# FORTRAN - NUMERIC PRECISION

 

We have already discussed that, in older versions of Fortran, there were two **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** specifie.

## The Kind Attribute

Different kind of numbers are stored differently inside the computer. The **kind** attribute allows you to specify how a number is stored internally. For example,

```
real, kind = 2 :: a, b, c
real, kind = 4 :: e, f, g
integer, kind = 2 :: i, j, k
integer, kind = 3 :: l, m, n
```

In the above declaration, the real variables e, f and g have more precision than the real variables a, b and c. The integer variables l, m and n, can store larger values and have more digits for storage than the integer variables i, j and k. Although this is machine dependent.

**Example**

```
program kindSpecifier
implicit none

   real(kind = 4) :: a, b, c
   real(kind = 8) :: e, f, g
   integer(kind = 2) :: i, j, k
   integer(kind = 4) :: l, m, n
   integer :: kind_a, kind_i, kind_e, kind_l

   kind_a = kind(a)
   kind_i = kind(i)
   kind_e = kind(e)
   kind_l = kind(l)

   print *,'default kind for real is', kind_a
   print *,'default kind for int is', kind_i
   print *,'extended kind for real is', kind_e
   print *,'default kind for int is', kind_l

end program kindSpecifier
```

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

```
default kind for real is 4
default kind for int is 2
extended kind for real is 8
default kind for int is 4
```

## Inquiring the Size of Variables

There are a number of intrinsic functions that allows you to interrogate the size of numbers.

For example, the **bit_size*i*** intrinsic function specifies the number of bits used for storage. For real numbers, the **precision*x*** intrinsic function, returns the number of decimal digits of precision, while the **range*x*** intrinsic function returns the decimal range of the exponent.

**Example**

```
program getSize
implicit none

   real (kind = 4) :: a
   real (kind = 8) :: b
   integer (kind = 2) :: i
   integer (kind = 4) :: j

   print *,'precision of real(4) =', precision(a)
   print *,'precision of real(8) =', precision(b)

   print *,'range of real(4) =', range(a)
   print *,'range of real(8) =', range(b)


   print *,'maximum exponent of real(4) =' , maxexponent(a)
   print *,'maximum exponent of real(8) =' , maxexponent(b)

   print *,'minimum exponent of real(4) =' , minexponent(a)
   print *,'minimum exponent of real(8) =' , minexponent(b)

   print *,'bits in integer(2) =' , bit_size(i)
   print *,'bits in integer(4) =' , bit_size(j)

end program getSize
```

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

```
precision of real(4) = 6
precision of real(8) = 15
range of real(4) = 37
range of real(8) = 307
maximum exponent of real(4) = 128
maximum exponent of real(8) = 1024
minimum exponent of real(4) = -125
minimum exponent of real(8) = -1021
bits in integer(2) = 16
bits in integer(4) = 32
```

## Obtaining the Kind Value

Fortran provides two more intrinsic functions to obtain the kind value for the required precision of integers and reals:

- selected_int_kind $r$
- selected_real_kind $[p, r]$

The selected_real_kind function returns an integer that is the kind type parameter value necessary for a given decimal precision p and decimal exponent range r. The decimal precision is the number of significant digits, and the decimal exponent range specifies the smallest and largest representable number. The range is thus from 10-r to 10+r.

For example, selected_real_kind $p = 10, r = 99$ returns the kind value needed for a precision of 10 decimal places, and a range of at least 10-99 to 10+99.

### Example

```
program getKind
implicit none

   integer:: i
   i = selected_real_kind (p = 10, r = 99)
   print *,'selected_real_kind (p = 10, r = 99)', i

end program getKind
```

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

```
selected_real_kind (p = 10, r = 99) 8
```