

Topic: FORTRAN Programming

Objectives

- Basics of FORTRAN programming: Read, write

Topic: FORTRAN Programming

Reading from standard input

```
program test
  implicit none

  integer :: s1, s2, s3, total

  write(*,*) 'Enter s1, s2, s3 values'
  read(*,*) s1, s2, s3

  total = s1 + s2 + s3

  write(*,*) ' Sum ', total

end program test
```

```
sandeep $ ./a.out
Enter s1, s2, s3 values
27 23 22
Sum 72
sandeep $ ./a.out
Enter s1, s2, s3 values
27
23
22
Sum 72
sandeep $ █
```

Topic: FORTRAN Programming

Reading/writing from/to standard input/output

```
program test
  implicit none

  integer :: s1, s2, s3, total

  write(6,*) 'Enter s1, s2, s3 values'
  read(5,*) s1, s2, s3

  total = s1 + s2 + s3

  write(*,*) ' Sum ', total

end program test
```

`read(*,*) == read(5,*)`

`write(*,*) == write(6,*)`

Format status – either
unformatted or formatted

Topic: FORTRAN Programming

Read statement

used to read input data from both terminal and text file

```
read(unit, fmt, ERR = err_label, iostat= val, end = end_label)
```

where **unit** is any integer

fmt is format specification mainly used for reading binary files

err_label is statement label where the control is transferred in case of error

end_label is statement label where the control is transferred after reaching the end of file

iostat = value is to check the status of execution of the statement

```
read(1, *)
```

```
read(10000, *)
```

```
read(10 + i, *)
```

```
read(1, *, err=100)
```

```
read(1, *, end=100)
```

```
read(1, *, iostat=value)
```

Topic: FORTRAN Programming

Write statement

used to write output to both terminal and text file

```
write(unit, fmt, ERR = err_label, iostat= val)
```

where **unit** is any integer

fmt is format specification mainly used for reading binary files

err_label is statement label where the control is transferred in case of error

iostat = value is to check the status of execution of the statement

```
write(1, *)  
write(10000, *)  
write(10 + i, *)  
write(1, *, err=100)  
write(1, *, iostat=value)
```

Topic: FORTRAN Programming

Opening files

```
open (unit = number, file = "name", action= action_string, status=status_string,  
iostat=val).
```

unit is any integer

file is filename (string) of any length

action possible values (string) read/write/readwrite -- default is readwrite

status possible values (string) old/new/scratch

iostat = value is to check the status of execution of the statement

```
open(unit=1, file='coord.xyz', action='read')
```

```
open(unit=1, file='coord.xyz', action='readwrite')
```

```
open(unit=1, file='coord.xyz')
```

```
filename = "coord.xyz"
```

```
open(unit=1,file=filename,action=write)
```

```
open(unit=1, file='coord.xyz', status=new, action='read',  
status='new', iostat=x)
```

Topic: FORTRAN Programming

open statement - example

```
program test
  implicit none

  character(len=200) :: filename
  integer :: val,i

  i=1
  filename='coord.xyz'

  open(unit=i,file=filename,action='read')

end program test
```

Topic: FORTRAN Programming

close statement - example

```
program test
  implicit none

  character(len=200) :: filename
  integer :: val,i

  i=1
  filename='coord.xyz'

  open(unit=i,file=filename,action='read')

  close(i)

end program test
```

- Files are automatically closed after the termination of the program
- The close statement flushes the output in the buffer to file and disconnects the unit with the filename
- Any file opened should be closed after writing the output

Topic: FORTRAN Programming

Hands-on

1. Write a Fortran program to check if a file exists in the current directory
2. Write a program to read the data from a text file using 2D array, and change the x-coordinate of each atom by $x+3$. And, write the same data to another file 'output3.xyz'. The program must use DO loops. Input file='input3.xyz'

Topic: FORTRAN Programming

```
program test
  implicit none

  integer, parameter          :: natoms=9

  integer                    :: i, atom
  real(kind=8)               :: atm_coor(natoms,3)
  character(len=10)          :: atm_name(natoms)

! unit = x where x can be any number except 5 and 6

  open(unit=100,file='coordinates.xyz',action='read')
  open(unit=101,file='output.xyz',action='write')

  do atom = 1, natoms
    ??????????????????????

  enddo

  do atom = 1, natoms
    ??????????????????????

  enddo

end program test
```

Writing the data to a file

Topic: FORTRAN Programming

Tips

- Don't worry about declaring variables initially. Identify the main part of the program and start writing
- All real numbers should be in double precision (add d0 in the end), eg. 10.0d0
- Always use indentation, leave black spaces to improve readability
- Always use 'parameter' in case when assigning the values to integer datatype
- Use internal functions to convert datatypes, eg. real(x)
- Read compiler error messages more carefully
- For debugging, use 'write' statement at several places in the program and check for the output

Topic: FORTRAN Programming

FORTRAN – Reading material

- Please go through this FORTRAN program for a quick overview,
 - <https://learnxinyminutes.com/docs/fortran95/>
- Please go through this document for quick overview of FORTRAN
 - <https://www.ldeo.columbia.edu/~mspieg/mmm/Fortran.pdf>
- Book: Computer Programming in Fortran 90 and 95, V. Rajaraman