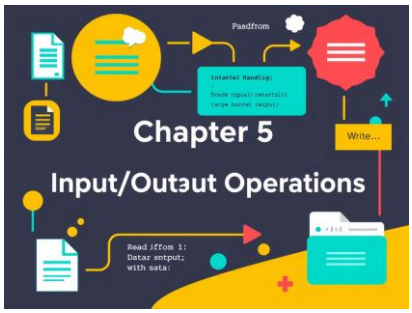
Chapter 5:
Input/Output Operations



Chapter 5: Input/Output Operations in Fortran

5.1 Introduction

Input/Output (I/O) operations in Fortran allow communication between the program and external sources, such as the keyboard, screen, or files. Fortran provides several methods to read data from input devices and write results to output devices.

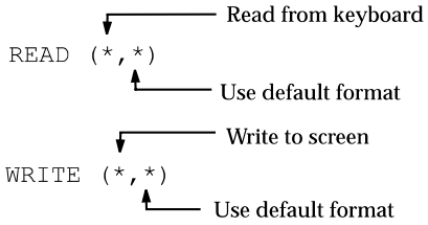
Basic I/O Statements:

- READ is used to read data from the keyboard or a file.
- WRITE is used to display data on the screen or write to a file.

5.2 Understanding WRITE(*,*) and READ(*,*)

In Fortran, the asterisk (*) in WRITE(*,*) and READ(*,*) signifies default input/output devices.

- The **first *** indicates the default unit:
  - WRITE(*,*) writes to the screen (standard output).
  - READ(*,*) reads from the keyboard (standard input).
- The **second *** in WRITE(*,*) or READ(*,*) specifies the default format (free format I/O).



Example:

```
PROGRAM io_example
  IMPLICIT NONE
  INTEGER :: age
  REAL :: height

  PRINT *, "Enter your age and height:"
  READ(*,*) age, height
```

```
    WRITE(*,*) "You entered:", age, "years and", height, "meters."
END PROGRAM io_example
```

Example Input:

25 1.75

Example Output:

You entered: 25 years and 1.75 meters.

5.3 General Format: WRITE(n,m) and READ(n,m)

For more control over I/O formatting, Fortran allows specifying n and m:

- n: Refers to the **unit number** (device or file to be used).
- m: Refers to the **format statement** or label defining the data layout.

Example with Explicit Format:

```
PROGRAM format_example
IMPLICIT NONE
INTEGER :: age
REAL :: height

PRINT *, "Enter your age and height:"
READ(5,*) age, height   ! 5 is the standard input unit

   WRITE(6,100) age, height   ! 6 is the standard output unit
100 FORMAT("Age:", I3, " Height:", F5.2)
END PROGRAM format_example
```

Example Output:

Age: 25 Height: 1.75

5.4 Display Formatting (M: Format Specification)

Fortran provides several format descriptors for output control:

Descriptor	Meaning
Iwd	Integer format (w: width)
Fw.d	Floating-point format (w: width, d: decimals)
Ew.d	Scientific notation format
A	String format
Lw	Logical format
X	Space control (spacing between values)

Example with Various Formats:

```
PROGRAM formatted_output
IMPLICIT NONE
INTEGER :: age
REAL :: salary
age = 30
salary = 5000.75
WRITE(*,100) age, salary
100 FORMAT("Age:", I5, " Salary:", F10.2)
END PROGRAM formatted_output
```

Output:

```
Age:   30  Salary:   5000.75
```

5.5 File Handling (N: Unit Numbers)

Fortran allows reading and writing files using file unit numbers (n).

5.5.1 Creating and Writing to a File

```
PROGRAM file_write
  IMPLICIT NONE
  INTEGER :: file_unit
  file_unit = 10

  OPEN(UNIT=file_unit, FILE="output.txt", STATUS="NEW")
  WRITE(file_unit,*) "This is a test file."
```

```
    WRITE(file_unit,*) "Fortran File Handling."
    CLOSE(file_unit)
END PROGRAM file_write
```

- OPEN initializes the file.
- WRITE(file_unit,*) writes data.
- CLOSE closes the file.

Content of output.txt:

This is a test file.

Fortran File Handling.

5.5.2 Reading from a File

```
PROGRAM file_read
    IMPLICIT NONE
    INTEGER :: file_unit
    CHARACTER(50) :: text
    file_unit = 10

    OPEN(UNIT=file_unit, FILE="output.txt", STATUS="OLD")
    READ(file_unit,*) text
    PRINT *, "File content:", text
    CLOSE(file_unit)
END PROGRAM file_read
```

Example Output:

File content: This is a test file.

5.6 Summary

Feature	Description
WRITE(*,*)	Outputs to the screen using default format
READ(*,*)	Reads from the keyboard using default format
WRITE(n,m), READ(n,m)	Uses unit n and format m for controlled I/O
Format Descriptors	I, F, E, A, L, X for structured output
File Handling	OPEN, WRITE, READ, CLOSE for file operations

Using structured I/O improves data readability and accuracy in Fortran programs.

5.6 Different Forms of OPEN in Fortran

1. **Compact Format**: Uses minimal options to open a file.

   OPEN(UNIT=10, FILE="data.txt")

   - UNIT=10: File unit number.
   - FILE="data.txt": Name of the file to open.

2. **Format with Options**: Allows specifying additional attributes.

   OPEN(UNIT=10, FILE="data.txt", STATUS="OLD", ACTION="READ", FORM="FORMATTED")

   - STATUS="OLD": The file must already exist.
   - ACTION="READ": Opens the file in read-only mode.
   - FORM="FORMATTED": Specifies that the file contains formatted data.

3. **Unformatted Format**: For reading/writing binary files.

   OPEN(UNIT=20, FILE="binary.dat", FORM="UNFORMATTED", ACCESS="DIRECT", RECL=4)

   - FORM="UNFORMATTED": Data is stored in binary format.
   - ACCESS="DIRECT": Direct access to records.
   - RECL=4: Record length in bytes.

These different options allow precise control over file opening and manipulation in Fortran programs.

5.7 Summary

Feature	Description
WRITE(*,*)	Outputs to the screen using default format
READ(*,*)	Reads from the keyboard using default format
WRITE(n,m), READ(n,m)	Uses unit n and format m for controlled I/O
Format Descriptors	I, F, E, A, L, X for structured output
File Handling	OPEN, WRITE, READ, CLOSE for file operations

Using structured I/O improves data readability and accuracy in Fortran programs.