An Introductory Guide to C

Sam Moore

January 2020

1 Introduction

Having built a number of guides to Python, a relatively common request is for students to learn C. In fairness, this is completely understandable. C is a highly versatile language, and is extremely popular because of this. I have also personally seen C used for a number of applications in astrophysics, from high-level cosmological simulations to Particle-in-Cell codes modelling plasmas. It is for this reason that I have decided to create this introductory guide to C. I also confess that I am a firm believer that the best way to fully understand something is to teach it, so here goes nothing. As a quick aside, I assume no prior knowledge of Python or coding, but I will make references to Python where it helps.

Let's talk a bit more about C before we delve into the actual coding. C is a relatively old programming language developed in 1972 by Dennis Ritchie as a method of making things on UNIX. The language was later used for the kernel on UNIX, and has gained so much popularity as a language that it is commonly learnt in all sorts of fields.

Let us begin with the most basic program you can write: Hello World!

2 Hello World!

```
#include <stdio.h>
int main() {
  printf{"Hello World!"}
}
return 0;
```

Let us dissect this a bit and discuss some important features of C. First of all, note the line:

```
#include <stdio.h>
```

You should include this at the beginning of all of your coding projects in C. The include statement tells the compiler to call the stdio.h file, in which includes the prototypes of functions such as printf(). Without this line, your compiler wouldn't understand the printf() line.

The next part of the code is the main() function. The main() function is special because it is always the first block of code that is read by the compiler. Since there are no other functions, it is in fact the only function that is read.

Within this function, printf() is used to return a string/integer/etc to stdout. In this case, it returns the string "Hello World!" as an output.

The final part of this code that may come as a surprise to experienced programmers in languages such as Python and Java is the final line:

```
return 0;
```

In C, the main() function is of type int, so it has to return an integer. Typically, 0 is a standard convention for noting that the code was successful in its running. This is also a convention in C++, but whereas in C++ where 0 can be returned implicitly, in C you must explicitly return an integer.

The final thing that I shall note at this point (especially if you are used to Python) is the importance of semi-colons. They are required to end statements.

3 Data Types

4 Sorting

5 Newton-Raphson Method:

Let us look at some numerical computing codes that use C.

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>

// Newton-Raphson Method

double f, df;

// define function
double dB (double x) {
        return (x-5)*exp(x) + 5;
}

// define derivative
```

```
double ddB(double x) {
        return (x-4)*exp(x);
}

int main (void) {
        double xold = 4.1;                // initial guess for NR
        int itrtnmax = 1000;              // max. number of iterations
        double eps = 1*pow(10, -10);      // tolerance for convergence

        // Begin NR:

        int notdone = 1;
        int itrtn = 0;
        double xnew, error, root;

        while (notdone == 1 && itrtn < itrtnmax) {
                f = dB(xold);
                df = ddB(xold);
                if (df != 0) {
                        xnew = xold - f / df;
                        error = fabs(xnew - xold);
                        if (error > eps*xnew) {
                                xold = xnew;
                                itrtn = itrtn + 1;
                        } else {
                                root = xnew;
                                notdone = 0;
                        printf("At iteration %2d, the root is %.13f with itera
                        }
                } else {
                        printf("Derivative is zero\n");
                        break;
                        }
        }
        return 0;
}
```