FORTRAN - BASIC INPUT OUTPUT

We have so far seen that we can read data from keyboard using the **read *** statement, and display output to the screen using the **print*** statement, respectively. This form of input-output is **free format** I/O, and it is called **list-directed** input-output.

The free format simple I/O has the form:

```
read(*,*) item1, item2, item3...
print *, item1, item2, item3
write(*,*) item1, item2, item3...
```

However the formatted I/O gives you more flexibility over data transfer.

Formatted Input Output

Formatted input output has the syntax as follows:

```
read fmt, variable_list
print fmt, variable_list
write fmt, variable_list
```

Where,

- fmt is the format specification
- variable-list is a list of the variables to be read from keyboard or written on screen

Format specification defines the way in which formatted data is displayed. It consists of a string, containing a list of **edit descriptors** in parentheses.

An **edit descriptor** specifies the exact format, for example, width, digits after decimal point etc., in which characters and numbers are displayed.

For example:

```
Print "(f6.3)", pi
```

The following table describes the descriptors:

Descriptor	Description	Example
I	This is used for integer output. This takes the form ‘rIw.m’ where the meanings of r, w and m are given in the table below. Integer values are right justified in their fields. If the field width is not large enough to accommodate an integer then the field is filled with asterisks.	print "$3i5$", i, j, k
F	This is used for real number output. This takes the form ‘rFw.d’ where the meanings of r, w and d are given in the table below. Real values are right justified in their fields. If the field width is not large enough to accommodate the real number then the field is filled with asterisks.	print "$f12.3$",pi
E	This is used for real output in exponential notation. The ‘E’ descriptor statement takes the form ‘rEw.d’ where the meanings of r, w and d are given in the table below. Real values are right justified in their	print "$e10.3$",123456.0 gives ‘0.123e+06’

	fields. If the field width is not large enough to accommodate the real number then the field is filled with asterisks. Please note that, to print out a real number with three decimal places a field width of at least ten is needed. One for the sign of the mantissa, two for the zero, four for the mantissa and two for the exponent itself. In general, w ≥ d + 7.	
ES	This is used for real output *scientificnotation*. This takes the form ‘rESw.d’ where the meanings of r, w and d are given in the table below. The ‘E’ descriptor described above differs slightly from the traditional well known ‘scientific notation’. Scientific notation has the mantissa in the range 1.0 to 10.0 unlike the E descriptor which has the mantissa in the range 0.1 to 1.0. Real values are right justified in their fields. If the field width is not large enough to accommodate the real number then the field is filled with asterisks. Here also, the width field must satisfy the expression w ≥ d + 7	print "*es*10.3",123456.0 gives ‘1.235e+05’
A	This is used for character output. This takes the form ‘rAw’ where the meanings of r and w are given in the table below. Character types are right justified in their fields. If the field width is not large enough to accommodate the character string then the field is filled with the first ‘w’ characters of the string.	print "*a*10", str
X	This is used for space output. This takes the form ‘nX’ where ‘n’ is the number of desired spaces.	print "5*x*, *a*10", str
/	Slash descriptor – used to insert blank lines. This takes the form ‘/’ and forces the next data output to be on a new line.	print "/, 5*x*, *a*10", str

Following symbols are used with the format descriptors:

Symbol	Description
c	Column number
d	Number of digits to right of the decimal place for real input or output
m	Minimum number of digits to be displayed
n	Number of spaces to skip
r	Repeat count – the number of times to use a descriptor or group of descriptors
w	Field width – the number of characters to use for the input or output

Example 1

```
program printPi

   pi = 3.141592653589793238

   Print "(f6.3)", pi
   Print "(f10.7)", pi
   Print "(f20.15)", pi
   Print "(e16.4)", pi/100

end program printPi
```

When the above code is compiled and executed, it produces the following result:

```
3.142
3.1415927
3.141592741012573
0.3142E-01
```

Example 2

```
program printName
implicit none

   character (len=15) :: first_name
   print *,' Enter your first name.'
   print *,' Up to 20 characters, please'

   read *,first_name
   print "(1x,a)",first_name

end program printName
```

When the above code is compiled and executed, it produces the following result:
assumetheuserentersthenameZara

```
Enter your first name.
Up to 20 characters, please
Zara
```

Example 3

```
program formattedPrint
implicit none

   real :: c = 1.2786456e-9, d = 0.1234567e3
   integer :: n = 300789, k = 45, i = 2
   character (len=15) :: str="Tutorials Point"

   print "(i6)", k
   print "(i6.3)", k
   print "(3i10)", n, k, i
   print "(i10,i3,i5)", n, k, i
   print "(a15)",str
   print "(f12.3)", d
   print "(e12.4)", c
   print '(/,3x,"n = ",i6, 3x, "d = ",f7.4)', n, d

end program formattedPrint
```

When the above code is compiled and executed, it produces the following result:

```
45
045
300789 45  2
300789 45  2
Tutorials Point
123.457
0.1279E-08

n = 300789 d = *******
```

The Format Statement

The format statement allows you to mix and match character, integer and real output in one statement. The following example demonstrates this:

```
program productDetails
implicit none

   character (len=15) :: name
   integer :: id
   real :: weight
   name = 'Ardupilot'
   id = 1
   weight = 0.08

   print *,' The product details are'

   print 100
   100 format (7x,'Name:', 7x, 'Id:', 1x, 'Weight:')

   print 200, name, id, weight
   200 format(1x, a, 2x, i3, 2x, f5.2)

end program productDetails
```

When the above code is compiled and executed, it produces the following result:

```
The product details are
Name:       Id:    Weight:
Ardupilot   1        0.08
```