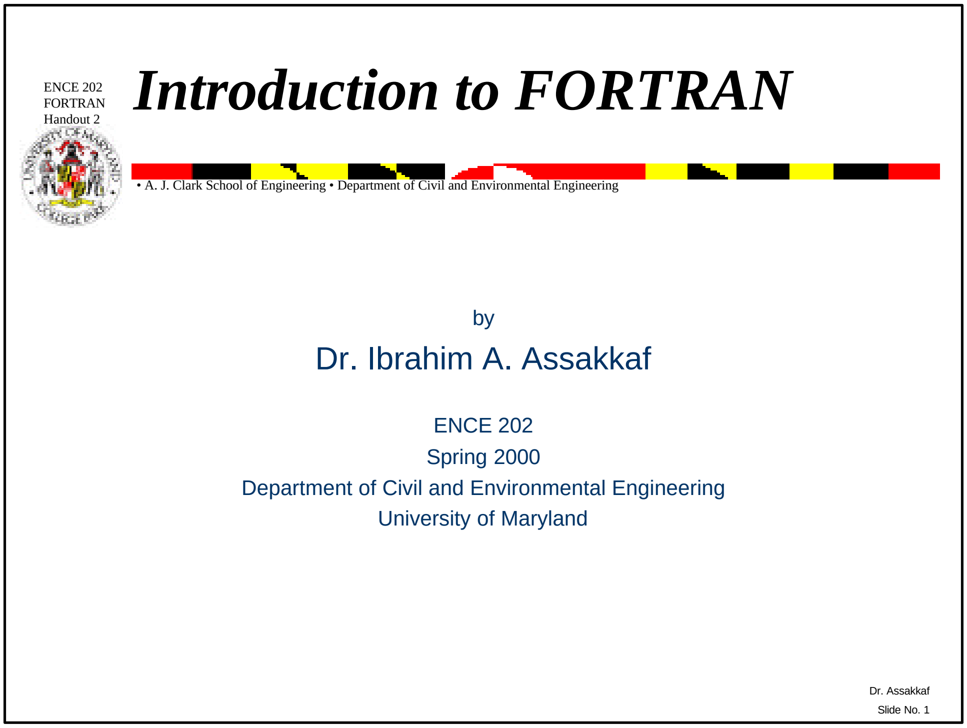


ENCE 202
FORTRAN
Handout 2

Introduction to FORTRAN

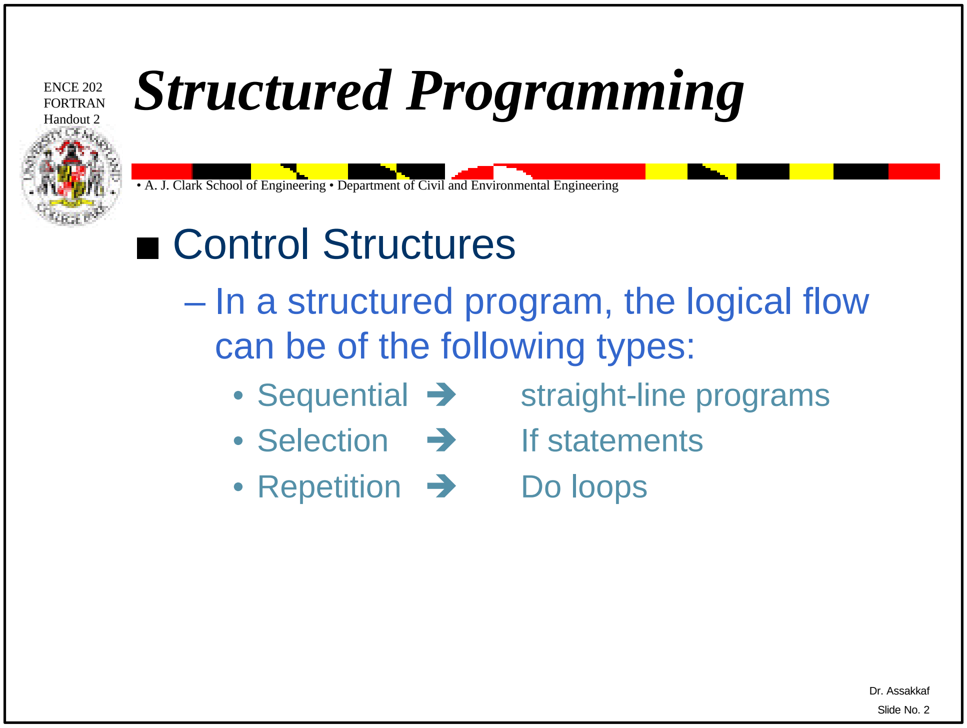
• A. J. Clark School of Engineering • Department of Civil and Environmental Engineering

by

Dr. Ibrahim A. Assakkaf

ENCE 202
Spring 2000
Department of Civil and Environmental Engineering
University of Maryland

Dr. Assakkaf
Slide No. 1



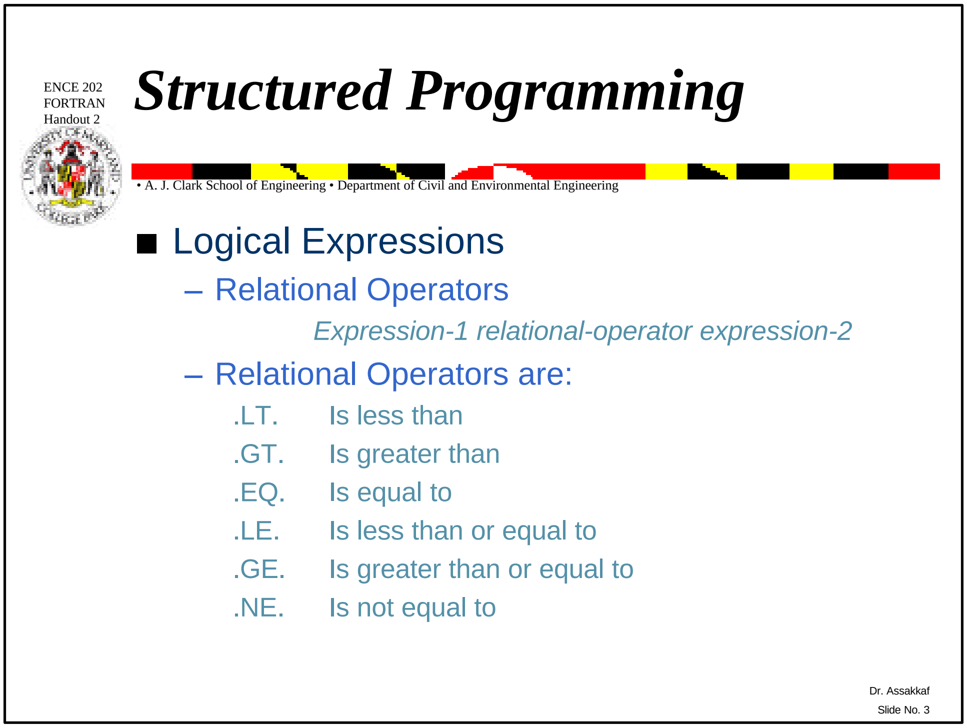
ENCE 202
FORTRAN
Handout 2

Structured Programming

• A. J. Clark School of Engineering • Department of Civil and Environmental Engineering

