FORTRAN - DATA TYPES

Fortran provides five intrinsic data types, however, you can derive your own data types as well. The five intrinsic types are:

- Integer type
- Real type
- Complex type
- Logical type
- Character type

Integer Type

The integer types can hold only integer values. The following example extracts the largest value that can be held in a usual four byte integer:

```
program testingInt
implicit none

   integer :: largeval
   print *, huge(largeval)

end program testingInt
```

When you compile and execute the above program it produces the following result:

```
2147483647
```

Note that the **huge** function gives the largest number that can be held by the specific integer data type. You can also specify the number of bytes using the **kind** specifier. The following example demonstrates this:

```
program testingInt
implicit none

   !two byte integer
   integer(kind=2) :: shortval

   !four byte integer
   integer(kind=4) :: longval

   !eight byte integer
   integer(kind=8) :: verylongval

   !sixteen byte integer
   integer(kind=16) :: veryverylongval

   !default integer
   integer :: defval

   print *, huge(shortval)
   print *, huge(longval)
   print *, huge(verylongval)
   print *, huge(veryverylongval)
   print *, huge(defval)

end program testingInt
```

When you compile and execute the above program, it produces the following result:

```
32767
2147483647
9223372036854775807
170141183460469231731687303715884105727
2147483647
```

Real Type

It stores the floating point numbers, such as 2.0, 3.1415, -100.876, etc.

Traditionally there are two different real types, the default **real** type and **double precision** type.

However, Fortran 90/95 provides more control over the precision of real and integer data types through the**kind**specifier, which we will study in the chapter on Numbers.

The following example shows the use of real data type:

```
program division
implicit none

   ! Define real variables
   real :: p, q, realRes

   ! Define integer variables
   integer :: i, j, intRes

   ! Assigning  values
   p = 2.0
   q = 3.0
   i = 2
   j = 3

   ! floating point division
   realRes = p/q
   intRes = i/j

   print *, realRes
   print *, intRes

end program division
```

When you compile and execute the above program it produces the following result:

```
0.666666687
0
```

Complex Type

This is used for storing complex numbers. A complex number has two parts, the real part and the imaginary part. Two consecutive numeric storage units store these two parts.

For example, the complex number $3.0, -5.0$ is equal to 3.0 – 5.0i

We will discuss Complex types in more detail, in the Numbers chapter.

Logical Type

There are only two logical values: **.true.** and **.false.**

Character Type

The character type stores characters and strings. The length of the string can be specified by len specifier. If no length is specified, it is 1.

For example,

```
character (len=40) :: name
name = “Zara Ali”
```

The expression, **name1:4** would give the substring “Zara”.

Implicit Typing

Older versions of Fortran allowed a feature called implicit typing, i.e., you do not have to declare the variables before use. If a variable is not declared, then the first letter of its name will determine its type.

Variable names starting with i, j, k, l, m, or n, are considered to be for integer variable and others are real variables. However, you must declare all the variables as it is good programming practice. For that you start your program with the statement:

```
implicit none
```

This statement turns off implicit typing.