PH502: Scientific Programming Concepts

Irish Centre for High End Computing (ICHEC)

September 23, 2020

Overview

ICHEC
Irish Centre for High-End Computing

- In this lecture will we discuss arrays.
- Arrays harness the full power of a computer in that multiple values can be stored in a single variable.
- Together with loops multiple operations (of the same type, also called SIMD), can be performed on an array.
- Multi-dimensional data can be represented by arrays, *e.g.* matrices and 3D data grids.

Arrays

- Up to now the variables we have used hold only a single $r - value$.
- It is possible for one variable name to hold many $r - values$, the variable is then called a variable array or just an array.
- An array is a sequence of data item of the same type.

One dimensional Arrays

- The array length is given in the brackets. All element in an array are of the same type

```
data_type array_name[length];
```

```
data_type :: array_name(length)
```

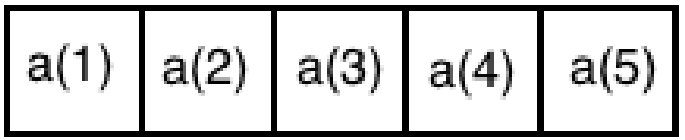
Some examples:

```
int a[10];
float x[100];
char abc[26], ABC[26];
```

```
integer (kind=4) :: a(10)
real (kind=4) :: x(100)
character (len=26) :: abclower
character (len=26) :: absupper
```

Arrays

- Size of array defines the number of elements in an array.
- Individual elements of the array can be accessed using an index. In C the elements are indexed from $0 \rightarrow n-1$ and Fortran from $1 \rightarrow n$, in a length n array.



- The size of array a is 5 times the size of int.
- Below is an example that initialises the array element to its index.

```c
int i,a[5];
for (i=0; i<5; i++) {
   a[i] = i;
}
```

```fortran
integer (kind=4) :: i,a(5)
do i=1,5,1
  a(i) = i
end do
```

- Access to a[6] can cause a fatal error during program execution.

Arrays

Higher Dimension Arrays

- Arrays can be of any dimension depending on the problem.

```
data_type array_name[size1][size2]...[sizeN];
```

```
data_type :: array_name(size1, size2, ..., sizeN)
```

- When using matrices it is convenient to use two dimensional arrays:

```
int i,j;
float A[10][10];
for (i=0; i<10 i++) {
  for (j=0; j<10; j++) {
     if (i == j) {
       A[i][i] = 1.0;
     } else {
       A[i][j] = 0.0;
     }
  }
}
```

```
integer (kind=4) :: i,j
real (kind=4) :: A(10,10)
do i = 1,10,1
  do j = 1,10,1
    if (i .eq. j) then
      A(i,i) = 1.0
    else
      A(i,j) = 0.0
    endif
  end do
end do
```

Arrays in Memory

- In memory a multidimensional array is saved like a 1D object. Thus there is no computational advantage to using them. There maybe a disadvantage if they are accessed badly.
- In C, the array is saved in row major form. That is to say $[i, j]$ and $[i, j+1]$ are contiguous in memory. In the Fortran the opposite is true $[i, j]$ and $[i+1, j]$ are contiguous in memory.

	0	1	2	3
0	1	2	3	4
1	5	6	7	8
2	9	10	11	12

Row-major

1	2	3	4	5	6	7	8	9	10	11	12

Column-major

1	5	9	2	6	10	3	7	11	4	8	12