- Control Structures
  - In a structured program, the logical flow can be of the following types:
    - Sequential ➔ straight-line programs
    - Selection ➔ If statements
    - Repetition ➔ Do loops

Dr. Assakkaf
Slide No. 2



ENCE 202
FORTRAN
Handout 2

Structured Programming

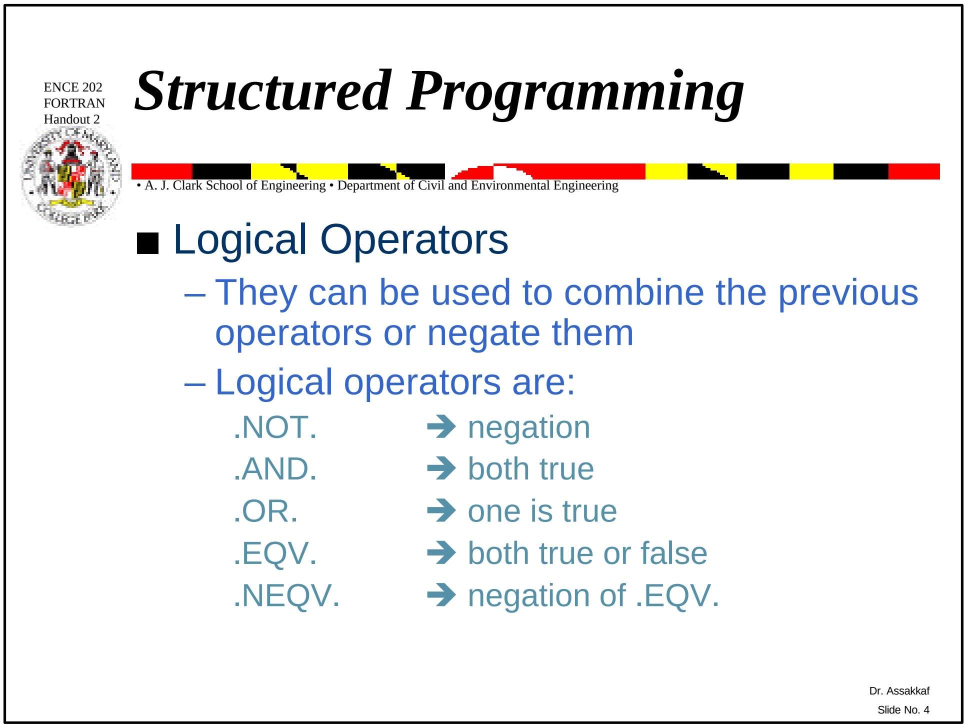
• A. J. Clark School of Engineering • Department of Civil and Environmental Engineering

- Logical Expressions
  - Relational Operators

    Expression-1 relational-operator expression-2
  - Relational Operators are:

.LT.	Is less than
.GT.	Is greater than
.EQ.	Is equal to
.LE.	Is less than or equal to
.GE.	Is greater than or equal to
.NE.	Is not equal to

Dr. Assakkaf
Slide No. 3



ENCE 202
FORTRAN
Handout 2

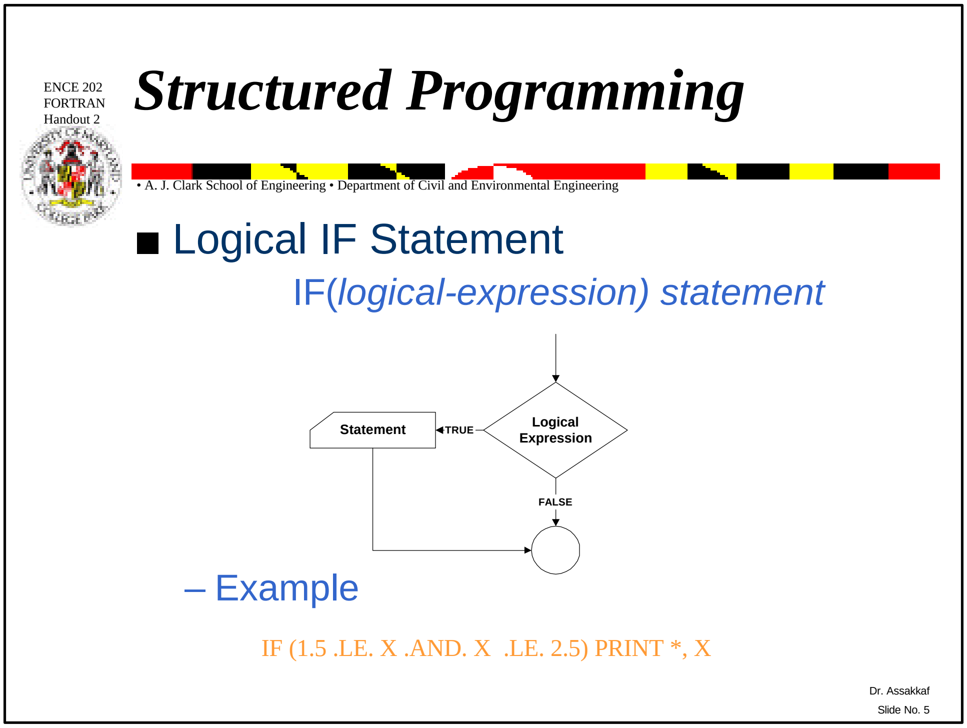
Structured Programming

• A. J. Clark School of Engineering • Department of Civil and Environmental Engineering

- Logical Operators
  - They can be used to combine the previous operators or negate them
  - Logical operators are:

.NOT.	➔ negation
.AND.	➔ both true
.OR.	➔ one is true
.EQV.	➔ both true or false
.NEQV.	➔ negation of .EQV.

