C

Ritchie of Bell Labs created the C programming language in 1972. He and Kernagan created the ANSI C standard some time later. The basics of C are something like this. Write source code, compile a object file and then link the object file to libraries to create an executable. C is a very powerful language. For example it can take advantage of direct hardware access but it can also allow the programmer to create a program monster - syntax that is quite legal in C but devastating to the computer. One such example is overwriting system memory.

In Linux, the C compiler is **gcc**. Linux also has a C++ compiler, **g++**. The C compiler needs text input files. To create a new file in Linux use **pico**, for example:

```
# pico hello.c
```

To complie **hello.c** type the following:

```
# gcc hello.c
```

This creates an executable named **a.out** by default.

Now onto the C language itself. As mentioned before C has several different types of files. Source file typically end in **.c**. Header files end in **.h**. Object files end in **.o** and executables vary, for example under DOS they end in **.exe** or **.com**. In Linux no extension is needed. Header files contain declarations for the C preprocessor and source files contain definitions for the C compiler. More about this later. Source files have a reasonably standard format. Here is **hello.c**

```
#include <stdio.h>
// A program to start programming in C
int main(int argc, char * argv[])
{
      printf("Hello world!\n");
      return 0;
}
```

This code, when run, will print "Hello World!" At the top is one #**include** statement. This "includes" the header file, **stdio.h**. Almost all C programs running in Linux (not true for our robot, however) include **stdio.h** because **stdio.h** declares the **printf** function. **main()** is always the entry point into an executable program in C. All blocks of code are enclosed in curly braces **{** and **}**. Functions have parenthesis **()** after them in which arguments go. Executable statements have semicolons **;** after them.

Log onto the supercomputer and edit a file called **hello.c** with **pico** and type in the above code. Once typed in, save, compile, run, and verify it works. Below is what to type on the command line:

```
# pico hello.c
```

Type in the above code, save and exit pico.

```
# gcc hello.c
# ./a.out
Hello world!
```

Linux is case sensitive.

That's really it for C syntax. Almost everything else is made of function calls like **printf()** and curly braced blocks.