PH502: Scientific Programming Concepts

Irish Centre for High End Computing (ICHEC)

September 23, 2020

Overview

- In this lecture we will discuss programming blocks or code blocks.
- Code blocks allow the program to be split into functional units, which help with debugging and interpretability.
- Every program must have a main block in C or program block in FORTRAN.
- The compiler needs to know which code line is the first in the program.
- Other code blocks, called functions or subroutines, can be called from the main block.

Functions & Subroutines

- There is only one main program, the other subprograms are called functions/subroutines. Every program starts from the main.
- When a command (in main) uses a function or subroutine, we say that it has been called from main.
- Function Declaration:

```
return_type function_name(argument-list);
```

- Definition:

```
return_type function_name(argument-list)
{
   body of the function
   return var;
}
```

-
```
Subroutine subroutine_name(argument-list)
   body of the subroutine
End Subroutine subroutine_name
```

C Function

```c
#include <stdio.h>
float degtorad(float arg);

int main(void) {
    float degang, radang;
    degang = 10.0;
    radang = degtorad(degang);
    printf(" Deg %f, Rad %f\n",degang,radang);
    return 0;
}

float degtorad(float arg) {
    float pi = 3.1415927;
    return( (pi * arg)/180.0 );
}
```

Fortran Example

```
program fexample
  real (kind=4) :: degang,radang
  degang = 10.0
  call degtorad(degang,radang)
  write(*,*) " Deg ",degang," Rad ",radang
end program fexample

subroutine degtorad(arg,arg2)
  real (kind=4), intent(in) :: arg
  real (kind=4), intent(out) :: arg2
  real (kind=4) :: pi=3.1415927
  arg2=(pi*arg)/180.0
end subroutine degtorad
```

Arguments

ICHEC
Irish Centre for High-End Computing

- In the above example the argument is passed by $r-value$. *degang*'s $r-value$ is copied to that of the dummy argument *arg*.
- *degang* is called the function argument and *arg* is a dummy argument.
- A function must be self contained for it to work properly, it has no access to the variables defined in main (unless passed as a arguments).
- The variables in the function all have different $l-values$ to those in main, even if they have the same name.
- Memory is allocated for these variables each time the function is called and destroyed afterwards, including the dummy arguments.

Example

```c
#include <stdio.h>
float degtorad(float degang);
int main(void) {
    float degang, radang;
    degang = 10.0;
    radang = degtorad(degang);
    printf(" Deg %f, Rad %f\n",degang,radang);
    return 0;
}

float degtorad(float degang) {
    float pi = 3.1415927;
    degang = degang + 10.0; // Not passed to main
    return( (pi * degang)/180.0 );
}
```