Dr. Assakkaf
Slide No. 4



ENCE 202
FORTRAN
Handout 2

Structured Programming

• A. J. Clark School of Engineering • Department of Civil and Environmental Engineering

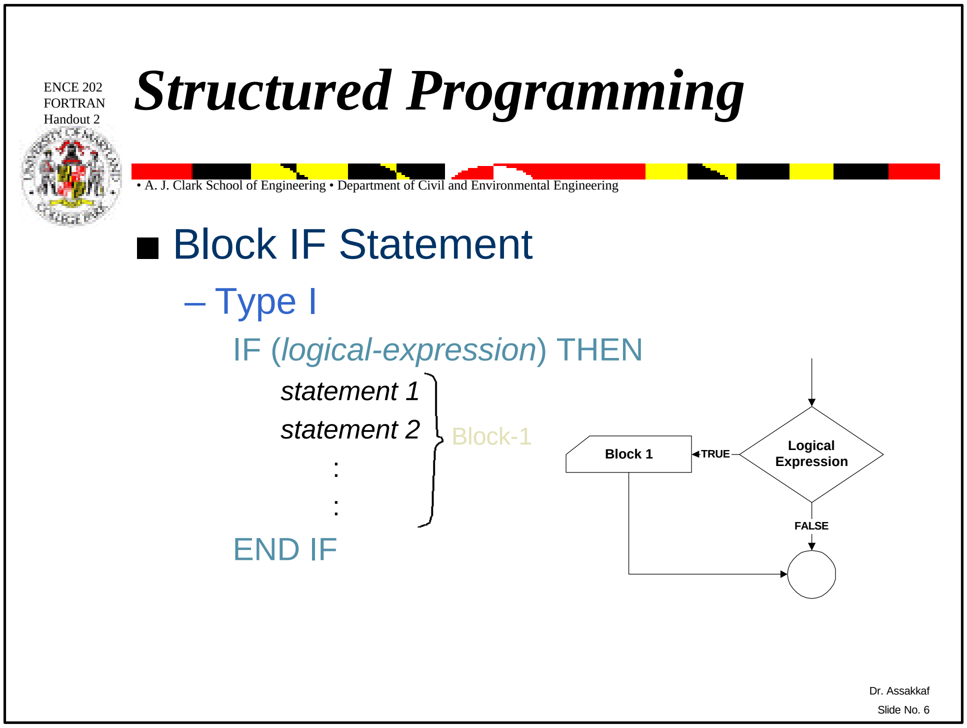
■ Logical IF Statement

IF(*logical-expression) statement*

– Example

```
IF (1.5 .LE. X .AND. X  .LE. 2.5) PRINT *, X
```

Dr. Assakkaf
Slide No. 5



ENCE 202
FORTRAN
Handout 2

Structured Programming

• A. J. Clark School of Engineering • Department of Civil and Environmental Engineering

■ Block IF Statement

– Type I

```
IF (logical-expression) THEN
     statement 1
     statement 2      Block-1
          :
          :
END IF
```

Dr. Assakkaf
Slide No. 6

Structured Programming

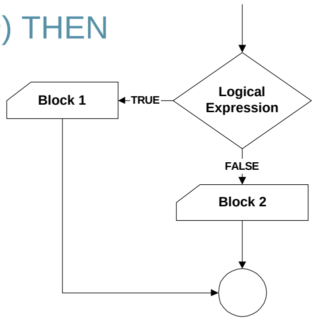
- Block IF Statement
  - Example: Type 1

```
IF (X .GT. 0) THEN
    Y = X * X
    Z = SQRT (X)
    Slope = TAN (Z)
END IF
```

Structured Programming

- Block IF Statement
  - Type II

```
IF (logical-expression) THEN
    Block-1
ELSE
    Block-2
END IF
```



ENCE 202
FORTRAN
Handout 2

Structured Programming

• A. J. Clark School of Engineering • Department of Civil and Environmental Engineering

■ Example: Type II

```
PROGRAM POLLUT
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C Program that reads 3 pollution LEVELS, calculates a pollution INDEX as their
C average, and then displays a “safe condition” message if this index is less than some
C CUTOFF value, otherwise displays a “hazardous condition” message.
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      INTEGER CUTOFF, LEVEL1, LEVEL2, LEVEL3, INDEX
      PARAMETER (CUTOFF = 50)

      PRINT *, ` ENTER 3 POLLUTION READINGS:`
      READ *, LEVEL1, LEVEL2, LEVEL3
      INDEX = (LEVEL! + LEVEL2 + LEVEL3) / 3.0
      IF (INDEX .LT. CUTOFF) THEN
            PRINT *, `SAFE CONDITION`
      ELSE
            PRINT *, `HAZARDOUS CONDITION`
      END IF
      END
```

Sample runs:

```
ENTER 3 POLLUTION READINGS:
55, 39, 48
SAFE CONDITION

ENTER 3 POLUTION READINGS:
68, 49, 57
HAZARDOUS CONDITION
```

ENCE 202
FORTRAN
Handout 2

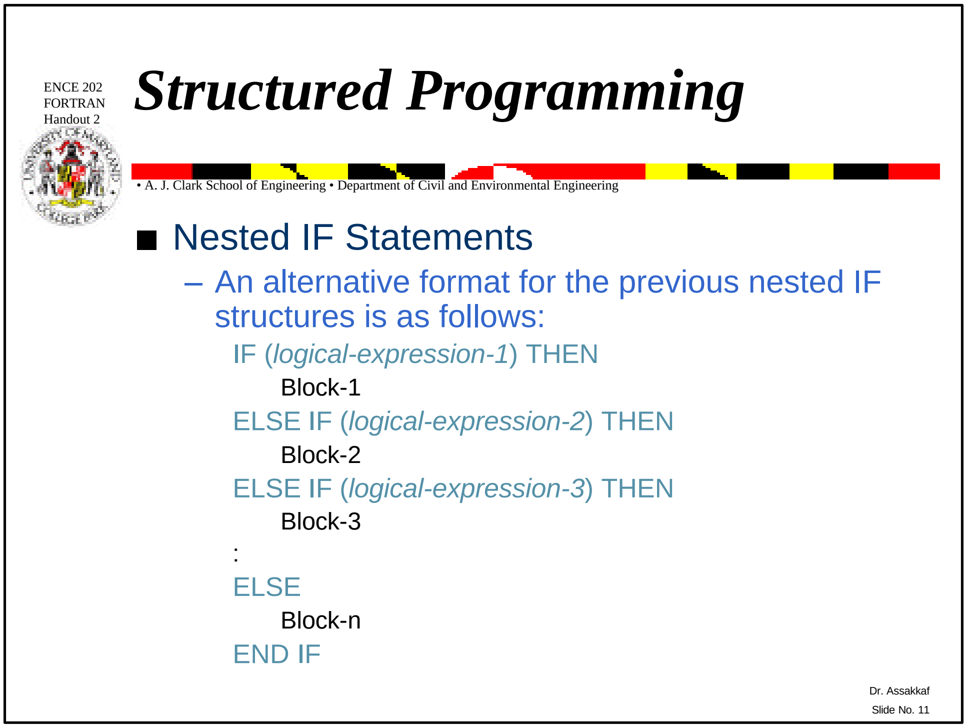
Structured Programming

• A. J. Clark School of Engineering • Department of Civil and Environmental Engineering

