**Practical example on difference between compilers and interpreters**

**Compiler -C**

1. Open a document in notepad ++ and type in the following exactly as given, replacing Yourname with your actual first name

#include <stdio.h>

int main() {

  printf("Hello World from Yourname!");

}

2. In Cdrive, create a folder comp#### where the hashes are your student number

3. Save the program you have just typed in as **hello.c**

4. Then open a command prompt (START - RUN - and type **cmd** then press enter)

5. Type cd .. until you have the c: prompt

6. Type cd comp#### to change to the directory the program was saved in

7. then type this:

gcc -m32 -o runhello.exe hello.c

and press enter

8. Now you should be able to type **runhello** and press enter, and it should print **Hello World from Yourname!** then come back to the command prompt.

8. Open the folder with your programs - and there should be two files there

**Interpreter - Python**

1. From programs, programmingopen Python 2.7

2. File new window

Type in

print("Hello World from Yourname!")

and save it as hello.py in your comp#### folder

3. F5 will “run” the program

4. Open the folder with your programs - note there is still only the hello.py program with the C files

The points to note:

* C program was **COMPILED** to an EXE file that can be directly run as required.  The compiler is called **GCC** (Maybe get them to research this program?)
* Python program HAS to be run using the **python.exe** program (The **INTERPRETER**)
* Python is High-level and only requires one line to do the same thing C needs multiple lines to do.
* Python disadvantage: need python.exe on whatever computer you want to run your python program on (although investigate **py2exe**for a solution)

Further points for compiler and interpreter

1. Go back to your hello.c program

2. Delete the last triangle bracket in the first line. Save and recompile in command line

GCC will pick this up and refuse to compile it.  Compilers are clever enough to spot problems like this.

3. Go back to your python program. Repeat the print statement three times and leave off the last bracket of the last line and try to “Run” it.

The program should stop at this line of error

PS: Extension to the C program... change it like this:

#include <stdio.h>

int main(int argc, char \*argv[]) {

  printf("Hello World! %d", argc);

}

then recompile it and see what changes when running **hello.exe**

Then try running like this:

hello.exe these are arguments

The result should be the number 4 on the end.  Have then investigate what C arguments and **argc** does... FYI: There is always one argument, the name of the program itself, hence the number 1 if you just run hello.exe.  Any distinct words or numbers after that are counted (and put in an array called the argument vector - **argv[]** - but accessing them is a little ugly and probably beyond most of them).

Things to investigate with this:

* argc and argv
* % operator in printf

Python has a way of including arguments, but its also a bit ugly...

**For your external report - you will need to have some screen shots snipped to describe/explain what is happening and show YOUR understanding of the difference between a compiler and interpreter**