Bytecode can not interact with or directly control the computer system's hardware: it needs to be changed to machine code first to run on the computer system's hardware.

The debugger

- A debugger is a computer program that is used to test and debug other programs(this can be any program).
- The debugger will check for syntax errors (mistakes in the syntax of a program, like spelling things wrong or forgetting punctuation).
- Debugging tools (called debuggers) help identify coding errors at various development stages. Some programming language packages include a facility for checking the code for errors as it is being written.
- Debugging is very useful when we are writing programs, we will see that when we start developing some cool games with python.

The Text Editor

- A text editor is a program that programmers use to write their programs!
- All the “magic” happens in a text editor. The source code for our programs is written inside a text editor. Almost all programming languages make you write your code in a text editor!
- The text editors that programmers use often come with a compiler or interpreter. Some also include a debugger to help us fix our programs.
- Examples of text editors are text mate, sublime text, note pad, etc..
- You can use any text editor you like, they all have the same purpose.

Python Language

The Fun journey begins!!

History of Python

- Python was conceptualized by Guido Van Rossum in the late 1980s.
- Rossum published the first version of the python code (0.9.0) in February 1991 at the CWI (Centrum Wiskunde & Informa tica) in the Netherlands, Amsterdam.
- He chose the name “Python” , since he was a big fun of Monty Python's flying circus.

Why Learn Python?

- It's an easy programming language to pick up quickly.
- You can create fun stuff with python, like games!
- Python is used almost everywhere, there's tons of resources online.
- The syntax is easy to understand.
- You can create just about anything using python.

How is Python Different ?

- Dynamic vs Static Types
- Interpreted vs Compiled
- Prototyping

Dynamic vs Static Types

- A statically typed language requires the programmer to explicitly tell the computer what type of “thing” each data value is. Is it a number? A letter? A list?
- With Python you simply give your variables names and values and according to the value given to a variable python will figure out the type.
- With Python you just need to learn how to declare variables and assign values to them and python will take care of the rest.

Interpreted vs Compiled

- Python is considered an interpreted language. It doesn't have a compiler; the interpreter processes the code line by line and produces a bytecode.
- Because of this in-between state, bytecode is more transferable between operating systems than machine code; this helps Python to be cross-platform (run on many different kinds of computers).
- It is processed line by line, that means you can immediately see the results of your code...you don’t have to wait for the compiler to finish making machine code.

Prototyping

- Because of interpretation, Python and similar languages are used for rapid application development and program prototyping. For example, a simple program can be created in just a few hours and shown to a customer in the same visit.
- Programmers can repeatedly modify the program and see the results quickly. This allows them to try different ideas and see which one is best without investing a lot of time on dead-ends.

  This rapid development is an essential part of programming and it makes programming a fun subject to learn.

Conclusion

- We have covered the basics now. You should know what a programming language is, what a computer language is, and what makes the Python computer language special.
- In the following classes we will look deep into python covering all the key concepts you need to know to be a good python programmer.
- We will learn how to make Games with Pygame, and how to run those games on the Raspberry Pi computer.

Let's start making great stuff