■ Nested IF Statements

– They are used for multi-alternative selection structure as follows:

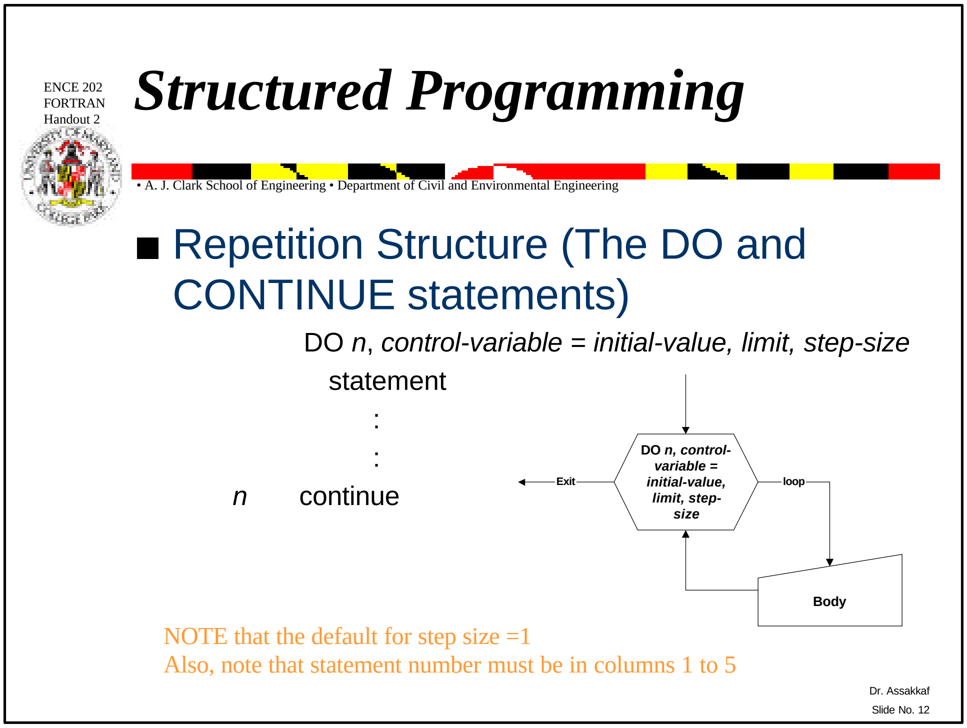
```
IF (logical-expression-1) THEN
      Block-1
ELSE
      IF (logical-expression-2) THEN
            Block-2
      ELSE
            Block-3
      END IF
END IF
```



Structured Programming

- Nested IF Statements
  - An alternative format for the previous nested IF structures is as follows:

```
IF (logical-expression-1) THEN
    Block-1
ELSE IF (logical-expression-2) THEN
    Block-2
ELSE IF (logical-expression-3) THEN
    Block-3
:
ELSE
    Block-n
END IF
```

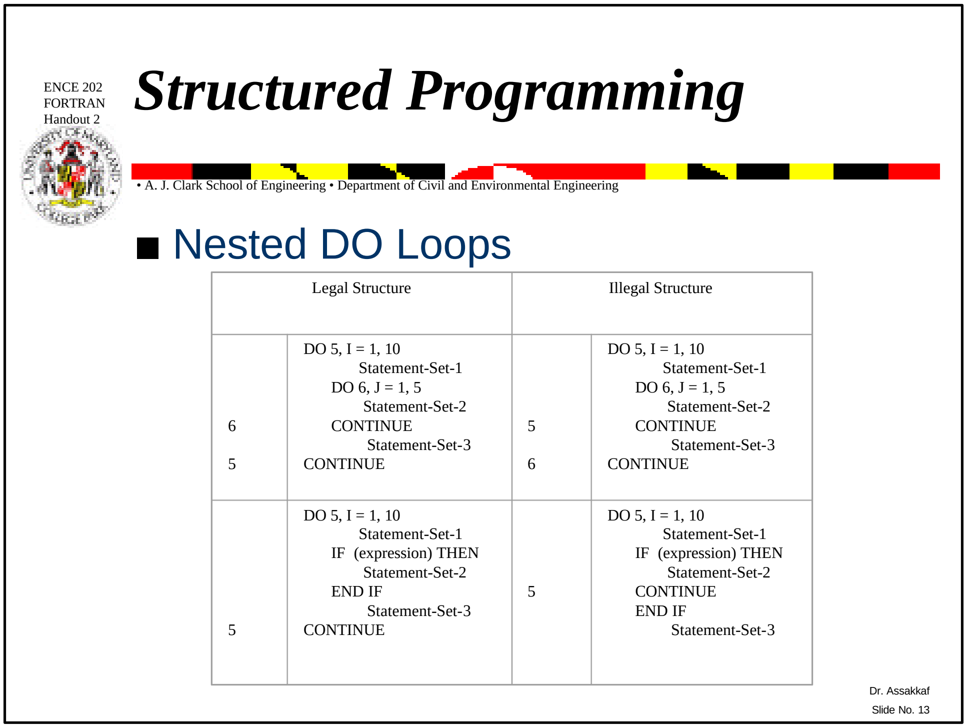


Structured Programming

- Repetition Structure (The DO and CONTINUE statements)

```
        DO n, control-variable = initial-value, limit, step-size
            statement
                :
                :
n       continue
```

NOTE that the default for step size =1
Also, note that statement number must be in columns 1 to 5



