Some Fortran Basics (FORTRAN 77)

- start from 7th column (end at 72)
- 6th column (any) character for continuation
- 1-5th column for number
- 1st column C for comment

A. Declarations

         Real: rest of letters

Can change:

INTEGER A, B, C, AA, AB, A5
REAL I, J, I5

Arrays are declared with a dimension statement at the beginning of each program and Subroutine

DIMENSION A(100), B(100,200)
A way to control dimensions is the parameter statement:

```
PARAMETER ( I = 100, J = 200 )
```

```
DIMENSION A ( I ), B ( I , J )
```

to do double precision:

```
DOUBLE PRECISION A , B
```

to have common variables between various subroutines / functions:

```
COMMON / X / A , B
COMMON / Y / C , D
```

(repeat these statements in all parts that the common variables are needed)

** Common blocks have to be of equal length, or else "bad" things happen

(when they are repeated in various subroutines)
B. Do loops, if statements

Do loops repeat an instruction; can be nested

example:

   ICOUNT = 50
   JCOUNT = 60
   SUM = 0.
   DO 200  I = 1, ICOUNT
             DO 100 J = 1, JCOUNT
                   SUM = SUM + A(I,J)
   100 CONTINUE
   200 CONTINUE

If statements can be constructed as follows:

IF (X .GT. 1)
   THEN
       A = 1.
       B = 2.
   ELSE
       A = 2.
       B = 1.
ENDIF
C. Subroutines & Functions

Functions:
1. **build-in functions**
   
   **Example:**
   \[
   A = \sqrt{B^2 + C^2}
   \]

   Other: EXP, ALOG, ABS, SIN, COS, TAN etc

   Prefix of D: double precision (e.g. DSQRT)
   C: complex number

2. **external functions**
   
   Example (function evaluating a factorial)

   ![Code snippet]

   [in the main program:]

   ```
   k = IFACT(j)
   STOP
   END
   ```
[after the main program:]

FUNCTION IFACT(I)

C
C COMPUTES I FACTORIAL
C

IFACT = 1
IF (I.EQ.0) RETURN
DO 10 J = 1, I
    IFACT = IFACT * J
10 CONTINUE
RETURN
END

* note that functions (and subroutines)
use the RETURN statement, whereas main
programs use the STOP statement

- functions can have more than one
variable
- no recursive functions in fortran
For more complicated tasks subroutines are used.

- use CALL statement to call them from the main program:

```
CALL LARGE (D, E, F, G, H)
```

after the main program:

```
SUBROUTINE LARGE (A, B, C, BIG, SUM)
SUM = A + B + C
BIG = A
IF (BIG.LT. B) BIG = B
IF (BIG.LT. C) BIG = C
RETURN
END
```

This subroutine evaluates the sum and the largest of 3 numbers A, B, C
A, B, C are input variables; BIG, SUM are output variables
no variable needs to have the name LARGE
D. Input - Output

_file declarations_: files for input/output can be declared in the main program:

```
OPEN (UNIT=10, FILE = 'PANEL.DAT',
* STATUS = 'UNKNOWN')
OPEN (UNIT=11, FILE = 'PANEL.OUT',
* STATUS = 'UNKNOWN')
```

default unit #5: 5 for input

6 for output

For input use the READ statement; it needs 2 numbers:
- # of unit for the data file
- # of FORMAT statement

Format fields:

*I*  
I: implies integer #
W: implies width (in columns)

*F*.*d*  
F: implies real # in decimal form
W: implies width
d: # of decimal digits
(ignored in input if dot appears)
E.w.d E: implies real # in exp. form
W: implies width
d = # of decimal digits

Example:

READ (10,100) I, A, B, C
100 FORMAT (I5, 2F12.0, E15.0)

in the data file (specified with the open command) the first line should be:
[b implies blank]

bbb50 bbbbb 35.532 bbbbb 150.325 bbb - 4.03125E+02
I5 F12.0 F12.0 E15.0

it is better to use the .0 option in the F and E formats and put the * in the data to avoid mistakes

- unformatted read can also be used, e.g.

READ *, ALPHA
(reads variable alpha directly from the screen, unformatted)
For output use the WRITE statement e.g.

```
WRITE (11, 110) I, A, B, C
110 FORMAT (I5, 2F12.3, E15.5)
```

this will produce the line of the previous page

note that if we use statement # 100 for
the format statement, no decimal digits will
be displaced.

the PRINT statement writes thinks on the
screen; so

```
PRINT 110, I, A, B, C
```

will print the previous line on the screen

using * will write the numbers unformatted:

```
PRINT * I, A, B, C
WRITE (11, *) I, A, B, C
```

you may also write comments

```
WRITE (11, 200)
200 FORMAT (5X, 'CP DISTRIBUTION')
```

5 blank spaces
A more advanced format field is the general format: \[GW.d\]

If it is an integer variable, \[GW.d\] has the same meaning as \[Iw\] in both input & output.

If it is a real value, then:

- input: \[GW.d\] has the same meaning as \[Ew.d\] or \[Fw.d\] depending on whether the input line has E or not typed.

- output: \[d\] is the \# of significant digits.

  - the output will be in the F-form if the variable can be printed in \[w-4\] width.
  (then the variable is printed with exactly 4 blanks at the right end of the field)

  - the output will be in the E-form if the variable cannot be printed in \[w-4\] width.