ENCE 202
FORTRAN
Handout 2

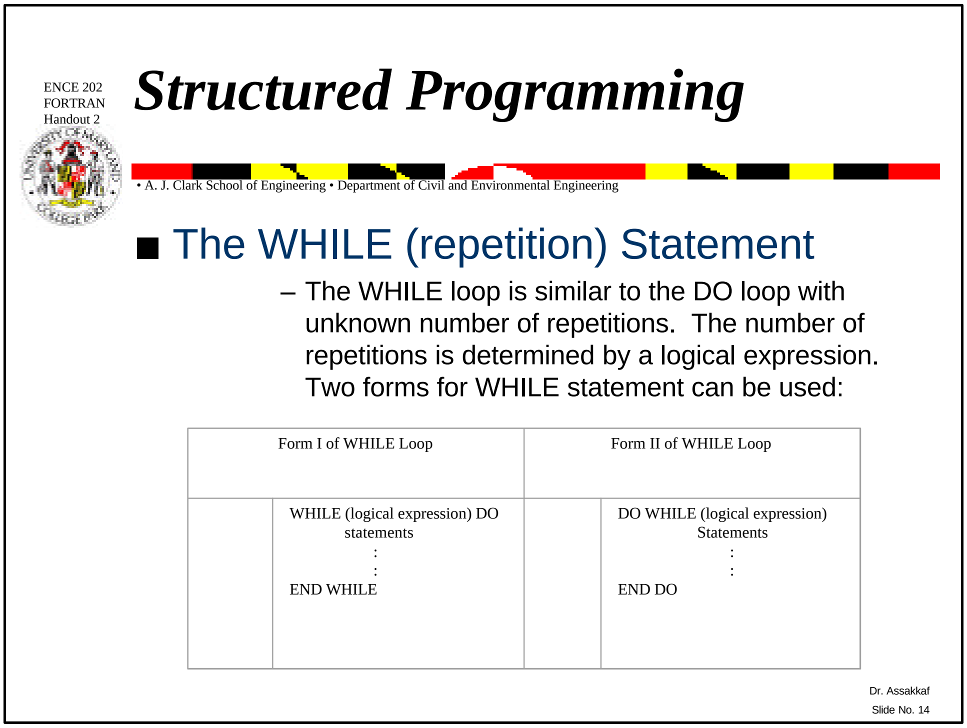
Structured Programming

• A. J. Clark School of Engineering • Department of Civil and Environmental Engineering

■ Nested DO Loops

Legal Structure		Illegal Structure	
6 5	DO 5, I = 1, 10 Statement-Set-1 DO 6, J = 1, 5 Statement-Set-2 CONTINUE Statement-Set-3 CONTINUE	5 6	DO 5, I = 1, 10 Statement-Set-1 DO 6, J = 1, 5 Statement-Set-2 CONTINUE Statement-Set-3 CONTINUE
5	DO 5, I = 1, 10 Statement-Set-1 IF (expression) THEN Statement-Set-2 END IF Statement-Set-3 CONTINUE	5	DO 5, I = 1, 10 Statement-Set-1 IF (expression) THEN Statement-Set-2 CONTINUE END IF Statement-Set-3

Dr. Assakkaf
Slide No. 13



ENCE 202
FORTRAN
Handout 2

Structured Programming

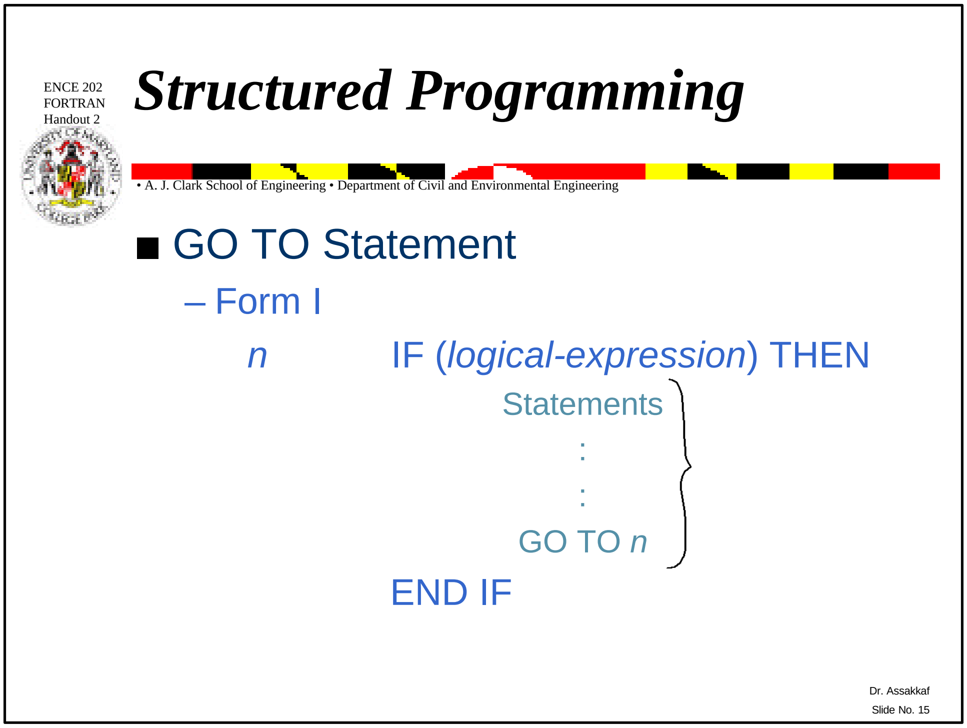
• A. J. Clark School of Engineering • Department of Civil and Environmental Engineering

■ The WHILE (repetition) Statement

– The WHILE loop is similar to the DO loop with unknown number of repetitions. The number of repetitions is determined by a logical expression. Two forms for WHILE statement can be used:

Form I of WHILE Loop		Form II of WHILE Loop	
	WHILE (logical expression) DO statements : : END WHILE		DO WHILE (logical expression) Statements : : END DO

Dr. Assakkaf
Slide No. 14



ENCE 202
FORTRAN
Handout 2

Structured Programming

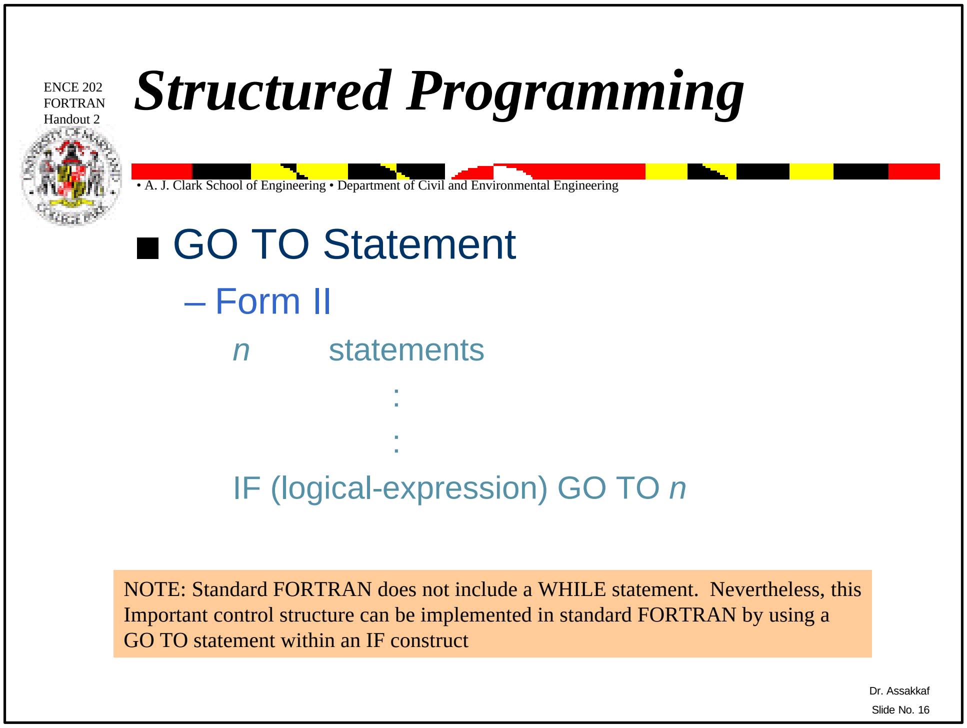
• A. J. Clark School of Engineering • Department of Civil and Environmental Engineering

■ GO TO Statement

– Form I

```
n          IF (logical-expression) THEN
                Statements
                    :
                    :
                 GO TO n
           END IF
```

Dr. Assakkaf
Slide No. 15



ENCE 202
FORTRAN
Handout 2

Structured Programming

• A. J. Clark School of Engineering • Department of Civil and Environmental Engineering

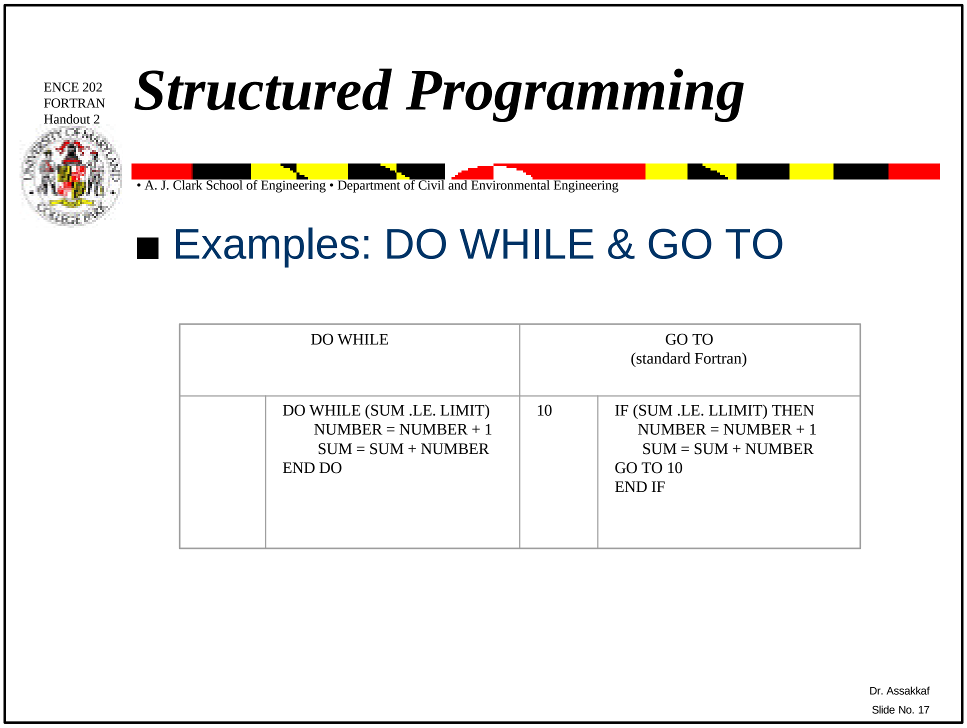
■ GO TO Statement

– Form II

```
n      statements
            :
            :
IF (logical-expression) GO TO n
```

NOTE: Standard FORTRAN does not include a WHILE statement. Nevertheless, this Important control structure can be implemented in standard FORTRAN by using a GO TO statement within an IF construct

Dr. Assakkaf
Slide No. 16

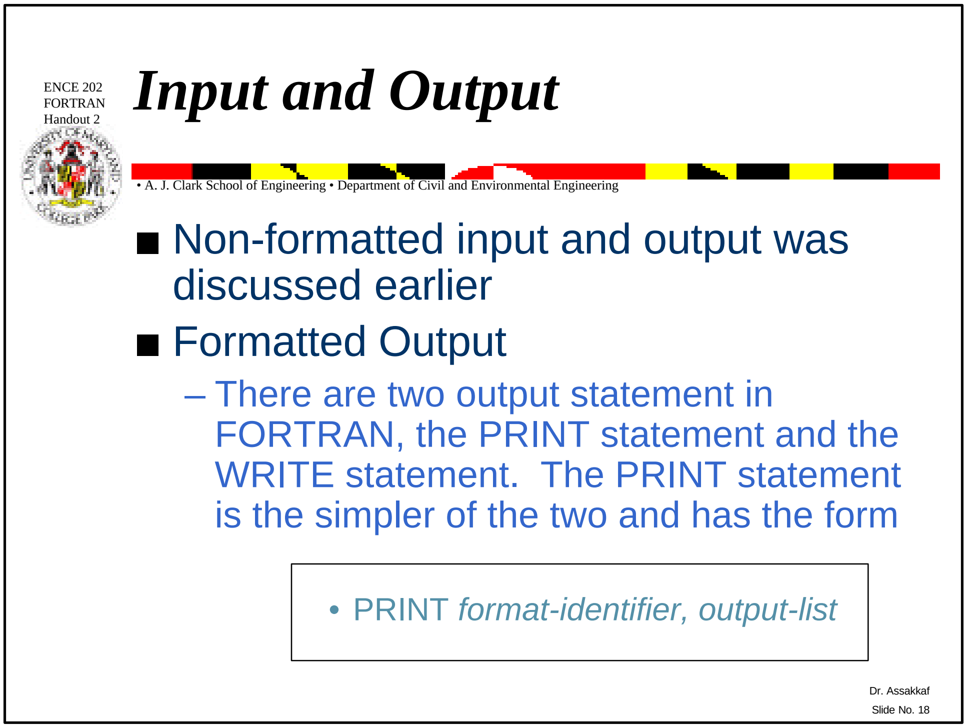


Structured Programming

- Examples: DO WHILE & GO TO

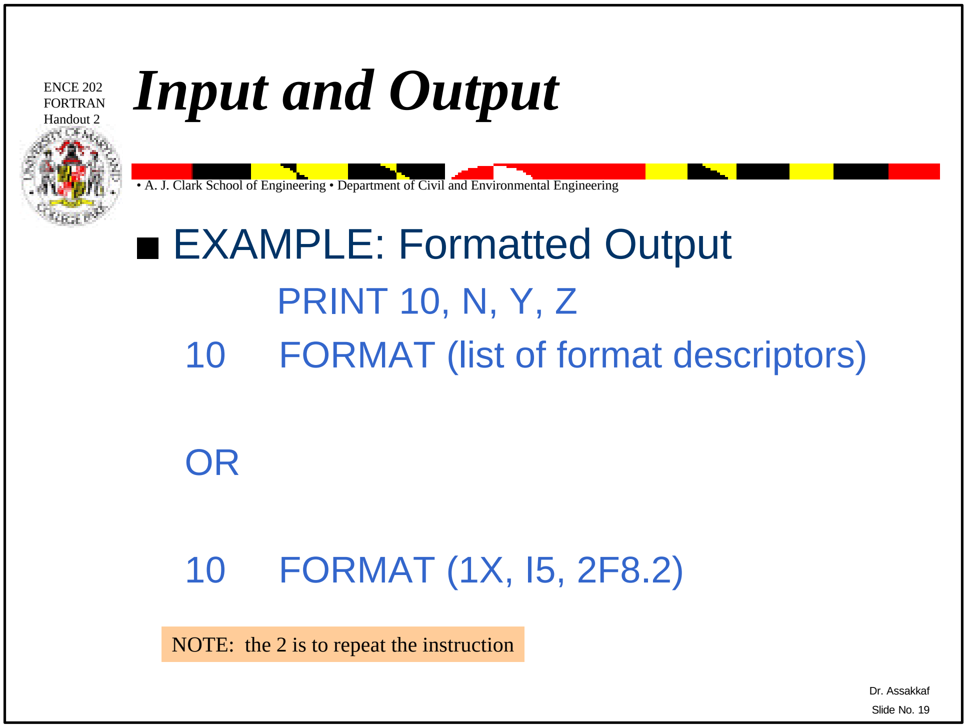
DO WHILE		GO TO (standard Fortran)	

	DO WHILE (SUM .LE. LIMIT)
	NUMBER = NUMBER + 1
	SUM = SUM + NUMBER
	END DO
		10	IF (SUM .LE. LLIMIT) THEN
			NUMBER = NUMBER + 1
			SUM = SUM + NUMBER
			GO TO 10
			END IF



Input and Output

- Non-formatted input and output was discussed earlier
- Formatted Output
  - There are two output statement in FORTRAN, the PRINT statement and the WRITE statement. The PRINT statement is the simpler of the two and has the form
    - PRINT *format-identifier, output-list*



Input and Output

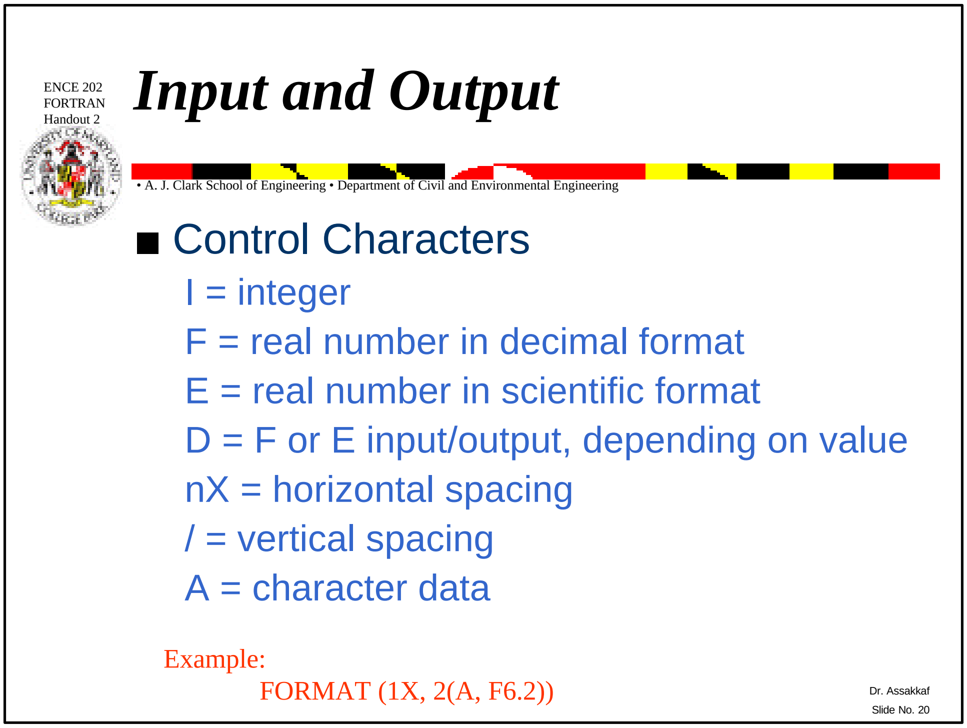
- EXAMPLE: Formatted Output

```
        PRINT 10, N, Y, Z
10      FORMAT (list of format descriptors)

OR

10      FORMAT (1X, I5, 2F8.2)
```

NOTE: the 2 is to repeat the instruction



Input and Output

- Control Characters

I = integer
F = real number in decimal format
E = real number in scientific format
D = F or E input/output, depending on value
nX = horizontal spacing
/ = vertical spacing
A = character data

Example:

```
FORMAT (1X, 2(A, F6.2))
```

ENCE 202
FORTRAN
Handout 2

Input and Output

• A. J. Clark School of Engineering • Department of Civil and Environmental Engineering

- Formatted Input
  - The READ statement is for reading from the default input device (such as the keyboard)

```
READ format-identifier, input-list
```

  - It is common to use non-formatted (or called free format) read as follows:

```
READ *, input-list
```

ENCE 202
FORTRAN
Handout 2

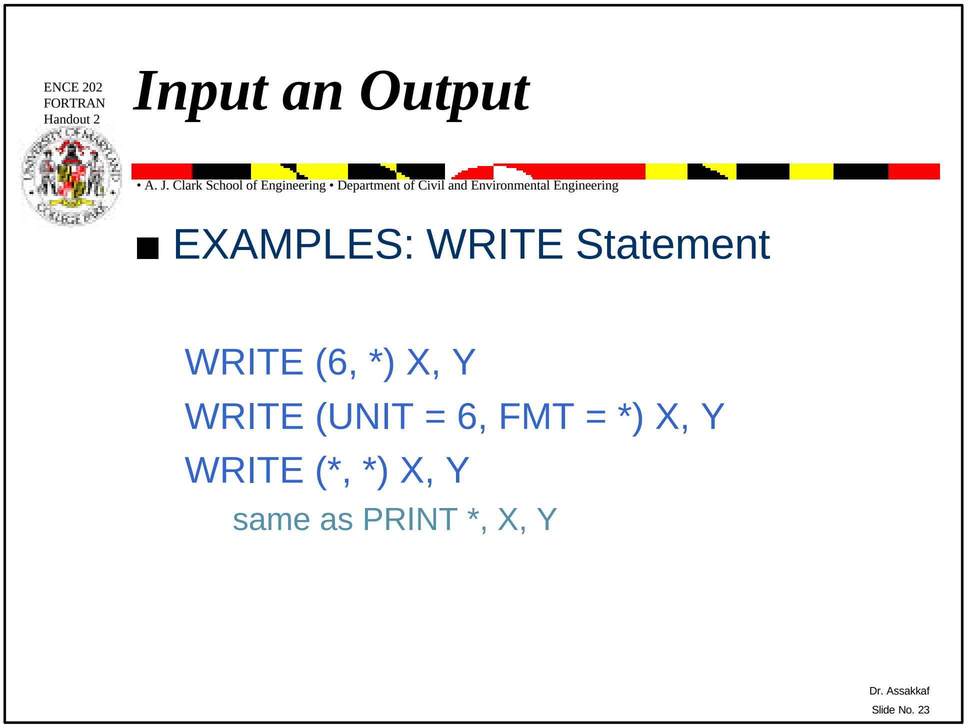
Input and Output

• A. J. Clark School of Engineering • Department of Civil and Environmental Engineering

- The WRITE Statement
  - This statement is used to write to output files

```
WRITE (control-list) output-list
```

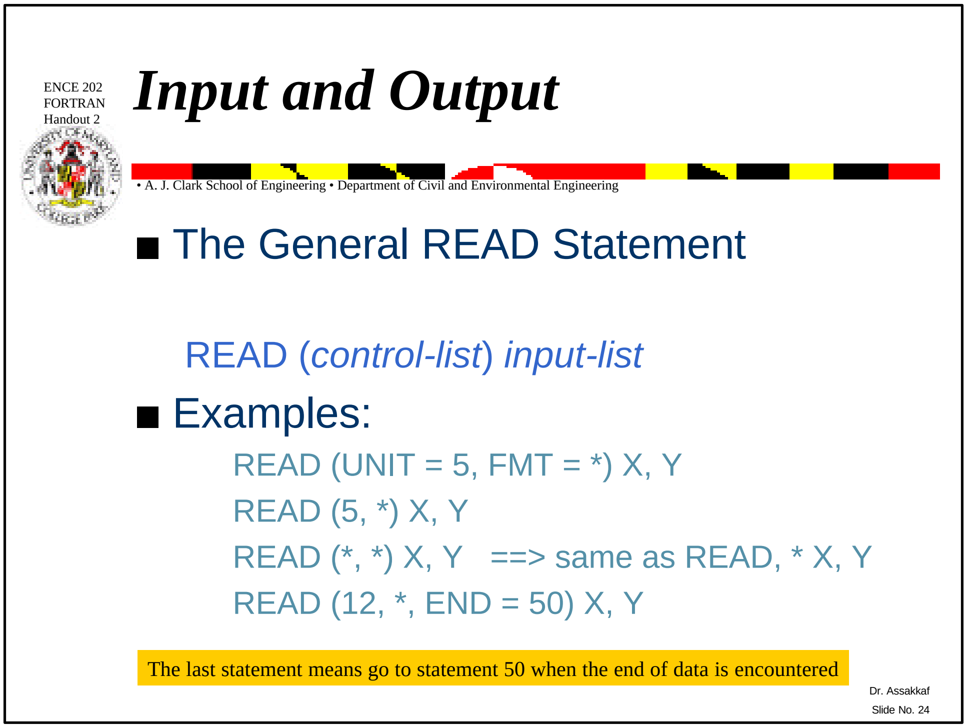
  - Control-list consists of the following:
    - unit-specifier for output device such as printers (*prints on screen)
    - Format-identifier statement (*means free format)



Input an Output

- EXAMPLES: WRITE Statement

```
WRITE (6, *) X, Y
WRITE (UNIT = 6, FMT = *) X, Y
WRITE (*, *) X, Y
    same as PRINT *, X, Y
```



Input and Output

- The General READ Statement

READ (*control-list*) *input-list*

- Examples:

```
READ (UNIT = 5, FMT = *) X, Y
READ (5, *) X, Y
READ (*, *) X, Y   ==> same as READ, * X, Y
READ (12, *, END = 50) X, Y
```

The last statement means go to statement 50 when the end of data is encountered

Input and Output

■ File Processing

– It is common to have a need for large input/output. Files can be on magnetic tapes, disks, or hard drive. Using files requires the following steps:

- To open an existing file or create a new file:

  OPEN (*open-list*)

- To close a file:

  CLOSE (*close-list*)

Input and output

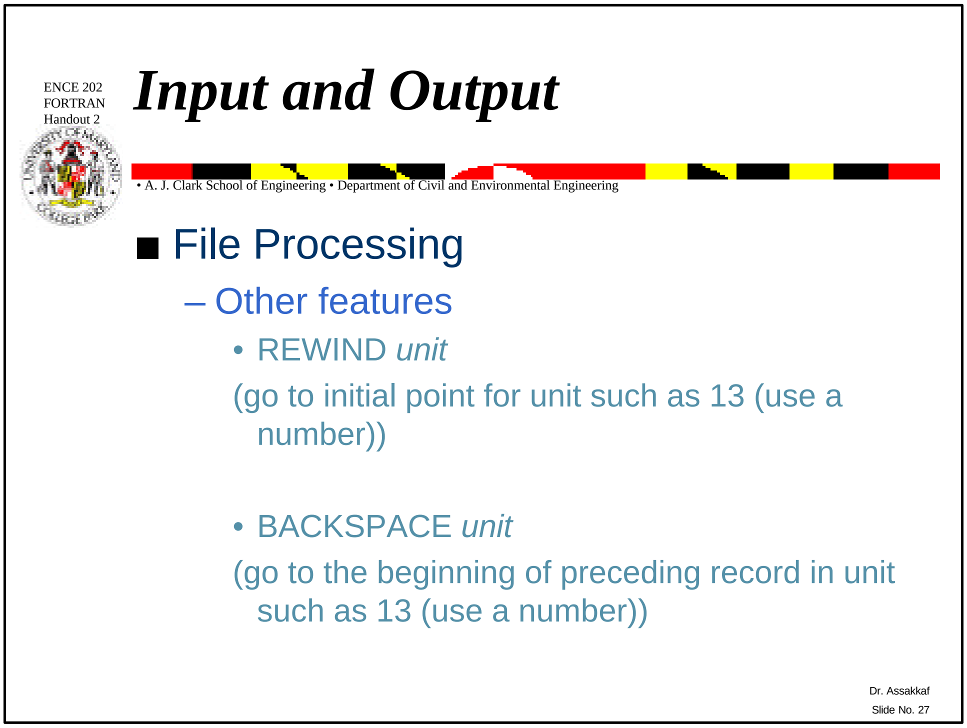
■ Example: File Processing

- OPEN (UNIT = 13, FILE = \`File-name\`, SATAUS = \`NEW\`)

(You may use also STATUS = \`OLD\` to open an existing file)

- CLOSE (13)

(where 13 is the unit number



Input and Output

■ File Processing

- Other features
  - REWIND *unit*

    (go to initial point for unit such as 13 (use a number))

  - BACKSPACE *unit*

    (go to the beginning of preceding record in unit such as 13 (use a number))